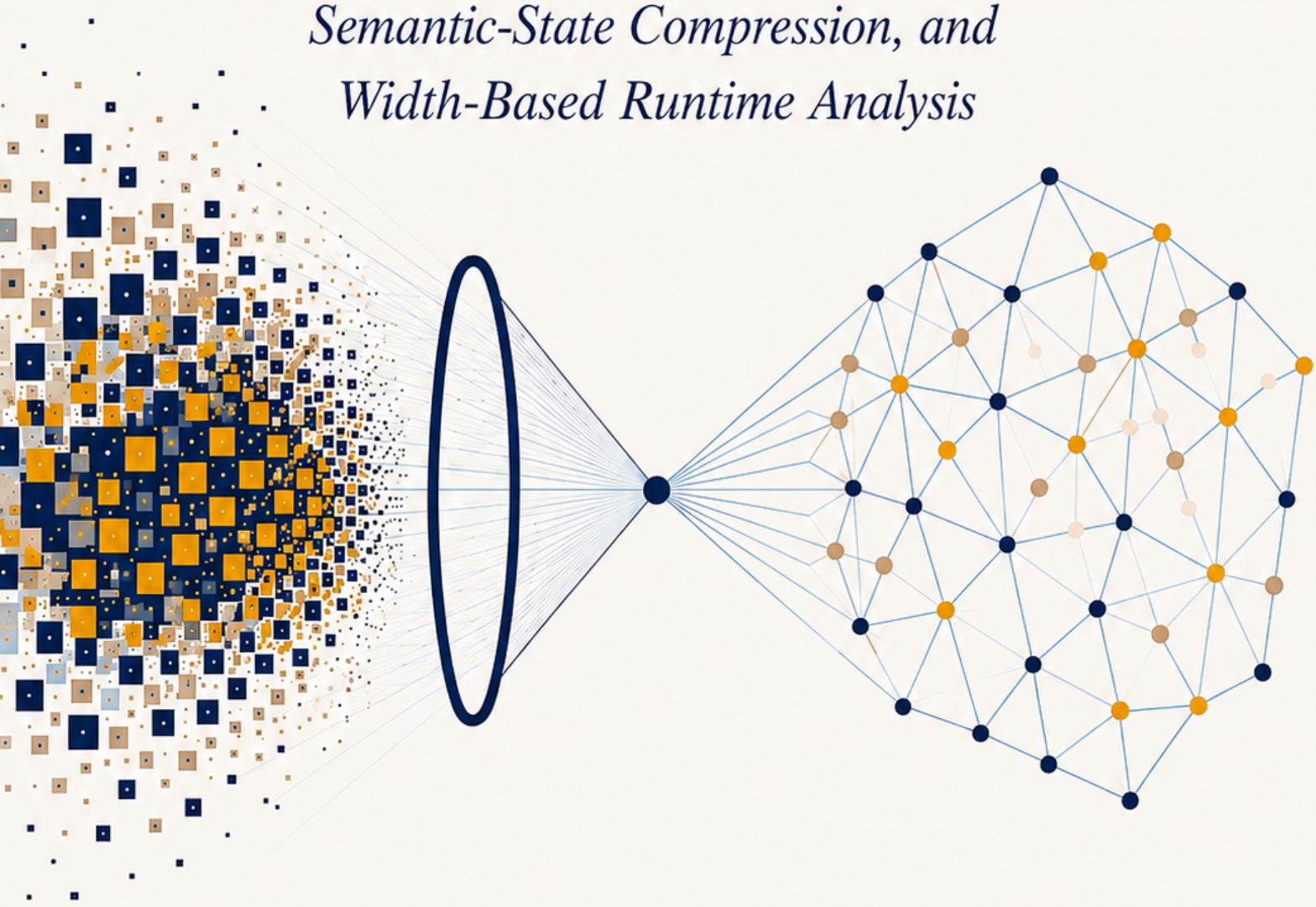
CPR-WIDTH

Controlled Partial Reconstruction Width

A Structural Framework for Active Reconstruction Interfaces, Semantic-State Compression, and Width-Based Runtime Analysis



$$\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \longrightarrow S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \longrightarrow N_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}).$$

REZA HESAMIY

MUNICH, GERMANY

2026

CPR-WIDTH-V2.1

CPR-WIDTH

Controlled Partial Reconstruction Width

A Structural Framework for Active Reconstruction Interfaces, Semantic-State Compression, and Width-Based Runtime Analysis

The Theory of Controlled and Constant CPR-Width

Reza Hesamiy – CPR-WIDTH-V2.1 – Munich, Germany — 2026

Abstract

Controlled Partial Reconstruction Width (*CPR-Width*) is a reconstruction-oriented structural parameter for finite combinatorial problems. Instead of measuring the complete candidate space, CPR-Width measures the maximum number of task-relevant components that must remain simultaneously active during a sound and complete reconstruction process.

Let P be a finite problem instance and let π be an admissible reconstruction ordering. At each reconstruction stage, an active interface records the processed components whose information can still influence an admissible future continuation. For a fixed admissible ordering, the unnormalized CPR-Width is defined as the maximum cardinality of this interface over the complete reconstruction process. A minimum over admissible orderings may subsequently be taken as a separate derived quantity. A fixed offset convention yields the normalized CPR-Width $\boldsymbol{\omega}_{\mathrm{rec}}$ used in the structural chain below.

The framework distinguishes raw interface states, reachable states, and task-relative semantic states obtained through reconstruction-safe signature equivalence, yielding the structural chain $\boldsymbol{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) \to S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) \to N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M})$ (with τ and $\mathfrak{M}$ suppressed when fixed), where S_{CPR} denotes the maximum cardinality of the stage-wise semantic quotient. The chain therefore connects reconstruction width, maximum semantic-state count, and complete computational cost. A full runtime model separately accounts for construction, reduction, semantic transition, reconstruction, verification, and output costs. Four boundary conditions are then stated for explicit representation, polynomial constructibility, sound and complete reconstruction, and polynomial semantic-state and transition-structure control.

Representative realizations are given for Boolean satisfiability, constraint satisfaction, graph coloring, tensor representations, and digital arithmetic, including a dedicated theory of controlled and constant CPR-Width subclasses. Exact finite verification is provided for the full-adder model. For ripple addition, a uniform structural and operation-count result is established for the declared all-input carry-transition schema, while the fixed-target reconstruction-counting experiment of Chapter 8 is treated separately under its own counting task.

This work establishes a formal framework for analyzing controlled partial reconstruction. It does not establish universal tractability, a general polynomial-time algorithm for NP-complete problems, or a proof of $P = NP$.

Keywords: Controlled Partial Reconstruction Width; CPR-Width; active reconstruction interface; semantic-state compression; reconstruction-safe signature equivalence; structural width; finite combinatorial reconstruction; runtime analysis; constraint satisfaction; graph coloring.

Scope and Non-Claims. The results apply only to the explicitly defined reconstruction models and to problem families satisfying all stated boundary conditions and runtime assumptions. A small reconstruction width alone does not imply polynomial runtime. Structural compression alone does not imply a wall-clock advantage. No NP-complete language is proved in this work to satisfy the complete CPR tractability conditions.

Contents

1 Introduction and Motivation 4

2 Controlled Partial Reconstruction Width 5

2.1 Declared CPR Reconstruction Model . . . 5
2.2 Admissible Reconstruction Orderings . . . 6
2.3 Processed Prefix and Unresolved Suffix . . . 6
2.4 Reachable Prefix States . . . 6
2.5 Future Continuations and Task Evaluation . . . 7
2.6 Stage Transitions . . . 7
2.7 Reconstruction-Safe Future Signatures . . . 7
2.7.1 Signature-Update Procedure . . . 7
2.8 Task-Sufficient Active Reconstruction Interface . . . 8
2.9 Interface Factorization . . . 8

2.10 Minimum Active Interface . . . 8
2.11 Why the Componentwise Test Is Insufficient . . . 9
2.12 Controlled Partial Reconstruction Width . . . 9
2.13 Interface Bound . . . 9
2.14 Minimum CPR-Width over Admissible Orderings . . . 10
2.15 Ordering Dependence . . . 10
2.16 Structural Interpretation . . . 10

3 Raw States, Semantic States, and Reconstruction 10
3.1 Raw Interface States . . . 11
3.2 Raw Interface-State Bound . . . 11
3.3 Reachable Interface States . . . 12
3.4 Induced Interface Signatures . . . 12
3.5 Task-Relative Semantic Equivalence . . . 12
3.6 Semantic Quotient . . . 13
3.7 Quotient Transition . . . 13
3.8 Width–State Bound . . . 13
3.9 Binary Semantic-State Dimension . . . 14
3.10 Task Dependence . . . 14
3.11 Semantic Safety . . . 14
3.12 Structural Interpretation . . . 14

4 Complete Runtime Framework 14
4.1 Complete Cost Model . . . 15
4.2 Build and Reduction Cost . . . 15
4.3 Reconstruction Stages and Branching . . . 16
4.4 Semantic Transition Cost . . . 16
4.5 Width-Based Transition Bound . . . 17
4.6 Reconstruction, Verification, and Output Cost . . . 18
4.7 Complete Width-Based Runtime Bounds . . . 18
4.8 Structural Polynomial-State Criterion . . . 19
4.9 Classical Baseline, Count-Level Gain, and Wall-Clock Gain . . . 19
4.10 Structural Runtime Chain and Non-Implications . . . 19
4.11 Conclusion . . . 20

5 Boundary Conditions for CPR Tractability 20
5.1 Overview . . . 20
5.2 Boundary Condition BC1 . . . 21
5.3 Boundary Condition BC2 . . . 21
5.4 Boundary Condition BC3 . . . 21
5.5 Boundary Condition BC4 . . . 22
5.6 CPR-Tractable Families and the Language Class . . . 22
5.7 Uniform Tractability . . . 22
5.8 Conditional Complexity Consequence . . . 23
5.9 Conclusion . . . 23

6 Problem-Class Realizations of CPR-Width 24
6.1 Boolean Satisfiability . . . 24
6.2 Constraint Satisfaction Problems . . . 25
6.3 Graph Coloring . . . 25
6.3.1 Quantitative Verification for C_4 . . . 25
6.3.2 Stage-by-Stage Interface Tracking and Ordering Dependence for C_5 . . . 26
6.4 Common Reconstruction Pattern . . . 26
6.5 Theory of Controlled and Constant CPR-Width . . . 26
6.5.1 Conditional Framework for a Partially Reconstructible Subclass . . . 27
6.5.2 Constant-Width Special Case . . . 27
6.5.3 Meaning of the Theory . . . 27
6.6 Common Reconstruction Chain . . . 28
6.7 Tensor Representations . . . 28
6.8 Full Adder . . . 28
6.9 Synthesis, Scope, and Non-Claims . . . 28

6.10 Transfer to Width-Controlled Reconstruction Beyond Finite Combinatorics . . . 29
6.11 Conclusion . . . 29

7 Adder and Arithmetic Reconstruction **29**
7.1 Full-Adder Map and Hamming-Weight Factorization . . . 30
7.2 Ripple-Adder Function Family . . . 30
7.3 CPR Component Representation and Reachability . . . 30
7.4 Carry Signature, Interface Sufficiency, and Minimality . . . 31
7.5 Evaluation, Semantic-State Control, and Runtime . . . 32
7.5.1 Construction Complexity of the Semantic Quotient . . . 33
7.6 Boundary-Condition Record . . . 34
7.7 Structural Interpretation, Scope, and Conclusion . . . 37
7.7.1 Ripple Addition as a Resource-Separation Example . . . 37

8 Finite Verification Records **37**
8.1 Purpose and Finite Input/Output Spaces . . . 38
8.2 Complete Finite Verification Table . . . 38
8.3 Hamming-Weight Projection and Fibers . . . 39
8.4 Exact Output and Output-Equivalence Verification . . . 40
8.5 Verification Record and Evidence Status . . . 42
8.6 Empirical Illustration: Reconstruction Counting and Interface Growth . . . 42
8.6.1 Implementation, Environment, and Protocol . . . 43
8.6.2 Ripple-Adder Reconstruction Counting . . . 43
8.6.3 Interface Growth on Random 3-SAT . . . 46
8.7 Scope, Evidence Status, Non-Claims, and Conclusion . . . 47

9 Complexity-Theoretic Scope, Open Problems, and Future Research **49**
9.1 Related Work: Classical Structural Width Parameters . . . 50
9.2 Generalizations and Subclasses . . . 50
9.2.1 Possible Generalization to CPR-Tractable Subclasses . . . 50
9.3 Structural and Algorithmic Open Problems . . . 53
9.3.1 Reconstruction Orderings, Width Ratio, and Semantic-State Construction . . . 53
9.3.2 Width-Bound Slack and Relationship to Classical Width Parameters . . . 55
9.3.3 Identification of Additional CPR-Tractable Families . . . 58
9.4 Validation, Comparative Analysis, and Future Research Program . . . 59
9.4.1 Prototype Architecture and Reference Pipeline . . . 59
9.4.2 Comparative Baselines, Width Parameters, and Validation Audit . . . 60
9.4.3 Evidence Classification and Future Research Sequence . . . 62
9.5 Final Conclusion . . . 63

A Core Definitions and Dependency Map **64**
A.1 Declared Reconstruction Setting . . . 64
A.2 Definition Dependency Chain . . . 65
A.3 Operational and Normalized Width . . . 65
A.4 Three Distinct State Levels . . . 65
A.5 Width–State–Runtime Dependency . . . 66
A.6 Complete Runtime Decomposition . . . 66
A.7 Task Relativity . . . 66
A.8 Consistency and Audit Map . . . 66
A.9 Interpretive Summary . . . 66

B Core Definition Register **67**

C Proof Details **68**
C.1 Interface Bound . . . 68
C.2 Raw State Bound . . . 68
C.3 Reachable State Bound . . . 69
C.4 Semantic-State Bound . . . 69
C.5 Quotient Preservation . . . 70

D Audit Checklist **71**

E Figure Register **74**

1 Introduction and Motivation

Classical combinatorial computation is traditionally described in terms of complete candidate spaces. A computational problem is represented by the set of all admissible assignments, configurations, routes, colorings, certificates, or other candidate objects, and algorithmic complexity is usually analyzed with respect to the size of this global search space. From the viewpoint of reconstruction, however, the complete candidate space need not be the quantity that determines the essential computational difficulty. During a reconstruction process, previously processed information may gradually lose its relevance. At every stage, some information can be discarded because it can no longer influence any admissible continuation, whereas other information must remain available. The present manuscript studies the amount of task-relevant information that must remain simultaneously active during a sound and complete reconstruction process. The resulting reconstruction-oriented structural framework is called *Controlled Partial Reconstruction Width (CPR-WIDTH)*. Unlike classical graph-width parameters, CPR-WIDTH is not defined directly on graphs or decomposition trees. Parameters such as treewidth and pathwidth relate structural restrictions of graphs to algorithmic tractability [7, 1], while parameterized complexity separates the influence of input size from additional structural parameters [4, 3]. CPR-WIDTH is introduced in relation to this line of work but measures the active information maintained under a declared reconstruction process. Any formal comparison with classical width parameters must therefore fix the representation, computational task, reconstruction model, and ordering convention. Such comparisons are discussed in Chapter 9. The framework is also related in spirit to dynamic programming and state-space reduction, since these approaches likewise exploit restricted information carried between computational stages. CPR-WIDTH does not, however, identify its structural parameter with a decomposition bag, dynamic-programming table, or complete search frontier. Its defining structural object is a minimum task-sufficient active interface under a declared reconstruction model; reachable and semantic state spaces are analyzed separately. Accordingly, CPR-WIDTH depends on the problem representation, the declared computational task, the reconstruction model, and the chosen admissible ordering. Different reconstruction procedures for the same underlying combinatorial structure may therefore have different reconstruction widths. The theory proceeds through three successive levels:

1. the active reconstruction interface,
2. the corresponding reachable and semantic state spaces,
3. the complete computational cost of controlled reconstruction.

The central structural-runtime relationship is

$$\boldsymbol{\omega_{\mathrm{rec}}}(\boldsymbol{P},\boldsymbol{\pi},\boldsymbol{\tau};\boldsymbol{\mathfrak{M}}) \longrightarrow \boldsymbol{S_{\mathrm{CPR}}}(\boldsymbol{P},\boldsymbol{\pi},\boldsymbol{\tau};\boldsymbol{\mathfrak{M}}) \longrightarrow \boldsymbol{N_{\mathrm{CPR}}}(\boldsymbol{P},\boldsymbol{\pi},\boldsymbol{\tau};\boldsymbol{\mathfrak{M}}) \tag{1.1}$$

The first quantity in this chain is the normalized CPR-Width,

$$\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = \max\left\{0,\, \widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) - 1\right\}, \tag{1.2}$$

obtained from the unnormalized interface width

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}).$$

The distinction is operationally important:

$$\boxed{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}} = \text{maximum active-interface cardinality}}$$

whereas

$$\boxed{\boldsymbol{\omega}_{\mathrm{rec}} = \text{normalized CPR-Width.}}$$

Thus, the two quantities have different mathematical roles even though their notation is intentionally related. The second quantity in Equation (1.1), S_{CPR}, denotes the maximum cardinality of the stage-wise semantic quotient,

$$S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) = \max_i \left|S_i(P,\pi,\tau;\mathfrak{M})\right|,$$

while N_{CPR} represents the complete computational effort of the declared reconstruction model. The arrows in Equation (1.1) express structural and accounting dependencies. They do not imply that a small reconstruction width, or even a polynomially bounded semantic state space, is sufficient by itself to establish polynomial-time tractability. No statement in this manuscript should be interpreted as a proof of $P = NP$, as a universal polynomial-time algorithm for NP-complete problems, or as evidence that structural reduction alone implies tractability. The purpose is to develop a rigorous theory of Controlled Partial Reconstruction Width and its structural, semantic, and computational consequences. Chapter 2 introduces the declared reconstruction model, reconstruction-safe signatures, task-sufficient interfaces, and the formal definition of CPR-WIDTH. Chapter 3 develops the raw, reachable, and semantic-state framework. Chapter 4 establishes the complete runtime model, and Chapter 5 formulates the boundary conditions for uniform CPR tractability. Chapter 6 develops representative structural realizations and

controlled or constant CPR-Width subclasses. Chapter 7 treats arithmetic reconstruction for the ripple-adder family. Chapter 8 provides finite verification and empirical illustrations. Chapter 9 develops the complexity-theoretic scope, comparative questions, open problems, and future research directions.

2 Controlled Partial Reconstruction Width

This chapter introduces the central structural parameter of the CPR-WIDTH framework: how many processed components must remain simultaneously available during a sound, complete, and task-relative reconstruction process. The construction proceeds in the following non-circular order:

$$\mathcal{R}_i^\pi \longrightarrow E_i^\pi(r) \longrightarrow \mathrm{Sig}_{\tau,i}^\pi(r) \longrightarrow B_i^{\pi,\tau} \longrightarrow b_i^{\pi,\tau} \longrightarrow \widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \longrightarrow \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}).$$

Thus, reachable prefix states and their future behavior are defined before the active reconstruction interface is introduced.

> **Running Example.** To keep the abstract definitions of this chapter and Chapters 3–4 anchored to a concrete instance, fix once and for all the four-vertex path graph P_4 with vertices v_1, v_2, v_3, v_4 and edges $\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}$, the task of properly coloring P_4 with $k = 2$ colors, the natural ordering $\pi = (v_1, v_2, v_3, v_4)$, and the structural graph-coloring candidate interface of Chapter 6. Every "Running Example" remark below instantiates the definition just given on this same (P_4, π, τ) so a reader can check each abstract construction by hand before Chapter 6 develops the general graph-coloring realization. Since the running example uses the natural ordering $\pi = (v_1, v_2, v_3, v_4)$, one has $v_{\pi(i)} = v_i$ throughout this example. For $i = 0$, fix $\mathrm{Sig}_{\tau,0}^\pi(\varnothing) = \textsc{start}$, a constant initial signature; this choice is well defined and vacuously satisfies S1–S3 at stage 0, because $\mathcal{R}_0^\pi(P) = \{\varnothing\}$ is a singleton, so the hypotheses of S1–S3 (equality of the signature on two reachable stage-0 states) hold only for the single case $r = r' = \varnothing$. For $1 \le i \le 3$, fix the reconstruction-safe signature $\mathrm{Sig}_{\tau,i}^\pi(r) = r|_{\{v_i\}}$, the color assigned by r to the most recently processed vertex, and $\mathrm{Sig}_{\tau,4}^\pi(r) = \textsc{accept}$, a constant terminal signature; since every reachable prefix of a proper 2-coloring of P_4 extends to a complete proper 2-coloring, this family satisfies S1–S3 for $0 \le i \le 4$, and every numeric interface size quoted below is computed relative to it. This instantiation fixes the *decision* variant of the coloring task, $\tau = \tau_{\mathrm{dec}}$, for which the terminal signature may collapse to the constant value ACCEPT once feasibility is certified. Where a later illustration instead materializes the four color values themselves (Chapter 4), a distinct *witness/exact-reconstruction* task $\tau_{\mathrm{witness}} \neq \tau_{\mathrm{dec}}$ on the same instance (P_4, π) is meant. No equality of the two task-relative widths is inferred merely from the common graph structure. For the witness task, the reconstruction-safe signature family and the task-sufficient interfaces must be justified separately. If color values are emitted irreversibly as output records while reconstruction proceeds, those emitted records are no longer part of the future-active interface, but their materialization and storage/write costs must be charged completely to $N_{\mathrm{Reconstruct}}$ and N_{Output}. If such output materialization is not part of the declared executing procedure, the terminal signature must retain whatever information is still required to reconstruct the complete coloring.

2.1 Declared CPR Reconstruction Model

Let P be a finite problem instance equipped with a finite reconstruction-component representation $V(P) = \{v_1, \ldots, v_n\}$. The component representation is part of the declared reconstruction model; it is not assumed to be a canonical representation of the abstract problem instance. Every component $v \in V(P)$ has a finite nonempty domain D_v, $\varnothing \neq D_v$, $|D_v| < \infty$. The complete assignment space is $\mathcal{X}(P) = \prod_{v \in V(P)} D_v$, and $\mathrm{Sol}(P) \subseteq \mathcal{X}(P)$ denotes the set of complete feasible reconstructions. A declared computational task is denoted by τ; typical tasks include decision, counting, enumeration, optimization, and exact output reconstruction.

Definition 2.1 (Declared CPR Reconstruction Model)**.** A declared CPR reconstruction model for P and task τ is a tuple

$$\mathfrak{M}(P,\tau) = (\mathrm{Adm}_{P,\tau}, \mathfrak{R}, \mathrm{Sig}_\tau),$$

where $\mathrm{Adm}_{P,\tau}$ is the admissibility predicate for reconstruction orderings, $\mathfrak{R}$ specifies the reachable prefix-state spaces $\mathcal{R}_i^\pi(P)$ for every admissible π and stage i, and Sig_τ specifies the reconstruction-safe, task-relative signature family. For every admissible ordering π, the admissible-next-value maps and transition maps are canonical derived objects:

$$\mathfrak{R}, \pi \Longrightarrow \Lambda_i^\pi[\mathfrak{R}], \qquad \delta_i^\pi[\mathfrak{R}].$$

Thus, Λ and δ are not independent degrees of freedom of $\mathfrak{M}(P,\tau)$. The primitive model data are $\mathrm{Adm}_{P,\tau}$, $\mathfrak{R}$, and Sig_τ. All widths below are relative to the fixed representation, task, and model.

Executing CPR procedure. The structural reconstruction model $\mathfrak{M}(P,\tau)$ does not, by itself, specify every algorithmic action whose cost is charged in the complete runtime model. For runtime statements, it is therefore paired with an executing CPR procedure

$$\mathcal{A}_{\mathrm{CPR}}(P,\tau;\mathfrak{M}) = (\mathfrak{M}(P,\tau), \mathcal{R}_{\mathrm{out}}, \mathcal{V}_\tau, \mathcal{O}_\tau),$$

where $\mathcal{R}_{\text{out}}$ is the declared output-reconstruction operator, $\mathcal{V}_\tau$ is the verification rule for the declared task, and $\mathcal{O}_\tau$ is the output representation and output rule. These objects are execution data, not additional primitive degrees of freedom of the structural width model $\mathfrak{M}$. Consequently, the structural quantities $\widehat{\boldsymbol{\omega}}_{\text{rec}}$, $\boldsymbol{\omega}_{\text{rec}}$, and the stage-wise semantic quotients remain relative to $\mathfrak{M}$, whereas $N_{\text{Reconstruct}}$, N_{Verify}, and N_{Output} are costs of the fixed executing procedure $\mathcal{A}_{\text{CPR}}$. Whenever the shorter runtime notation uses only $\mathfrak{M}$ as an argument, the associated executing procedure is understood to have been fixed in advance; no runtime bound may suppress the cost of constructing or applying any of its components. This separation is deliberate: two executing procedures may share the same structural tuple $(\text{Adm}_{P,\tau}, \mathfrak{R}, \text{Sig}_\tau)$ and therefore the same structural CPR-Width while having different reconstruction, verification, or output costs. A tractability claim must consequently satisfy the complete cost accounting of Chapter 4 and the uniform execution requirements of Chapter 5 for the chosen $\mathcal{A}_{\text{CPR}}$.

Admissibility and non-gameability of the declared model. The primitive components $\text{Adm}_{P,\tau}$ and Sig_τ are model-relative but not unrestricted degrees of freedom. They may not encode, presuppose, or conceal the solution of the declared computational task merely in order to reduce the measured CPR-Width or semantic-state cardinality. In particular, a signature whose construction requires solving the original problem, enumerating an exponentially large continuation space, or performing any other unaccounted superpolynomial computation does not yield a tractability conclusion from its small resulting state space. Any admissibility predicate, signature family, ordering-supply rule, semantic reduction, or auxiliary representation used in a family-level CPR tractability claim must satisfy the uniform representation, constructibility, correctness, and semantic-transition requirements of BC1–BC4, and its full construction cost must be charged to the complete operation-count model of Chapter 4. Thus, CPR-Width is relative to a declared model, but a small value obtained only through an expensive, nonuniform, or solution-encoding model choice is not evidence of computational tractability.

Notation roadmap. The principal stage-wise objects used throughout the framework follow the structural chain

$$\mathcal{R}_i^\pi(P) \longrightarrow B_i^{\pi,\tau} \longrightarrow \mathcal{U}_i^{\text{reach}}(P,\pi,\tau;\mathfrak{M}) \longrightarrow \mathcal{S}_i(P,\pi,\tau;\mathfrak{M}),$$

corresponding respectively to reachable prefix states, the active task-sufficient interface, reachable interface states, and semantic quotient states. The associated width quantities satisfy

$$B_i^{\pi,\tau} \longrightarrow b_i^{\pi,\tau} = |B_i^{\pi,\tau}| \longrightarrow \widehat{\boldsymbol{\omega}}_{\text{rec}} \longrightarrow \boldsymbol{\omega}_{\text{rec}},$$

where $B_i^{\pi,\tau}$ is a set, $b_i^{\pi,\tau}$ its cardinality, $\widehat{\boldsymbol{\omega}}_{\text{rec}}$ the maximum operational interface cardinality, and $\boldsymbol{\omega}_{\text{rec}}$ the normalized CPR-Width.

2.2 Admissible Reconstruction Orderings

A reconstruction ordering is a permutation $\pi = (v_{\pi(1)}, v_{\pi(2)}, \ldots, v_{\pi(n)})$.

Definition 2.2 (Admissible Reconstruction Ordering)**.** An ordering π is admissible for the model $\mathfrak{M}(P,\tau)$ if $\text{Adm}_{P,\tau}(\pi) = 1$. The admissibility predicate must encode all declared structural, transition, representation, and task-preservation conditions required by the reconstruction model.

The set of admissible reconstruction orderings is

$$\Pi_{\text{adm}}(P,\tau;\mathfrak{M}) = \{\pi \in \text{Sym}(V(P)) : \text{Adm}_{P,\tau}(\pi) = 1\}.$$

Assume throughout that $\Pi_{\text{adm}}(P,\tau;\mathfrak{M}) \neq \varnothing$. When the task and reconstruction model are fixed, the shorter notation $\Pi_{\text{adm}}(P)$ may be used.

2.3 Processed Prefix and Unresolved Suffix

Let $\pi \in \Pi_{\text{adm}}(P,\tau;\mathfrak{M})$ be fixed. After reconstruction stage i, the processed prefix is $V_i^\pi = \{v_{\pi(1)}, \ldots, v_{\pi(i)}\}$, $0 \le i \le n$, and the unresolved suffix is $\overline{V}_i^\pi = V(P) \setminus V_i^\pi$, with the overline placed over the base symbol V alone so that the prefix V_i^π and the suffix $\overline{V}_i^\pi$ remain typographically distinct. The boundary cases are $V_0^\pi = \varnothing$, $V_n^\pi = V(P)$, and correspondingly $\overline{V}_0^\pi = V(P)$, $\overline{V}_n^\pi = \varnothing$. The complete prefix-assignment space at stage i is $\mathcal{A}_i^\pi(P) = \prod_{v \in V_i^\pi} D_v$, with the empty Cartesian product interpreted as the singleton containing the empty assignment, $\mathcal{A}_0^\pi(P) = \{\varnothing\}$. The complete suffix-assignment space at stage i is $\mathcal{C}_i^\pi(P) = \prod_{v \in \overline{V}_i^\pi} D_v$, so $\mathcal{C}_n^\pi(P) = \{\varnothing\}$.

2.4 Reachable Prefix States

Definition 2.3 (Reachable Prefix-State Space)**.** For every reconstruction stage i, the reachable prefix-state space is a set $\mathcal{R}_i^\pi(P) \subseteq \mathcal{A}_i^\pi(P)$. A prefix assignment belongs to $\mathcal{R}_i^\pi(P)$ if it can be generated after the first i reconstruction steps under the declared transition and admissibility rules of $\mathfrak{M}(P,\tau)$.

Initially, $\mathcal{R}_0^\pi(P) = \{\varnothing\}$. Reachability does not imply extendability: a reachable prefix state may satisfy all local conditions tested so far while still admitting no feasible complete continuation. For $r \in \mathcal{R}_i^\pi(P)$ and $B \subseteq V_i^\pi$, the restriction of r to the components in B is denoted by $r|_B$.

2.5 Future Continuations and Task Evaluation

For a reachable prefix state $r \in \mathcal{R}_i^\pi(P)$, define its feasible continuation set by

$$E_i^\pi(r) = \{s \in \mathcal{C}_i^\pi(P) : r \cup s \in \mathrm{Sol}(P)\}.$$

At a fixed stage i, all sets $E_i^\pi(r)$ are subsets of the same continuation universe $\mathcal{C}_i^\pi(P)$. The declared task τ determines a stage-relative evaluation map $\Theta_{\tau,i}^\pi : \mathcal{R}_i^\pi(P) \times \mathcal{P}(\mathcal{C}_i^\pi(P)) \to Y_{\tau,i}$, and the task-relative future value of a reachable state is $\mathrm{Ans}_{\tau,i}^\pi(r) = \Theta_{\tau,i}^\pi(r, E_i^\pi(r))$. For a decision task, one may use $\mathrm{Ans}_{\mathrm{dec},i}^\pi(r) = \mathbf{1}[E_i^\pi(r) \neq \varnothing]$; for counting, $\mathrm{Ans}_{\mathrm{count},i}^\pi(r) = |E_i^\pi(r)|$; for enumeration, $\mathrm{Ans}_{\mathrm{enum},i}^\pi(r) = \{r \cup s : s \in E_i^\pi(r)\}$. For an optimization task, the map must specify whether the required object is an optimum value, an optimal witness, the set of all optimal witnesses, or another declared optimization certificate.

2.6 Stage Transitions

For every nonterminal stage $i < n$, define the next component by $w_i^\pi = v_{\pi(i+1)}$. For a reachable prefix state $r \in \mathcal{R}_i^\pi(P)$, the set of admissible next values is

$$\Lambda_i^\pi(r) = \left\{a \in D_{w_i^\pi} : r \cup \{w_i^\pi \mapsto a\} \in \mathcal{R}_{i+1}^\pi(P)\right\},$$

and for every $a \in \Lambda_i^\pi(r)$ the transition map is $\delta_i^\pi(r, a) = r \cup \{w_i^\pi \mapsto a\} \in \mathcal{R}_{i+1}^\pi(P)$.

Typing of transition data. In this instance-level model, the second argument of δ_i^π is always the value assigned to the next declared reconstruction component w_i^π. Any fixed data already contained in the encoded problem instance P may of course be consulted when deciding whether that value is admissible or when computing a declared signature update. By contrast, an additional stage label that varies independently across executions may not silently replace the component value a in S2–S3 or in upd_i. Such varying data must either be included explicitly in the declared reconstruction component value/domain or be introduced through a separately defined family-level/transducer construction. Unless such an additional construction is declared, all reachable-prefix spaces, signatures, interfaces, and widths in this chapter remain relative to one fixed problem instance P.

2.7 Reconstruction-Safe Future Signatures

A task value alone may be too coarse for a stage-wise reconstruction process: two decision states may both admit a feasible continuation while requiring different admissible next values. The active interface must therefore preserve not only the current task answer but also the transition behavior required for future reconstruction.

Definition 2.4 (Reconstruction-Safe Future-Signature Family)**.** A family of maps $\mathrm{Sig}_{\tau,i}^\pi : \mathcal{R}_i^\pi(P) \to \Sigma_{\tau,i}^\pi$, $0 \le i \le n$, is called reconstruction-safe if the following conditions hold. **S1 – Task preservation.** For all $r, r' \in \mathcal{R}_i^\pi(P)$, $\mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r')$ implies $\mathrm{Ans}_{\tau,i}^\pi(r) = \mathrm{Ans}_{\tau,i}^\pi(r')$. **S2 – Preservation of admissible next values.** For every $i < n$, $\mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r')$ implies $\Lambda_i^\pi(r) = \Lambda_i^\pi(r')$. **S3 – Successor congruence.** If $\mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r')$ and $a \in \Lambda_i^\pi(r) = \Lambda_i^\pi(r')$, then $\mathrm{Sig}_{\tau,i+1}^\pi(\delta_i^\pi(r,a)) = \mathrm{Sig}_{\tau,i+1}^\pi(\delta_i^\pi(r',a))$.

S1 preserves the declared task value, while S2–S3 preserve the stage-wise transition behavior. The identity signature $\mathrm{Sig}_{\tau,i}^\pi(r) = r$ is always reconstruction-safe, although it provides no compression. A constant signature is admissible only when S1–S3 hold.

2.7.1 Signature-Update Procedure

Conditions S1–S3 constrain which signature families are admissible, but they do not by themselves say how $\mathrm{Sig}_{\tau,i+1}^\pi(\delta_i^\pi(r,a))$ is computed from $\mathrm{Sig}_{\tau,i}^\pi(r)$ once a next value a is chosen. Two computation strategies are possible in principle. *Brute-force recomputation* evaluates $\mathrm{Sig}_{\tau,i+1}^\pi$ on the full successor state $\delta_i^\pi(r,a) = r \cup \{w_i^\pi \mapsto a\}$ from the definition of the signature family directly, in general re-examining the entire processed prefix and, for signatures whose direct evaluation requires explicit continuation information, potentially re-examining a large portion of the continuation space. This strategy is always available whenever a reconstruction-safe signature family is declared, but it does not by itself justify a polynomial bound on $N_{\mathrm{Transition}}$. *Incremental update* instead declares, as an optional realization of the signature family Sig_τ already present in the reconstruction model $\mathfrak{M}$ of Definition 2.1 — not as an additional component of the tuple $\mathfrak{M}(P,\tau)$ itself — a local update function defined on the set of reachable signature/value pairs

$$\mathcal{D}_i^{\pi,\tau} = \left\{(s,a) \in \Sigma_{\tau,i}^\pi \times D_{w_i^\pi} : \exists r \in \mathcal{R}_i^\pi(P),\ s = \mathrm{Sig}_{\tau,i}^\pi(r),\ a \in \Lambda_i^\pi(r)\right\},$$

namely $\mathrm{upd}_i : \mathcal{D}_i^{\pi,\tau} \to \Sigma_{\tau,i+1}^\pi$ satisfying $\mathrm{upd}_i(\mathrm{Sig}_{\tau,i}^\pi(r), a) = \mathrm{Sig}_{\tau,i+1}^\pi(\delta_i^\pi(r,a))$ for every reachable r and every $a \in \Lambda_i^\pi(r)$; condition S3 guarantees this is well defined on $\mathcal{D}_i^{\pi,\tau}$, since the right-hand side depends on r only through $\mathrm{Sig}_{\tau,i}^\pi(r)$. Condition S3 makes no claim about pairs $(s,a) \notin \mathcal{D}_i^{\pi,\tau}$; upd_i need not, and in general does not, extend beyond $\mathcal{D}_i^{\pi,\tau}$. By condition S2, the admissible next-value set $\Lambda_i^\pi(r)$ likewise depends on r only through $\mathrm{Sig}_{\tau,i}^\pi(r)$; the pseudocode below therefore writes $\Lambda_i^\pi(s)$ for $s = \mathrm{Sig}_{\tau,i}^\pi(r)$ without ambiguity. When such an upd_i exists and is itself computable in time polynomial in the encoding length $\ell(P)$ of P (formally defined in Section 4.8; typically $O(1)$ or $O(\log \ell(P))$ per call for some structured realizations

developed later in the manuscript, such as the ripple-adder carry signature of Chapter 7), the signature at stage $i+1$ is obtained without re-examining the processed prefix V_i^π at all, which is the precise sense in which the semantic transition cost of Chapter 4 can avoid brute-force re-evaluation. The following pseudocode summarizes the incremental strategy for one reconstruction step:

```
function AdvanceSignature(s, a, i):
  % s = Sig^π_{τ,i}(r) for the current semantic state, a the chosen next value
  if a ∉ Λ^π_i(s):
    reject a (not an admissible next value)
  s' ← upd_i(s, a)
  % O(1) or problem-class-specific local update, no re-scan of V^π_i
  return s'   % = Sig^π_{τ,i+1}(δ^π_i(r, a)) by construction of upd_i
```

Whether a polynomial-time upd_i exists is a property of the declared problem class and signature family, not a consequence of S1–S3 alone; S1–S3 only guarantee that *some* well-defined update exists, not that it is cheap. Any later problem-class realization that claims a local $O(1)$ update must use exactly the transition typing declared above: the update argument must be the next reconstruction-component value, with any fixed instance-data treated as fixed context, unless a separate family-level/transducer construction has first been defined.

2.8 Task-Sufficient Active Reconstruction Interface

Definition 2.5 (Task-Sufficient Interface)**.** Let i be a reconstruction stage. A set $B \subseteq V_i^\pi$ is called a task-sufficient reconstruction interface if

$$\forall r, r' \in \mathcal{R}_i^\pi(P): \qquad r|_B = r'|_B \Longrightarrow \mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r').$$

This condition guarantees that the complete reconstruction-safe future signature is determined by the values retained on B; all processed components outside B may be forgotten jointly without changing the declared task-relative and transition-relative future behavior. The family of all task-sufficient interfaces at stage i is denoted $\mathfrak{B}_i^{\pi,\tau}(P)$.

Lemma 2.1 (Existence of a Task-Sufficient Interface)**.** *For every reconstruction stage i, $\mathfrak{B}_i^{\pi,\tau}(P) \neq \varnothing$.*

Proof. The complete processed prefix V_i^π is task-sufficient: if $r|_{V_i^\pi} = r'|_{V_i^\pi}$, then $r = r'$, so $\mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r')$. Therefore $V_i^\pi \in \mathfrak{B}_i^{\pi,\tau}(P)$. □

> **Running Example (continued).** On (P_4, π, τ) with $\pi = (v_1, v_2, v_3, v_4)$, after processing v_1 and v_2 (stage 2), the coloring of v_1 no longer affects any admissible completion, since v_1's only neighbor v_2 is already colored; only v_2's color constrains v_3. Hence the selected minimum interface may be taken as $B_2^{\pi,\tau} = \{v_2\}$, while the full processed prefix $V_2^\pi = \{v_1, v_2\}$ is also task-sufficient but is not cardinality-minimal. This is the concrete instance of the general fact, proved in the lemma above, that the full prefix is always task-sufficient even when a much smaller interface already suffices.

2.9 Interface Factorization

For $B \subseteq V_i^\pi$, define the coordinate projection $\rho_{i,B}^\pi : \mathcal{R}_i^\pi(P) \to \prod_{v \in B} D_v$ by $\rho_{i,B}^\pi(r) = r|_B$.

Proposition 2.1 (Interface Factorization)**.** *A set $B \subseteq V_i^\pi$ is task-sufficient if and only if there exists a unique map $\overline{\mathrm{Sig}}_{\tau,i,B}^\pi : \rho_{i,B}^\pi(\mathcal{R}_i^\pi(P)) \to \Sigma_{\tau,i}^\pi$ such that $\mathrm{Sig}_{\tau,i}^\pi = \overline{\mathrm{Sig}}_{\tau,i,B}^\pi \circ \rho_{i,B}^\pi$.*

Proof. Assume first that B is task-sufficient. For $z \in \rho_{i,B}^\pi(\mathcal{R}_i^\pi(P))$, choose a reachable state r satisfying $\rho_{i,B}^\pi(r) = z$ and define $\overline{\mathrm{Sig}}_{\tau,i,B}^\pi(z) = \mathrm{Sig}_{\tau,i}^\pi(r)$. If r' is another reachable state satisfying $\rho_{i,B}^\pi(r') = z$, then $r|_B = r'|_B$, and task sufficiency gives $\mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r')$; hence the map is well defined. Conversely, assume the factorization exists. If $r|_B = r'|_B$, then $\rho_{i,B}^\pi(r) = \rho_{i,B}^\pi(r')$, so $\mathrm{Sig}_{\tau,i}^\pi(r) = \overline{\mathrm{Sig}}_{\tau,i,B}^\pi(\rho_{i,B}^\pi(r)) = \overline{\mathrm{Sig}}_{\tau,i,B}^\pi(\rho_{i,B}^\pi(r')) = \mathrm{Sig}_{\tau,i}^\pi(r')$, so B is task-sufficient. Uniqueness follows because every element of the domain is the projection of at least one reachable prefix state. □

2.10 Minimum Active Interface

A task-sufficient interface need not be unique: different component subsets may preserve exactly the same reconstruction-safe signature. The central width parameter is therefore defined through the minimum required cardinality rather than through a supposedly unique interface.

Definition 2.6 (Minimum Task-Sufficient Interface Size)**.** The minimum task-sufficient interface size at stage i is

$$b_i^{\pi,\tau}(P) = \min\left\{|B| : B \in \mathfrak{B}_i^{\pi,\tau}(P)\right\}.$$

The minimum is taken relative to $\mathfrak{B}_i^{\pi,\tau}(P)$, the family of task-sufficient interfaces for the fixed reconstruction-safe signature family Sig_τ declared by $\mathfrak{M}$; it is not a minimum over all possible reconstruction-safe signature families for (P,τ), and a different declared $\mathfrak{M}$ may in general yield a different value of $b_i^{\pi,\tau}(P)$. The minimum exists because $\mathfrak{B}_i^{\pi,\tau}(P) \neq \varnothing$ and V_i^π is finite. An active reconstruction interface may be chosen as any cardinality-minimal task-sufficient set, $B_i^{\pi,\tau} \in \arg\min_{B \in \mathfrak{B}_i^{\pi,\tau}(P)} |B|$. The minimizing set need not be unique, but every minimizing set has the same cardinality $|B_i^{\pi,\tau}| = b_i^{\pi,\tau}(P)$; accordingly, the phrase "components that must remain active" refers to the minimum cardinality $b_i^{\pi,\tau}(P)$, not to one uniquely determined component set. The complete selected interface sequence is $\mathcal{B}(P,\pi,\tau) = (B_0^{\pi,\tau}, B_1^{\pi,\tau}, \ldots, B_n^{\pi,\tau})$. Since $\mathcal{R}_0^\pi(P) = \{\varnothing\}$, one has $b_0^{\pi,\tau}(P) = 0$ and the initial interface may be selected as $B_0^{\pi,\tau} = \varnothing$. A terminal empty interface is not imposed automatically: it is valid only when all required output information has already been materialized or when the terminal signature is independent of the processed component values.

2.11 Why the Componentwise Test Is Insufficient

The following observation explains why a former componentwise definition cannot serve as the general definition of an active interface.

Lemma 2.2 (Single-Component Necessity Witness)**.** *Let $v \in V_i^\pi$. Assume that there exist states $r, r' \in \mathcal{R}_i^\pi(P)$ such that $r|_{V_i^\pi \setminus \{v\}} = r'|_{V_i^\pi \setminus \{v\}}$, but $\mathrm{Sig}_{\tau,i}^\pi(r) \neq \mathrm{Sig}_{\tau,i}^\pi(r')$. Then every task-sufficient interface contains v.*

Proof. Let B be task-sufficient and suppose $v \notin B$. Then $B \subseteq V_i^\pi \setminus \{v\}$, so $r|_B = r'|_B$. Task sufficiency would imply $\mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r')$, contradicting the hypothesis. Therefore $v \in B$. □

Remark 2.1 (Failure of the Converse)**.** The converse of the preceding lemma is false. Suppose that two processed binary components a, b have only the reachable prefix states $\mathcal{R}_i^\pi(P) = \{00, 11\}$, with $\mathrm{Sig}_{\tau,i}^\pi(00) \neq \mathrm{Sig}_{\tau,i}^\pi(11)$. There are no two reachable states that differ only in a, and none that differ only in b. Nevertheless, the empty interface is not task-sufficient, because it would identify the two states 00 and 11; at least one of the components a or b must be retained. Therefore, a componentwise difference test is a valid necessity witness, but it is not a complete interface definition.

2.12 Controlled Partial Reconstruction Width

The unnormalized interface size is the primary operational quantity.

Definition 2.7 (Unnormalized CPR Interface Width)**.** For a fixed problem P, admissible ordering π, task τ, and reconstruction model $\mathfrak{M}$, define

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = \max_{0 \le i \le n} b_i^{\pi,\tau}(P). \tag{2.1}$$

The value $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})$ is the minimum number of components that must remain simultaneously active at the widest reconstruction stage, under the fixed representation, task, reconstruction model, and reconstruction-safe signature family declared by $\mathfrak{M}$; it is not an intrinsic property of P alone. This unnormalized quantity is used directly in every bound proved in Chapters 3–4. Accordingly, whenever a statement counts components that are physically or logically retained at a stage, $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}$, rather than the normalized reporting parameter $\boldsymbol{\omega}_{\mathrm{rec}}$, is the operative width. The normalized parameter is used only through the explicit relation between the two quantities.

Definition 2.8 (Controlled Partial Reconstruction Width)**.** The normalized Controlled Partial Reconstruction Width is

$$\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = \max\left\{0,\, \widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) - 1\right\}. \tag{2.2}$$

The two width quantities have distinct roles. The unnormalized quantity $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}$ is the operational maximum active-interface cardinality used directly in the state-count and runtime bounds of Chapters 3–4. The normalized quantity $\boldsymbol{\omega}_{\mathrm{rec}}$ is the reported width parameter. The minus-one convention asserts no formal identity with a classical width parameter. Hence $\boldsymbol{\omega}_{\mathrm{rec}} = 0$ is a normalized report and must not be read as "no active component": it may correspond either to an empty maximum interface or to a one-component maximum interface. Whenever that distinction matters, both $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}$ and $\boldsymbol{\omega}_{\mathrm{rec}}$ are stated. When τ and $\mathfrak{M}$ are fixed, the shorter forms $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi)$ and $\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi)$ may be used.

> **Running Example (continued).** On (P_4,π,τ), tracking the minimal task-sufficient interface stage by stage gives $b_1^{\pi,\tau} = |\{v_1\}| = 1$, $b_2^{\pi,\tau} = |\{v_2\}| = 1$, $b_3^{\pi,\tau} = |\{v_3\}| = 1$, and $b_4^{\pi,\tau} = 0$ once the suffix is empty, so $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P_4,\pi,\tau;\mathfrak{M}) = 1$ and $\boldsymbol{\omega}_{\mathrm{rec}}(P_4,\pi,\tau;\mathfrak{M}) = \max\{0, 1-1\} = 0$. This small numeric gap between the unnormalized and normalized width is exactly the distinction emphasized above: one component (the most recently colored vertex) genuinely remains active at every internal stage, even though the normalized width reads zero.

2.13 Interface Bound

Lemma 2.3 (Interface Bound)**.** *For every stage i, $b_i^{\pi,\tau}(P) \le \widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})$. Moreover, $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \le \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) + 1$. Consequently, for every selected minimum active interface, $|B_i^{\pi,\tau}| \le \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) + 1$.*

Proof. The first inequality follows immediately from $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = \max_{0\le j\le n} b_j^{\pi,\tau}(P)$. Let $M = \widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})$. If $M=0$, then $M = 0 \le 1 = \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1$. If $M \ge 1$, then $\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = M-1$, and therefore $M = \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1$. Finally, $|B_i^{\pi,\tau}| = b_i^{\pi,\tau}(P)$, because $B_i^{\pi,\tau}$ is cardinality-minimal. □

2.14 Minimum CPR-Width over Admissible Orderings

Because $\Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M}) \neq \varnothing$, the following minimum is well defined.

Definition 2.9 (Minimum Controlled Partial Reconstruction Width)**.** The minimum Controlled Partial Reconstruction Width of P, relative to task τ and model $\mathfrak{M}$, is

$$\boldsymbol{\omega}_{\mathrm{rec}}(P,\tau;\mathfrak{M}) = \min_{\pi\in\Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})} \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}). \tag{2.3}$$

The corresponding unnormalized minimum is $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\tau;\mathfrak{M}) = \min_{\pi\in\Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})} \widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})$. When the task and reconstruction model are fixed, these quantities may be written as $\boldsymbol{\omega}_{\mathrm{rec}}(P)$ and $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P)$. The minimization over orderings does not silently alter the component representation $V(P)$, the local domains D_v, the declared task τ, the admissibility predicate, the reachable-state rules, the transition system, or the reconstruction-safe future-signature family: only the admissible reconstruction ordering is varied.

2.15 Ordering Dependence

Proposition 2.2 (Ordering Dependence)**.** *There exist a finite problem instance P, a fixed task τ, a fixed reconstruction model $\mathfrak{M}$, and two admissible orderings π_1,π_2 such that $\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi_1,\tau;\mathfrak{M}) \neq \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi_2,\tau;\mathfrak{M})$.*

Proof. Let $V(P)=\{a,b,c\}$ with binary domains $D_a = D_b = D_c = \{0,1\}$. Impose the constraints $a=c$ and $b=c$, so that $\mathrm{Sol}(P) = \{000, 111\}$. Let the declared task be decision. A partial assignment is reachable if every constraint whose complete scope has already been processed is satisfied. For each of the two orderings below, use the following explicit finite-horizon signature. At the terminal stage set

$$\sigma_n^{\pi}(r) = \mathrm{Ans}_{\mathrm{dec},n}^{\pi}(r).$$

For $i = n-1, n-2, \dots, 0$, define recursively backward in the stage index

$$\sigma_i^{\pi}(r) = \left(\mathrm{Ans}_{\mathrm{dec},i}^{\pi}(r), \Lambda_i^{\pi}(r), \left\{\left(a, \sigma_{i+1}^{\pi}(\delta_i^{\pi}(r,a))\right) : a \in \Lambda_i^{\pi}(r)\right\}\right).$$

Because the recursion starts from the explicitly defined terminal stage n and proceeds only to smaller stage indices, this is a well-founded finite definition rather than a circular one. Taking $\mathrm{Sig}_{\mathrm{dec},i}^{\pi} = \sigma_i^{\pi}$ preserves the current decision value, the admissible next-value set, and the successor-signature class for every admissible next value; hence S1–S3 hold. Consider first the ordering $\pi_1 = (a,b,c)$. After stage 2, neither equality constraint has been completely processed, because both contain c. Therefore $\mathcal{R}_2^{\pi_1}(P) = \{00, 01, 10, 11\}$. The admissible next-value sets for c are $\Lambda_2^{\pi_1}(00) = \{0\}$, $\Lambda_2^{\pi_1}(11) = \{1\}$, and $\Lambda_2^{\pi_1}(01) = \Lambda_2^{\pi_1}(10) = \varnothing$. The singleton interface $\{a\}$ is not sufficient, because the states 00 and 01 agree on a but have different admissible next-value sets; the singleton interface $\{b\}$ is not sufficient, because the states 00 and 10 agree on b but have different admissible next-value sets. Hence $b_2^{\pi_1,\mathrm{dec}}(P) = 2$. At every other stage, an interface of size at most one is sufficient. Therefore $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi_1,\mathrm{dec};\mathfrak{M}) = 2$ and $\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi_1,\mathrm{dec};\mathfrak{M}) = 1$. Now consider the ordering $\pi_2 = (a,c,b)$. After stage 2, the constraint $a=c$ has already been processed, so the inconsistent prefixes are removed and $\mathcal{R}_2^{\pi_2}(P) = \{00, 11\}$. The admissible next-value sets for b are $\Lambda_2^{\pi_2}(00) = \{0\}$ and $\Lambda_2^{\pi_2}(11) = \{1\}$. Either component a or component c distinguishes these two reachable states, so $b_2^{\pi_2,\mathrm{dec}}(P) = 1$. At every stage, an interface of size at most one is task-sufficient. Hence $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi_2,\mathrm{dec};\mathfrak{M}) = 1$ and $\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi_2,\mathrm{dec};\mathfrak{M}) = 0$. Therefore $\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi_1,\mathrm{dec};\mathfrak{M}) \neq \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi_2,\mathrm{dec};\mathfrak{M})$. □

2.16 Structural Interpretation

CPR-Width measures the minimum maximal number of processed components whose values must remain active under the fixed representation, task, ordering, transition rules, and reconstruction-safe signature family. It therefore measures neither total candidate-space size nor the bit-level information content of retained components. Because the construction is task-relative, different declared tasks may induce different interfaces and different widths on the same finite instance. Figure 2.1 shows the complete stage-by-stage reconstruction process of this chapter's running example, (P_4,π,τ) with $\pi = (v_1,v_2,v_3,v_4)$: at every stage, the already-processed vertices are shaded, the vertex retained in the active interface $B_i^{\pi,\tau}$ is outlined, and the not-yet-processed vertices are left white. This is the dynamic, stage-by-stage counterpart to the static pipeline overview of Figure 2.2 below, which summarizes the structural chain from width to semantic-state growth to runtime cost rather than a single instance's stage-wise evolution.

Chapter 3 develops projected, reachable, and semantic interface-state spaces. Figure 2.2 summarizes the resulting width–state–runtime pipeline.

3 Raw States, Semantic States, and Reconstruction

Chapter 2 introduced reachable prefix states, task-sufficient active interfaces, reconstruction-safe future signatures, and CPR-Width. This chapter studies the state spaces induced by these objects through four

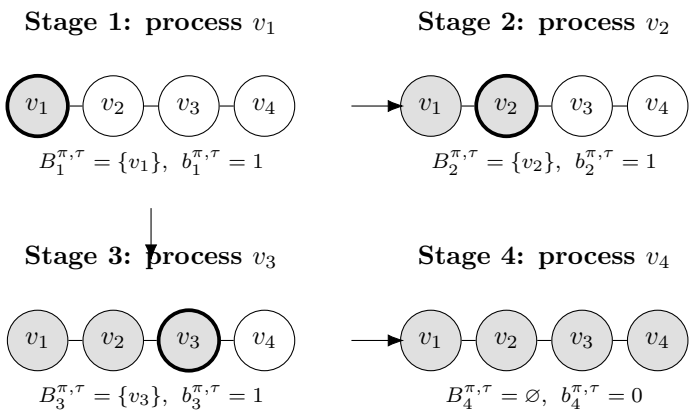


Figure 2.1: Stage-by-stage evolution of the active reconstruction interface $B_i^{\pi,\tau}$ on the running example (P_4, π, τ): outlined vertex, active interface; shaded vertex, processed but no longer active; white vertex, not yet processed. The maximum tracked size is $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P_4, \pi, \tau; \mathfrak{M}) = 1$, matching the Running Example computation in Section 2.12. Hence $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}} = 1$, and therefore the normalized CPR-Width is $\boldsymbol{\omega}_{\mathrm{rec}} = 0$.

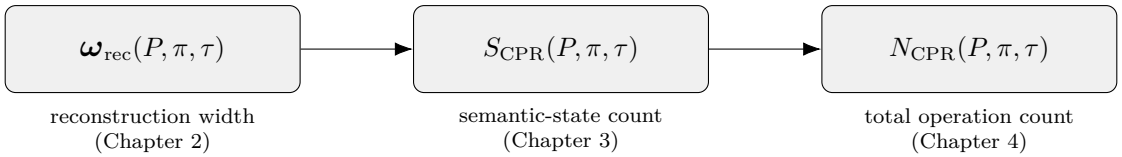


Figure 2.2: Semantic-runtime architecture of CPR-WIDTH: the structural chain connecting reconstruction width, semantic-state growth, and the complete operation-count model, as introduced in Chapter 1 and developed in full in Chapters 2–4. The reconstruction model $\mathfrak{M}$ is fixed throughout and suppressed from the notation in this figure.

distinct levels,

$$\mathcal{R}_i^{\pi}(P) \longrightarrow U_i(P, \pi, \tau; \mathfrak{M}) \longrightarrow U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M}) \longrightarrow S_i(P, \pi, \tau; \mathfrak{M}),$$

where $\mathcal{R}_i^{\pi}(P)$ is the complete reachable prefix-state space, $U_i(P, \pi, \tau; \mathfrak{M})$ is the raw assignment space of the active interface, $U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M})$ is the projection of reachable prefix states onto that interface, and $S_i(P, \pi, \tau; \mathfrak{M})$ is the semantic quotient induced by the reconstruction-safe signature. Throughout, the problem instance P, the declared task τ, the reconstruction model $\mathfrak{M}$, an admissible ordering $\pi \in \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M})$, and one cardinality-minimal task-sufficient interface $B_i^{\pi,\tau}$ at every stage are fixed; dependence on τ and $\mathfrak{M}$ may be suppressed.

3.1 Raw Interface States

At stage i, the active interface is $B_i^{\pi,\tau} \subseteq V_i^{\pi}$; every active component $v \in B_i^{\pi,\tau}$ has the finite nonempty domain D_v introduced in Chapter 2.

Definition 3.1 (Raw Interface-State Space)**.** The raw interface-state space at stage i is $U_i(P, \pi, \tau; \mathfrak{M}) = \prod_{v \in B_i^{\pi,\tau}} D_v$.

When the task and model are fixed, the shorter notation $U_i(P, \pi)$ may be used. This raw space contains all syntactically possible assignments to the selected interface, not only reachable ones. If $B_i^{\pi,\tau} = \varnothing$, the empty Cartesian product is the singleton $\{\varnothing\}$. Define

$$q(P) = \begin{cases} \max\limits_{v \in V(P)} |D_v|, & V(P) \neq \varnothing, \\ 1, & V(P) = \varnothing. \end{cases} \tag{3.1}$$

Thus $q(P) \geq 1$ is defined also for the degenerate empty-component instance, consistently with the empty-product convention above.

3.2 Raw Interface-State Bound

Theorem 3.1 (Raw Interface-State Bound)**.** *For every stage i,*

$$|U_i(P, \pi, \tau; \mathfrak{M})| \leq q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M})}, \tag{3.2}$$

$$|U_i(P, \pi, \tau; \mathfrak{M})| \leq q(P)^{\boldsymbol{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) + 1}. \tag{3.3}$$

Proof. By definition, $U_i(P, \pi, \tau; \mathfrak{M}) = \prod_{v \in B_i^{\pi,\tau}} D_v$, so $|U_i(P, \pi, \tau; \mathfrak{M})| = \prod_{v \in B_i^{\pi,\tau}} |D_v|$. Every factor satisfies $|D_v| \leq q(P)$, hence $|U_i(P, \pi, \tau; \mathfrak{M})| \leq q(P)^{|B_i^{\pi,\tau}|}$. Because the selected interface is cardinality-minimal, $|B_i^{\pi,\tau}| = b_i^{\pi,\tau}(P) \leq \widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M})$; since $q(P) \geq 1$, $q(P)^{|B_i^{\pi,\tau}|} \leq q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M})}$, which proves the first inequality. By the Interface Bound of Chapter 2, $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) \leq \boldsymbol{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) + 1$, which gives the

second. □

The bound using $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}$ is the sharper operational form; the normalized expression with $\boldsymbol{\omega}_{\mathrm{rec}} + 1$ is retained for compatibility with the normalized width convention.

3.3 Reachable Interface States

The reachable prefix-state space $\mathcal{R}_i^\pi(P) \subseteq \prod_{v \in V_i^\pi} D_v$ was defined in Chapter 2. The active-interface projection is $\rho_i^{\pi,\tau} : \mathcal{R}_i^\pi(P) \to U_i(P, \pi, \tau; \mathfrak{M})$, $\rho_i^{\pi,\tau}(r) = r|_{B_i^{\pi,\tau}}$.

Definition 3.2 (Reachable Interface-State Space)**.** The reachable interface-state space at stage i is the image $U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M}) = \rho_i^{\pi,\tau}(\mathcal{R}_i^\pi(P)) = \{r|_{B_i^{\pi,\tau}} : r \in \mathcal{R}_i^\pi(P)\}$.

Thus $U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M}) \subseteq U_i(P, \pi, \tau; \mathfrak{M})$. Raw and reachable interface states are different objects. **Example.** If a three-component Boolean interface has the raw space $U_i = \{000, 001, 010, 011, 100, 101, 110, 111\}$, the declared transition and admissibility rules may yield only $U_i^{\mathrm{reach}} = \{001, 011, 101, 111\}$: the first set contains every syntactically possible interface assignment, while the second contains only interface assignments that arise from reachable complete prefix states.

Corollary 3.1 (Reachable Interface-State Bound)**.** *For every stage i, $|U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M})| \le q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}$, and consequently $|U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M})| \le q(P)^{\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}$.*

Proof. Since $U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M}) \subseteq U_i(P, \pi, \tau; \mathfrak{M})$, $|U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M})| \le |U_i(P, \pi, \tau; \mathfrak{M})|$; the result follows from the Raw Interface-State Bound (Theorem 3.1). □

3.4 Induced Interface Signatures

Chapter 2 defined the reconstruction-safe prefix-signature map

$$\mathrm{Sig}_{\tau,i}^\pi : \mathcal{R}_i^\pi(P) \to \Sigma_{\tau,i}^\pi.$$

Because $B_i^{\pi,\tau}$ is task-sufficient, this map factors through the active-interface projection.

Proposition 3.1 (Induced Interface Signature)**.** *There exists a unique map*

$$\widehat{\mathrm{Sig}}_{\tau,i}^\pi : U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M}) \to \Sigma_{\tau,i}^\pi$$

such that $\mathrm{Sig}_{\tau,i}^\pi = \widehat{\mathrm{Sig}}_{\tau,i}^\pi \circ \rho_i^{\pi,\tau}$. Equivalently, for $u \in U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M})$ one may define $\widehat{\mathrm{Sig}}_{\tau,i}^\pi(u) = \mathrm{Sig}_{\tau,i}^\pi(r)$ for any reachable prefix state $r \in \mathcal{R}_i^\pi(P)$ satisfying $\rho_i^{\pi,\tau}(r) = u$.

Proof. Existence and uniqueness follow directly from the Interface Factorization Proposition of Chapter 2. If $\rho_i^{\pi,\tau}(r) = \rho_i^{\pi,\tau}(r') = u$, then $r|_{B_i^{\pi,\tau}} = r'|_{B_i^{\pi,\tau}}$, and since $B_i^{\pi,\tau}$ is task-sufficient, $\mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r')$; hence the value assigned to u is independent of the selected prefix-state representative. □

This construction avoids defining a future-continuation set directly for an interface assignment. In general, several reachable prefix states may project to the same interface state while possessing different literal continuation sets; what is guaranteed to be representative-independent is the induced reconstruction-safe signature.

3.5 Task-Relative Semantic Equivalence

Definition 3.3 (Reconstruction-Safe Semantic Equivalence)**.** For $u, v \in U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M})$, define $u \equiv_{P,\pi,i}^{\tau,\mathfrak{M}} v$ if and only if $\widehat{\mathrm{Sig}}_{\tau,i}^\pi(u) = \widehat{\mathrm{Sig}}_{\tau,i}^\pi(v)$.

When the problem, task, ordering, and model are fixed, the shorter notation $u \equiv_i^\tau v$ or $u \equiv_i v$ may be used. This is the primary semantic-equivalence relation of CPR-WIDTH; it is not defined merely by equality of the current decision, counting, or optimization answer, but incorporates the reconstruction-safe information required by conditions S1–S3 of Chapter 2.

Proposition 3.2 (Equivalence Relation)**.** *For every fixed P, τ, $\mathfrak{M}$, π, and stage i, the relation $\equiv_{P,\pi,i}^{\tau,\mathfrak{M}}$ is an equivalence relation on $U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M})$.*

Proof. Reflexivity and symmetry follow directly from equality of $\widehat{\mathrm{Sig}}_{\tau,i}^\pi$-values. For transitivity, if $\widehat{\mathrm{Sig}}_{\tau,i}^\pi(u) = \widehat{\mathrm{Sig}}_{\tau,i}^\pi(v)$ and $\widehat{\mathrm{Sig}}_{\tau,i}^\pi(v) = \widehat{\mathrm{Sig}}_{\tau,i}^\pi(w)$, then $\widehat{\mathrm{Sig}}_{\tau,i}^\pi(u) = \widehat{\mathrm{Sig}}_{\tau,i}^\pi(w)$. □

Remark 3.1 (Task-Answer Equality Is Not the Primary Quotient)**.** For a decision task, two prefix states may both satisfy $E_i^\pi(r) \neq \varnothing$ while admitting different next values and different successor behavior. For example, $E_i^\pi(r) = \{00\}$ and $E_i^\pi(r') = \{11\}$ produce the same current decision value $\mathbf{1}[E_i^\pi(r) \neq \varnothing] = \mathbf{1}[E_i^\pi(r') \neq \varnothing] = 1$, yet the two states may require incompatible next reconstruction choices. Therefore, equality of the current task answer is generally too coarse to define a dynamic reconstruction quotient; the reconstruction-safe signature additionally preserves admissible next values and successor classes.

3.6 Semantic Quotient

Definition 3.4 (Semantic Quotient State Space)**.** The semantic state space at stage i is the quotient $S_i(P,\pi,\tau;\mathfrak{M}) = U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})/\equiv_{P,\pi,i}^{\tau,\mathfrak{M}}$.

An element of $S_i(P,\pi,\tau;\mathfrak{M})$ is an equivalence class $[u]_i^{\tau,\mathfrak{M}} = \{v \in U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}) : v \equiv_{P,\pi,i}^{\tau,\mathfrak{M}} u\}$. The canonical quotient map $q_i^{\tau,\mathfrak{M}} : U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}) \twoheadrightarrow S_i(P,\pi,\tau;\mathfrak{M})$, $q_i^{\tau,\mathfrak{M}}(u) = [u]_i^{\tau,\mathfrak{M}}$, is surjective by construction, so $|S_i(P,\pi,\tau;\mathfrak{M})| \le |U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})|$.

Definition 3.5 (Maximum Semantic-State Count)**.** The maximum semantic-state count is

$$S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) = \max_{0\le i\le n} |S_i(P,\pi,\tau;\mathfrak{M})|. \tag{3.4}$$

When the task and model are fixed, one may write $S_i(P,\pi)$ and $S_{\mathrm{CPR}}(P,\pi)$. The symbol $S_i(P,\pi,\tau;\mathfrak{M})$ denotes a quotient set, while $S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M})$ denotes a number.

> **Running Example (continued).** For (P_4,π,τ) with $k=2$ colors, each internal interface contains one vertex, so $q(P)=2$ and
>
> $$|U_i| = |U_i^{\mathrm{reach}}| = |S_i| = 2.$$
>
> The two reachable colors induce different admissible next values, so no additional semantic collapse occurs. Hence $S_{\mathrm{CPR}}(P_4,\pi,\tau) = 2$. This valid CPR-WIDTH realization therefore exhibits reachability filtering without additional semantic compression.

3.7 Quotient Transition

To use semantic classes dynamically, stage-to-stage transitions must be well defined on quotient states. Let $u \in U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})$ and choose any reachable prefix representative $r \in \mathcal{R}_i^\pi(P)$ satisfying $\rho_i^{\pi,\tau}(r) = u$. Define the admissible next-value set of the interface state by $\widehat{\Lambda}_i^{\pi,\tau}(u) = \Lambda_i^\pi(r)$.

Lemma 3.1 (Well-Defined Interface Admissibility)**.** *The set $\widehat{\Lambda}_i^{\pi,\tau}(u)$ is independent of the selected prefix representative r.*

Proof. Let $r, r' \in \mathcal{R}_i^\pi(P)$ satisfy $\rho_i^{\pi,\tau}(r) = \rho_i^{\pi,\tau}(r') = u$. Task sufficiency implies $\mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r')$; condition S2 of Chapter 2 gives $\Lambda_i^\pi(r) = \Lambda_i^\pi(r')$. □

Definition 3.6 (Semantic Quotient Transition)**.** For a semantic class $[u]_i^{\tau,\mathfrak{M}} \in S_i(P,\pi,\tau;\mathfrak{M})$ and $a \in \widehat{\Lambda}_i^{\pi,\tau}(u)$, define

$$\overline{\delta}_i^{\pi,\tau}\left([u]_i^{\tau,\mathfrak{M}}, a\right) = \left[\rho_{i+1}^{\pi,\tau}(\delta_i^\pi(r,a))\right]_{i+1}^{\tau,\mathfrak{M}},$$

where r is any reachable prefix representative satisfying $\rho_i^{\pi,\tau}(r) = u$.

Proposition 3.3 (Well-Defined Quotient Transition)**.** *The semantic quotient transition $\overline{\delta}_i^{\pi,\tau}$ is independent of the selected interface representative u of the semantic class and of the selected prefix-state representative r of u.*

Proof. Let $[u]_i^{\tau,\mathfrak{M}} = [v]_i^{\tau,\mathfrak{M}}$, so $u \equiv_{P,\pi,i}^{\tau,\mathfrak{M}} v$ and $\widehat{\mathrm{Sig}}_{\tau,i}^\pi(u) = \widehat{\mathrm{Sig}}_{\tau,i}^\pi(v)$. Choose prefix representatives r, r' with $\rho_i^{\pi,\tau}(r) = u$ and $\rho_i^{\pi,\tau}(r') = v$; by the induced-signature definition, $\mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r')$. Condition S2 gives $\Lambda_i^\pi(r) = \Lambda_i^\pi(r')$, so the same next value a is admissible from both representatives. Condition S3 gives $\mathrm{Sig}_{\tau,i+1}^\pi(\delta_i^\pi(r,a)) = \mathrm{Sig}_{\tau,i+1}^\pi(\delta_i^\pi(r',a))$. Applying the induced interface-signature factorization at stage $i+1$ shows the two projected successors belong to the same semantic class at stage $i+1$; hence the quotient transition is well defined. □

3.8 Width–State Bound

Theorem 3.2 (Width–State Bound)**.** *For every finite problem instance P, declared task τ, reconstruction model $\mathfrak{M}$, and admissible ordering π,*

$$S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \le q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}, \tag{3.5}$$

$$S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \le q(P)^{\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}. \tag{3.6}$$

Proof. For every stage i, $|S_i(P,\pi,\tau;\mathfrak{M})| \le |U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})| \le q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}$ by the Reachable Interface-State Bound. Taking the maximum over all reconstruction stages gives $S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) = \max_{0\le i\le n}|S_i(P,\pi,\tau;\mathfrak{M})| \le q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}$. The normalized form follows from $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \le \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) + 1$. □

This theorem is a cardinality bound, not a runtime theorem: reachability and quotient construction may themselves be expensive. The unnormalized form is the sharper operational bound, while the normalized form follows from

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \le \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) + 1.$$

Therefore,

$$\boldsymbol{S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \le q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} \le q(P)^{\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}.}$$

In general, the stronger bound $S_{\mathrm{CPR}} \le q(P)^{\boldsymbol{\omega}_{\mathrm{rec}}}$ does not follow from the normalization convention.

3.9 Binary Semantic-State Dimension

Definition 3.7 (Binary Semantic-State Dimension)**.** The binary semantic-state dimension is

$$D_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) = \left\lceil \log_2 \max\left\{1, S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M})\right\} \right\rceil.$$

This is an information-theoretic fixed-length index size, not a memory or runtime cost. If $q(P) \ge 2$, the Width–State Bound implies $D_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \le \lceil \widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \log_2 q(P) \rceil \le \lceil (\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) + 1) \log_2 q(P) \rceil$. If $q(P) = 1$, every component domain is a singleton, so every raw interface-state space is a singleton, $S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) = 1$, and $D_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) = 0$.

3.10 Task Dependence

The semantic quotient is task-relative. Decision, counting, enumeration, and optimization may require different reconstruction-safe signatures even for the same representation and ordering. Therefore $\equiv^{\mathrm{dec},\mathfrak{M}}_{P,\pi,i}$, $\equiv^{\mathrm{count},\mathfrak{M}}_{P,\pi,i}$, $\equiv^{\mathrm{enum},\mathfrak{M}}_{P,\pi,i}$, and $\equiv^{\mathrm{opt},\mathfrak{M}}_{P,\pi,i}$ need not coincide, and accordingly $S_{\mathrm{CPR}}(P,\pi,\mathrm{dec};\mathfrak{M})$ may differ from $S_{\mathrm{CPR}}(P,\pi,\mathrm{count};\mathfrak{M})$, and similarly for enumeration and optimization. No general pairwise inequality is claimed; for a particular instance, two or more of these task-relative equivalence relations may coincide accidentally.

3.11 Semantic Safety

Semantic equivalence does not require preservation of every individual future continuation. The appropriate safety statement concerns the properties guaranteed by the reconstruction-safe signature.

Proposition 3.4 (Semantic Safety)**.** *Let $u, v \in U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})$ and suppose $u \equiv^{\tau,\mathfrak{M}}_{P,\pi,i} v$. Then the declared task-relative future values coincide, the admissible next-value sets coincide, for every admissible next value the corresponding successors belong to the same semantic class at stage $i+1$, and the quotient transition is independent of the selected representative.*

Proof. By semantic equivalence, $\widehat{\mathrm{Sig}}^{\pi}_{\tau,i}(u) = \widehat{\mathrm{Sig}}^{\pi}_{\tau,i}(v)$. Choose reachable prefix representatives r, r' with $\rho^{\pi,\tau}_i(r) = u$ and $\rho^{\pi,\tau}_i(r') = v$; the induced-signature definition gives $\mathrm{Sig}^{\pi}_{\tau,i}(r) = \mathrm{Sig}^{\pi}_{\tau,i}(r')$. Condition S1 of Chapter 2 gives $\mathrm{Ans}^{\pi}_{\tau,i}(r) = \mathrm{Ans}^{\pi}_{\tau,i}(r')$, proving task preservation. Condition S2 gives $\Lambda^{\pi}_i(r) = \Lambda^{\pi}_i(r')$, so the admissible next-value sets coincide. For every common admissible next value a, condition S3 gives $\mathrm{Sig}^{\pi}_{\tau,i+1}(\delta^{\pi}_i(r,a)) = \mathrm{Sig}^{\pi}_{\tau,i+1}(\delta^{\pi}_i(r',a))$, so the projected successors satisfy $\rho^{\pi,\tau}_{i+1}(\delta^{\pi}_i(r,a)) \equiv^{\tau,\mathfrak{M}}_{P,\pi,i+1} \rho^{\pi,\tau}_{i+1}(\delta^{\pi}_i(r',a))$; the quotient transition is consequently independent of the selected representative. □

Semantic safety does not generally imply $E^{\pi}_i(r) = E^{\pi}_i(r')$. For decision, counting, or optimization, different literal continuation sets may still induce the same reconstruction-safe quotient behavior. If exact equality of every continuation is required by the declared task, that stronger requirement must be included explicitly in the signature.

3.12 Structural Interpretation

The chapter establishes the stage-wise compression chain

$$\mathcal{R}^{\pi}_i(P) \xrightarrow{\rho^{\pi,\tau}_i} U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}) \xrightarrow{q^{\tau,\mathfrak{M}}_i} S_i(P,\pi,\tau;\mathfrak{M}).$$

The first map retains only the task-sufficient active interface; the second identifies reachable interface states with the same reconstruction-safe signature. Consequently,

$$|S_i| \le |U_i^{\mathrm{reach}}| \le |U_i|.$$

The resulting structural relation is

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}} \longrightarrow S_{\mathrm{CPR}}, \qquad \boldsymbol{\omega}_{\mathrm{rec}} \longrightarrow S_{\mathrm{CPR}}.$$

This chapter establishes cardinality control only; it does not claim that semantic classes are inexpensive to construct or that a small state space alone implies polynomial runtime. Complete runtime accounting is developed in Chapter 4.

4 Complete Runtime Framework

CPR-Width is a structural parameter. A small active reconstruction interface does not by itself establish a small runtime. A complete computational analysis must additionally account for the costs of constructing

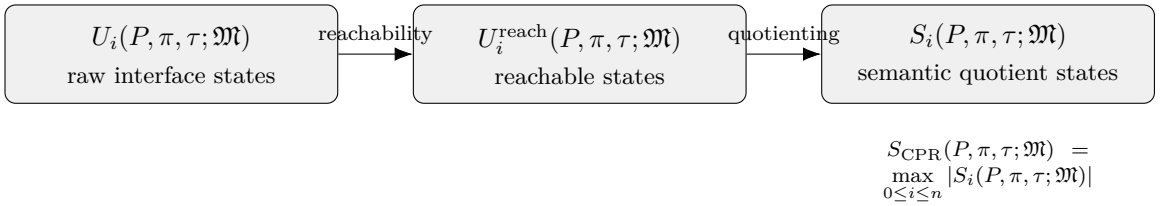


Figure 3.1: Controlled partial reconstruction and semantic-state compression: raw interface states are filtered to reachable states, then quotiented by reconstruction-safe signature equivalence to obtain semantic states, yielding the maximum semantic-state count S_{CPR}.

the declared reconstruction model, forming active interfaces and reduced representations, generating and processing semantic transitions, reconstructing task-relevant outputs, verifying correctness, and materializing the required output. This chapter introduces the complete operation-count model associated with the CPR-WIDTH framework. Throughout, let P be a finite problem instance, let τ be the declared computational task, let $\mathfrak{M}$ be the fixed CPR reconstruction model, and let $\pi \in \Pi_{\text{adm}}(P, \tau; \mathfrak{M})$ be an admissible reconstruction ordering; all runtime quantities are relative to $(P, \pi, \tau; \mathfrak{M})$, and this dependence may be suppressed when unambiguous. The runtime framework developed here is an accounting model: it specifies which elementary operations must be counted and how the complete operation count is decomposed. It is not, by itself, a hardware-specific implementation model or a measured wall-clock model. Throughout this chapter, N_{CPR} and its six components count elementary operations without yet fixing the underlying machine or bit-cost convention; the precise deterministic bit-cost computation model under which these operation counts are interpreted, and the conditions under which they are polynomially bounded in $\ell(P)$, are fixed separately in Chapter 5.

4.1 Complete Cost Model

Definition 4.1 (Complete CPR Operation Count)**.** The complete CPR operation count is

$$N_{\text{CPR}}(P, \pi, \tau; \mathfrak{M}) = N_{\text{Build}} + N_{\text{Reduce}} + N_{\text{Transition}} + N_{\text{Reconstruct}} + N_{\text{Verify}} + N_{\text{Output}}, \tag{4.1}$$

all terms evaluated at $(P, \pi, \tau; \mathfrak{M})$.

The six terms are disjoint accounting categories: every elementary operation must be assigned exactly once. In particular, semantic-class construction belongs either to N_{Reduce} or to $N_{\text{Transition}}$, output writing belongs to N_{Output}, and separately executed verification belongs to N_{Verify}. The accounting convention must be fixed before any count-level comparison. Table 4.1 provides the chapter-wide reference for these six components.

Table 4.1: The six components of N_{CPR} and where each is bounded in this chapter.

Symbol	Accounts for	Bounded in
N_{Build}	Constructing the initial explicit representation of P.	Section 4.2
N_{Reduce}	Reduction and semantic quotienting of raw interface states into semantic states.	Section 4.2
$N_{\text{Transition}}$	Stage-to-stage processing of semantic transitions along π.	Section 4.4
$N_{\text{Reconstruct}}$	Producing an admissible reconstructed object once the final stage is reached.	Section 4.6
N_{Verify}	Verifying soundness and completeness of the reconstructed object for the declared task.	Section 4.6
N_{Output}	Materializing the final output in the declared output representation.	Section 4.6

Running Example (continued). Continue with the decision task $(P_4, \pi, \tau_{\text{dec}})$ fixed in the running example of Chapters 2–3. For that declared task, the reconstruction-safe signature and the task-sufficient interfaces have already been justified, and the semantic-state count is at most 2 at every stage. Since P_4 is a fixed four-vertex instance, all six cost components are constant-size quantities on this instance. No asymptotic statement is inferred from P_4 alone; asymptotic claims require a uniformly encoded growing family. No corresponding witness-task state bound is asserted here without a separate witness-specific signature and interface proof.

4.2 Build and Reduction Cost

Definition 4.2 (Build Cost)**.** The build cost $N_{\text{Build}}(P, \pi, \tau; \mathfrak{M})$ is the number of elementary operations required to construct the initial CPR representation before semantic-state processing begins.

Depending on the declared model, this may include parsing and validating the encoded instance, constructing $V(P)$ and the local domains D_v, constructing or receiving the admissible ordering π, evaluating $\mathrm{Adm}_{P,\tau}(\pi)$, initializing reachable-prefix representations, constructing auxiliary indices or constraint tables, and allocating data structures. It does not include later semantic quotienting or stage-to-stage transitions. A favorable ordering is computationally useful only if it is either provided as part of the input or constructible within the declared cost bound.

Definition 4.3 (Reduction Cost)**.** The reduction cost $N_{\mathrm{Reduce}}(P, \pi, \tau; \mathfrak{M})$ is the number of elementary operations required to construct the task-sufficient structural reductions used by the CPR model.

This may include determining task-sufficient candidate interfaces, proving or testing interface sufficiency, selecting cardinality-minimal interfaces algorithmically, computing $b_i^{\pi,\tau}(P)$, projecting reachable prefix states onto active interfaces, constructing induced interface signatures, forming semantic equivalence classes, canonicalizing representatives, and constructing static quotient data used in later transitions. The reduction cost must include the work required to construct the semantic quotient: the existence of a small quotient does not imply that the quotient is inexpensive to compute. In particular,

$$|S_i(P, \pi, \tau; \mathfrak{M})| \ll |U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M})| \quad \not\Longrightarrow \quad N_{\mathrm{Reduce}}(P, \pi, \tau; \mathfrak{M}) \text{ is small.}$$

A semantic quotient is computationally useful only when its classes can be generated, recognized, canonicalized, or updated without enumerating the complete unresolved suffix; CPR-Width bounds the size of the quotient, not the cost of computing it. If a semantic signature is precomputed once and reused, its construction belongs to N_{Reduce}; if semantic equivalence is tested dynamically during transition processing, the corresponding tests belong to $N_{\mathrm{Transition}}$. The declared accounting rule must state which convention is used.

4.3 Reconstruction Stages and Branching

The reconstruction ordering processes the component set $V(P)$. For the component-by-component reconstruction model of Chapter 2, the number of nonterminal transition stages is $m_{\mathrm{stage}}(P, \pi) = |V(P)|$ (the symbol m_{stage} is used instead of $b(P, \pi)$ to avoid conflict with the minimum interface size $b_i^{\pi,\tau}(P)$). For $u \in U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M})$, let $\widehat{\Lambda}_i^{\pi,\tau}(u)$ denote the well-defined admissible next-value set of Chapter 3. Define the stage-wise branching number by

$$\lambda_i(P, \pi, \tau; \mathfrak{M}) = \begin{cases} \max\limits_{u \in U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})} \left|\widehat{\Lambda}_i^{\pi,\tau}(u)\right|, & U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M}) \neq \varnothing, \\ 0, & U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M}) = \varnothing, \end{cases}$$

and the maximum branching number by $\lambda_{\max}(P, \pi, \tau; \mathfrak{M}) = \max_{0 \le i < m_{\mathrm{stage}}(P,\pi)} \lambda_i(P, \pi, \tau; \mathfrak{M})$ (or 0 if $m_{\mathrm{stage}}(P, \pi) = 0$), so both quantities remain well defined in degenerate cases. If every stage is deterministic, then $\lambda_{\max}(P, \pi, \tau; \mathfrak{M}) \le 1$.

4.4 Semantic Transition Cost

A transition is an admissible move from one semantic state at stage i to one semantic state at stage $i + 1$; the quotient transition $\overline{\delta}_i^{\pi,\tau}$ was proved well defined in Chapter 3.

Definition 4.4 (Per-Edge Transition Cost)**.** Let $c_{\mathrm{edge}}(P, \pi, \tau; \mathfrak{M})$ be an upper bound on the elementary-operation cost of processing one admissible semantic transition edge: generating a successor candidate, testing local admissibility, evaluating or retrieving its signature, locating or constructing the successor class, merging duplicates, and updating transition data.

Incremental Signature Update. Evaluating or retrieving a signature, as counted in c_{edge} above, need not require recomputing the signature from the complete unresolved suffix. Suppose the reconstruction-safe signature admits a local update rule

$$\sigma_{i+1} = \Phi_i(\sigma_i, a, \eta_i),$$

where σ_i is the reconstruction-safe signature carried into stage i, a is the admissible next value being processed, and $\eta_i = \eta_i(P, i, a)$ is any additional locally needed information that is deterministically derivable from the fixed problem instance P, the stage index i, and the next value a already consulted by upd_i in Chapter 2 — for example a precomputed lookup value, a constant-size local table entry, or another quantity that is fixed once P is fixed. Consistently with the typing rule of Chapter 2 (Stage Transitions), η_i is *not* an independent, freely varying execution parameter: it may not silently replace or extend the transition argument a in S2–S3 or in upd_i, and any component of η_i that would vary independently across executions must instead be introduced through a separately defined family-level/transducer construction, exactly as required in Chapter 2.

Typing of η_i. Fix P, π, τ, and $\mathfrak{M}$ as elsewhere in this chapter. For each stage i, η_i is formally a function

$$\eta_i : D_{w_i^\pi} \longrightarrow H_i, \qquad H_i = H_i(P, \pi, \tau; \mathfrak{M}),$$

where H_i is a fixed, finite auxiliary-data set determined once P, π, τ, and $\mathfrak{M}$ are fixed, exactly as the signature codomain $\Sigma_{\tau,i}^{\pi}$ is fixed in Chapter 2; the notation $\eta_i = \eta_i(P, i, a)$ used above is a display convenience that makes the already-fixed dependence on P and i explicit and is *not* an additional free

argument varying across executions. With this typing, the local update rule is the function

$$\Phi_i : \Sigma^{\pi}_{\tau,i} \times D_{w_i^{\pi}} \times H_i \longrightarrow \Sigma^{\pi}_{\tau,i+1}.$$

CPR-WIDTH does not require semantic signatures to be recomputed by enumerating the complete continuation space. When such a rule Φ_i exists,

$$c_{\text{edge}}(P,\pi,\tau;\mathfrak{M}) \;\le\; c_{\text{succ}} + c_{\text{adm}} + c_{\text{sig}} + c_{\text{lookup}} + c_{\text{merge}},$$

where c_{succ}, c_{adm}, c_{sig}, c_{lookup}, and c_{merge} bound, respectively, generating the successor candidate, testing admissibility, applying Φ_i, locating or canonicalizing the resulting class, and merging duplicates. The existence of an incremental update rule Φ_i is model-dependent and is not guaranteed by small CPR-Width alone; where it does exist, it bounds c_{sig} without enumerating the unresolved future continuations, and where it does not, c_{sig} must be bounded separately by the declared model.

Definition 4.5 (Local State-Processing Cost)**.** Let $c_{\text{local}}(P,\pi,\tau;\mathfrak{M})$ be an upper bound on the state-local cost incurred once per semantic state during one reconstruction stage, independently of the number of outgoing transition edges: state initialization, local bookkeeping, retrieval of cached data, and state-level canonicalization.

Definition 4.6 (Per-State Transition Cost)**.** Let $c_{\text{state}}(P,\pi,\tau;\mathfrak{M})$ be an upper bound on the complete cost of processing one semantic state and all its admissible outgoing transitions during one stage.

The per-state and per-edge conventions are related by

$$c_{\text{state}}(P,\pi,\tau;\mathfrak{M}) \;\le\; \lambda_{\max}(P,\pi,\tau;\mathfrak{M})\, c_{\text{edge}}(P,\pi,\tau;\mathfrak{M}) \;+\; c_{\text{local}}(P,\pi,\tau;\mathfrak{M}).$$

The manuscript may use either convention, but the selected convention must be stated explicitly.

Definition 4.7 (Transition Cost)**.** The semantic transition cost $N_{\text{Transition}}(P,\pi,\tau;\mathfrak{M})$ is the total number of elementary operations required to process all reachable semantic transitions of the declared CPR procedure.

Using the per-state convention,

$$\begin{aligned} N_{\text{Transition}}(P,\pi,\tau;\mathfrak{M}) &\le \sum_{i=0}^{m_{\text{stage}}(P,\pi)-1} |S_i(P,\pi,\tau;\mathfrak{M})| \cdot c_{\text{state}}(P,\pi,\tau;\mathfrak{M}) \\ &\le m_{\text{stage}}(P,\pi)\, S_{\text{CPR}}(P,\pi,\tau;\mathfrak{M})\, c_{\text{state}}(P,\pi,\tau;\mathfrak{M}) \end{aligned}$$

(the sum is empty, hence 0, if $m_{\text{stage}}(P,\pi) = 0$). Using the per-edge convention, including the local cost incurred once per semantic state,

$$\begin{aligned} N_{\text{Transition}}(P,\pi,\tau;\mathfrak{M}) \le\; & m_{\text{stage}}(P,\pi)\, S_{\text{CPR}}(P,\pi,\tau;\mathfrak{M}) \\ & \times \left[\lambda_{\max}(P,\pi,\tau;\mathfrak{M})\, c_{\text{edge}}(P,\pi,\tau;\mathfrak{M}) + c_{\text{local}}(P,\pi,\tau;\mathfrak{M})\right], \end{aligned}$$

making both the branching contribution and the state-local overhead explicit. (No symbol $\tau(P,\pi)$ is used for transition cost, since τ is reserved for the declared computational task.)

4.5 Width-Based Transition Bound

By the Width–State Bound of Chapter 3, $S_{\text{CPR}}(P,\pi,\tau;\mathfrak{M}) \le q(P)^{\widehat{\boldsymbol{\omega}}_{\text{rec}}(P,\pi,\tau;\mathfrak{M})}$. Consequently, the per-state transition bound becomes

$$N_{\text{Transition}}(P,\pi,\tau;\mathfrak{M}) \le m_{\text{stage}}(P,\pi)\, q(P)^{\widehat{\boldsymbol{\omega}}_{\text{rec}}(P,\pi,\tau;\mathfrak{M})}\, c_{\text{state}}(P,\pi,\tau;\mathfrak{M}),$$

or, using the normalized width, $m_{\text{stage}}(P,\pi)\, q(P)^{\boldsymbol{\omega}_{\text{rec}}(P,\pi,\tau;\mathfrak{M})+1}\, c_{\text{state}}(P,\pi,\tau;\mathfrak{M})$. Under the per-edge convention,

$$\begin{aligned} N_{\text{Transition}}(P,\pi,\tau;\mathfrak{M}) \le\; & m_{\text{stage}}(P,\pi)\, q(P)^{\widehat{\boldsymbol{\omega}}_{\text{rec}}(P,\pi,\tau;\mathfrak{M})} \\ & \times \left[\lambda_{\max}(P,\pi,\tau;\mathfrak{M})\, c_{\text{edge}}(P,\pi,\tau;\mathfrak{M}) + c_{\text{local}}(P,\pi,\tau;\mathfrak{M})\right], \end{aligned}$$

equivalently with $q(P)^{\boldsymbol{\omega}_{\text{rec}}(P,\pi,\tau;\mathfrak{M})+1}$ using the normalized width. The bound using $\widehat{\boldsymbol{\omega}}_{\text{rec}}$ is the sharper operational form; the normalized form is retained for consistency with the principal CPR-Width notation.

> **Running Example (continued).** Instantiating the per-edge transition bound for the instance (P_4,π,τ): $m_{\text{stage}}(P_4,\pi) = 4$, $S_{\text{CPR}}(P_4,\pi,\tau) = 2$ (Section 3.6), and $\lambda_{\max}(P_4,\pi,\tau;\mathfrak{M}) = 2$. At stage 0, before any vertex has been colored, the active interface is empty, so both $k = 2$ colors are admissible for v_1, giving $\lambda_0 = 2$; at stages 1–3, fixing the color of the active interface vertex leaves exactly one admissible color available for the next vertex, giving $\lambda_1 = \lambda_2 = \lambda_3 = 1$. Hence $\lambda_{\max}(P_4,\pi,\tau;\mathfrak{M}) = \max\{2,1,1,1\} = 2$. Substituting into the per-edge Width-Based Transition Bound gives
>
> $$\begin{aligned} N_{\text{Transition}}(P_4,\pi,\tau;\mathfrak{M}) \;&\le\; 4 \cdot 2\,(2\, c_{\text{edge}}(P_4,\pi,\tau;\mathfrak{M}) + c_{\text{local}}(P_4,\pi,\tau;\mathfrak{M})) \\ &=\; 8\,(2c_{\text{edge}} + c_{\text{local}})\,. \end{aligned}$$
>
> This is not a performance claim; it is a transparent instantiation of the abstract bound, showing explicitly how $\boldsymbol{\omega}_{\text{rec}} \to S_{\text{CPR}} \to N_{\text{Transition}} \to N_{\text{CPR}}$ produces concrete numbers on this fixed four-vertex instance.

4.6 Reconstruction, Verification, and Output Cost

Definition 4.8 (Reconstruction Cost)**.** The reconstruction cost $N_{\mathrm{Reconstruct}}(P,\pi,\tau;\mathfrak{M})$ is the number of elementary operations required to recover the task-relevant output from the semantic reconstruction process.

The meaning of this cost depends on the declared task: for decision, it may require only an acceptance value or one witness; for counting, the propagation and recovery of exact multiplicities; for enumeration, the explicit recovery of every required solution; for optimization, an optimum value, an optimal witness, all optimal witnesses, or another declared certificate. It includes operations used to trace predecessor information, expand semantic representatives, recover component values, or assemble complete outputs, but not the physical writing of final outputs, which belongs to N_{Output}.

Definition 4.9 (Verification Cost)**.** The verification cost $N_{\mathrm{Verify}}(P,\pi,\tau;\mathfrak{M})$ is the number of elementary operations required to confirm that the reconstructed result satisfies the declared task conditions.

Verification may include feasibility checks, constraint checks, witness validation, objective-value validation, multiplicity consistency checks, completeness checks for enumeration, or comparison with a certified task value. It may be omitted only when correctness follows directly from the construction and this implication has been proved; merely asserting that an output is valid does not eliminate the verification term unless no separate verification procedure is actually executed.

Definition 4.10 (Output Cost)**.** The output cost $N_{\mathrm{Output}}(P,\pi,\tau;\mathfrak{M})$ is the number of elementary operations required to write, store, transmit, or otherwise materialize the result required by the task.

For a decision task the output cost may be constant; for a witness problem it is at least proportional to the witness length; for enumeration the output may be exponentially large in the input length. Let $L_{\mathrm{Output}}(P,\pi,\tau;\mathfrak{M})$ denote the total encoded output length, and assume an elementary output model in which writing one output symbol or bit requires at least a fixed positive number $c_{\mathrm{out}} > 0$ of elementary operations. Then $N_{\mathrm{Output}}(P,\pi,\tau;\mathfrak{M}) \geq c_{\mathrm{out}}\, L_{\mathrm{Output}}(P,\pi,\tau;\mathfrak{M})$; no runtime statement may ignore this unavoidable output-size lower bound. For compact notation, define the post-transition cost $R_{\mathrm{post}}(P,\pi,\tau;\mathfrak{M}) = N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}$, so that $N_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) = N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} + R_{\mathrm{post}}(P,\pi,\tau;\mathfrak{M})$ is only a notational abbreviation; the six-component decomposition of Definition 4.1 remains the primary accounting model.

4.7 Complete Width-Based Runtime Bounds

Theorem 4.1 (Complete Width-Based Runtime Bound)**.** *Under the per-state transition-cost convention,*

$$\begin{aligned} N_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \leq\ & N_{\mathrm{Build}} + N_{\mathrm{Reduce}} \\ & + m_{\mathrm{stage}}(P,\pi)\, q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}\, c_{\mathrm{state}}(P,\pi,\tau;\mathfrak{M}) + R_{\mathrm{post}}(P,\pi,\tau;\mathfrak{M}). \end{aligned} \tag{4.2}$$

and consequently, using the normalized width,

$$N_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \leq N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + m_{\mathrm{stage}}(P,\pi)\, q(P)^{\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}\, c_{\mathrm{state}}(P,\pi,\tau;\mathfrak{M}) + R_{\mathrm{post}}(P,\pi,\tau;\mathfrak{M}).$$

Proof. By the six-component decomposition and the definition of R_{post}, $N_{\mathrm{CPR}} = N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} + R_{\mathrm{post}}$. The per-state transition bound gives $N_{\mathrm{Transition}} \leq m_{\mathrm{stage}}\, S_{\mathrm{CPR}}\, c_{\mathrm{state}}$, and the Width–State Bound of Chapter 3 gives $S_{\mathrm{CPR}} \leq q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}}$; substitution proves the first bound. Chapter 3 established $q(P) \geq 1$ and $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}} \leq \boldsymbol{\omega}_{\mathrm{rec}} + 1$; since $x \mapsto q(P)^x$ is nondecreasing for $q(P) \geq 1$, it follows that $q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}} \leq q(P)^{\boldsymbol{\omega}_{\mathrm{rec}}+1}$, proving the normalized bound. □

Theorem 4.2 (Complete Edge-Based Runtime Bound)**.** *Under the per-edge transition-cost convention,*

$$\begin{aligned} N_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \leq\ & N_{\mathrm{Build}} + N_{\mathrm{Reduce}} \\ & + m_{\mathrm{stage}}(P,\pi)\, q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} \left[\lambda_{\max}(P,\pi,\tau;\mathfrak{M})\, c_{\mathrm{edge}}(P,\pi,\tau;\mathfrak{M}) \right. \\ & \qquad \left. + c_{\mathrm{local}}(P,\pi,\tau;\mathfrak{M})\right] \\ & + R_{\mathrm{post}}(P,\pi,\tau;\mathfrak{M}). \end{aligned}$$

Proof. At every stage, at most $S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M})$ semantic states are processed; each costs at most $c_{\mathrm{local}}(P,\pi,\tau;\mathfrak{M})$ for state-local work and has at most $\lambda_{\max}(P,\pi,\tau;\mathfrak{M})$ admissible outgoing transitions, each costing at most $c_{\mathrm{edge}}(P,\pi,\tau;\mathfrak{M})$. Therefore

$$\begin{aligned} N_{\mathrm{Transition}}(P,\pi,\tau;\mathfrak{M}) \leq\ & m_{\mathrm{stage}}(P,\pi)\, S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \\ & \times \left[\lambda_{\max}(P,\pi,\tau;\mathfrak{M})\, c_{\mathrm{edge}}(P,\pi,\tau;\mathfrak{M}) + c_{\mathrm{local}}(P,\pi,\tau;\mathfrak{M})\right]. \end{aligned} \tag{4.3}$$

using $S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \leq q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}$ and substituting into $N_{\mathrm{CPR}} = N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} + R_{\mathrm{post}}$ proves the bound. □

4.8 Structural Polynomial-State Criterion

Let $\ell(P)$ denote the encoding length of the problem instance. A polynomial-time conclusion requires polynomial control of every term in the complete runtime model; the criterion below controls only the width-dependent state factor and is deliberately not titled a tractability criterion on its own.

Corollary 4.1 (Constant-Domain Logarithmic-Width Criterion)**.** *Assume that there exists a constant $q_0 \geq 1$ such that $1 \leq q(P) \leq q_0$, and that $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = O(\log \ell(P))$. Then there exists a constant $c > 0$ such that $q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} \leq \ell(P)^c$ for all sufficiently large inputs; equivalently, $q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} = \ell(P)^{O(1)}$.*

Proof. If $q(P) = 1$, then $q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}} = 1$ for every input, which is trivially polynomially bounded. If $q(P) \geq 2$, since $q(P) \leq q_0$ and $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = O(\log \ell(P))$, there exists a constant $C > 0$ with $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) \leq C \log \ell(P)$ for all sufficiently large instances. Therefore $q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} \leq q_0^{C \log \ell(P)} = \ell(P)^{C \log q_0}$; since q_0 and C are constants, the exponent $C \log q_0$ is constant, so the right-hand side is polynomially bounded in $\ell(P)$. □

This corollary controls only the width-dependent state factor. A complete polynomial-time result additionally requires polynomials $p_{\mathrm{Build}}, p_{\mathrm{Reduce}}, p_{\mathrm{Transition}}, p_{\mathrm{Reconstruct}}, p_{\mathrm{Verify}}, p_{\mathrm{Output}}$ such that $N_j(P, \pi, \tau; \mathfrak{M}) \leq p_j(\ell(P))$ for every component j; in particular one must also control $m_{\mathrm{stage}}(P, \pi)$, $\lambda_{\max}(P, \pi, \tau; \mathfrak{M})$, and either $c_{\mathrm{state}}(P, \pi, \tau; \mathfrak{M})$ or $c_{\mathrm{edge}}(P, \pi, \tau; \mathfrak{M})$ together with $c_{\mathrm{local}}(P, \pi, \tau; \mathfrak{M})$, according to the selected transition-cost convention.

Theorem 4.3 (Complete Polynomial-Cost Closure)**.** *Assume there exist polynomials*

$$p_{\mathrm{Build}}, \quad p_{\mathrm{Reduce}}, \quad p_{\mathrm{Transition}},$$
$$p_{\mathrm{Reconstruct}}, \quad p_{\mathrm{Verify}}, \quad p_{\mathrm{Output}}$$

such that, for every one of the six cost components,

$$N_j(P, \pi, \tau; \mathfrak{M}) \leq p_j(\ell(P)).$$

Then there exists a polynomial p_{CPR} such that

$$N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) \leq p_{\mathrm{CPR}}(\ell(P)).$$

This is an accounting-closure statement for the six-component cost decomposition, not an independent structural tractability theorem. Its role is to make explicit that polynomial control of every declared cost component is preserved under the complete CPR cost summation.

Proof. By Equation (4.1), $N_{\mathrm{CPR}} = N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} + N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}$. Using the assumed bounds, $N_{\mathrm{CPR}} \leq p_{\mathrm{Build}}(\ell(P)) + p_{\mathrm{Reduce}}(\ell(P)) + p_{\mathrm{Transition}}(\ell(P)) + p_{\mathrm{Reconstruct}}(\ell(P)) + p_{\mathrm{Verify}}(\ell(P)) + p_{\mathrm{Output}}(\ell(P))$. A finite sum of polynomials is a polynomial, so a polynomial p_{CPR} with this property exists. □

4.9 Classical Baseline, Count-Level Gain, and Wall-Clock Gain

Let $N_{\mathrm{Classic}}(P, \tau; \mathcal{A}_{\mathrm{Classic}})$ denote the complete elementary-operation count of a declared classical baseline algorithm $\mathcal{A}_{\mathrm{Classic}}$ for the same problem representation, declared task, correctness requirement, and output requirement; the baseline must explicitly state the algorithm, the representation, all included preprocessing, the stopping condition, the verification procedure, and the required output. Assume $N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) > 0$.

Definition 4.11 (Count-Level Gain)**.** The count-level gain is an elementary-operation-count metric, not a measured execution-time metric. The count-level gain of the CPR model relative to the declared classical baseline is

$$G_{\mathrm{count}}(P, \pi, \tau; \mathfrak{M}, \mathcal{A}_{\mathrm{Classic}}) = \frac{N_{\mathrm{Classic}}(P, \tau; \mathcal{A}_{\mathrm{Classic}})}{N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M})}.$$

A favorable count-level regime is $G_{\mathrm{count}} > 1$. The comparison is meaningful only when both methods solve the same declared task under the same correctness and output requirements. Let $T_{\mathrm{Classic}}(P, \tau; \mathcal{I}_{\mathrm{Classic}})$ denote the measured wall-clock time of a declared classical implementation $\mathcal{I}_{\mathrm{Classic}}$, and let $T_{\mathrm{CPR}}(P, \pi, \tau; \mathcal{I}_{\mathrm{CPR}}) > 0$ denote the measured wall-clock time of a declared CPR implementation $\mathcal{I}_{\mathrm{CPR}}$.

Definition 4.12 (Measured Wall-Clock Gain)**.** The measured wall-clock gain is

$$G_{\mathrm{time}}(P, \pi, \tau; \mathcal{I}_{\mathrm{CPR}}, \mathcal{I}_{\mathrm{Classic}}) = \frac{T_{\mathrm{Classic}}(P, \tau; \mathcal{I}_{\mathrm{Classic}})}{T_{\mathrm{CPR}}(P, \pi, \tau; \mathcal{I}_{\mathrm{CPR}})}.$$

A count-level gain does not imply a wall-clock gain, $G_{\mathrm{count}} > 1 \not\Longrightarrow G_{\mathrm{time}} > 1$. Implementation, memory, hardware, software, and measurement effects may change wall-clock performance; operation-count and timing claims must therefore remain separate.

4.10 Structural Runtime Chain and Non-Implications

Chapters 2–4 yield the structural-accounting chain

$$\widehat{\boldsymbol{\omega}}_{\mathbf{rec}}(\boldsymbol{P}, \boldsymbol{\pi}, \boldsymbol{\tau}; \mathfrak{M}) \longrightarrow \boldsymbol{S}_{\mathbf{CPR}}(\boldsymbol{P}, \boldsymbol{\pi}, \boldsymbol{\tau}; \mathfrak{M}) \longrightarrow \boldsymbol{N}_{\mathbf{CPR}}(\boldsymbol{P}, \boldsymbol{\pi}, \boldsymbol{\tau}; \mathfrak{M}). \tag{4.4}$$

with the normalized form obtained by replacing $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}$ by $\boldsymbol{\omega}_{\mathrm{rec}}$. The first arrow is a state-space bound and the second an accounting dependency; neither is a universal tractability implication.

Proposition 4.1 (Width Does Not Determine Runtime Alone)**.** *The inequality*

$$\boldsymbol{\omega}_{\mathrm{rec}}(P, \pi_1, \tau; \mathfrak{M}) < \boldsymbol{\omega}_{\mathrm{rec}}(P, \pi_2, \tau; \mathfrak{M})$$

does not by itself imply

$$N_{\mathrm{CPR}}(P, \pi_1, \tau; \mathfrak{M}) < N_{\mathrm{CPR}}(P, \pi_2, \tau; \mathfrak{M}).$$

It also does not imply

$$T_{\mathrm{CPR}}(P, \pi_1, \tau; \mathcal{I}_{\mathrm{CPR}}) < T_{\mathrm{CPR}}(P, \pi_2, \tau; \mathcal{I}_{\mathrm{CPR}}),$$

even when both orderings are evaluated under the same declared CPR implementation and experimental conditions.

Proof. A smaller-width ordering may require more expensive ordering construction, more expensive interface-minimality tests, costlier semantic signatures, larger branching, more expensive quotient transitions, greater reconstruction or verification overhead, or less favorable memory behavior. Therefore width alone determines neither the complete operation count nor the measured wall-clock time. □

Proposition 4.2 (Semantic-State Count Does Not Determine Runtime Alone)**.** *The inequality* $S_{\mathrm{CPR}}(P, \pi_1, \tau; \mathfrak{M}) < S_{\mathrm{CPR}}(P, \pi_2, \tau; \mathfrak{M})$ *does not by itself imply* $N_{\mathrm{CPR}}(P, \pi_1, \tau; \mathfrak{M}) < N_{\mathrm{CPR}}(P, \pi_2, \tau; \mathfrak{M})$*.*

Proof. A smaller semantic quotient may be more expensive to construct, and may also require more expensive canonicalization, equivalence tests, or reconstruction steps. The complete operation count depends on all six cost components, not only on the maximum semantic-state count. □

4.11 Conclusion

CPR-Width controls active-interface structure but does not determine runtime by itself. The complete operation count remains the sum of the six disjoint components defined in Equation (4.1), and a polynomial-time conclusion requires polynomial control of every component. The central dependency is

$$\boldsymbol{\omega}_{\mathbf{rec}} \longrightarrow \boldsymbol{S}_{\mathbf{CPR}} \longrightarrow \boldsymbol{N}_{\mathbf{Transition}} \longrightarrow \boldsymbol{N}_{\mathbf{CPR}}.$$

The Width–State Bound controls the semantic-state contribution, but does not determine the costs of building, reduction, reconstruction, verification, or output.

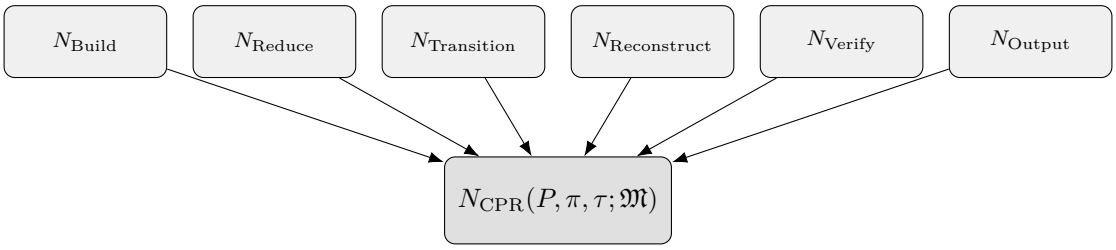


Figure 4.1: Complete runtime framework of CPR-WIDTH: the six disjoint operation-count components of Definition 4.1 sum to the total operation count N_{CPR}.

Chapter 5 formulates the boundary conditions and uniform polynomial-cost requirements required for CPR tractability.

5 Boundary Conditions for CPR Tractability

The structural and runtime bounds of the preceding chapters do not by themselves imply polynomial-time tractability. This chapter states the additional uniform conditions required for polynomial CPR execution over an encoded problem family. Throughout, all elementary-operation counts are interpreted in one fixed deterministic bit-cost computation model, which may be taken to be a deterministic multi-tape Turing machine; a deterministic RAM model may be used only when its word size, memory-access rules, and arithmetic-cost conventions are polynomially related to the input encoding length and all required bit-level costs are included in the operation count. No elementary operation may conceal the manipulation of an object whose encoded length is not itself accounted for in the complete runtime model. Let $\mathcal{F} = \{P_x : x \in \{0,1\}^*\}$ be a uniformly encoded family of finite problem instances, where x is the binary encoding of P_x, and let $\ell(P_x) = |x|$ denote the encoding length. Let τ be the declared computational task, let $\mathfrak{M}$ be a uniform CPR reconstruction model for the family, and let $\pi_x \in \Pi_{\mathrm{adm}}(P_x, \tau; \mathfrak{M})$ be the admissible reconstruction ordering used for the encoded instance P_x.

5.1 Overview

Polynomial-time CPR tractability requires four boundary conditions: BC1, explicit uniform structural representation; BC2, polynomial uniform constructibility; BC3, task-relative soundness and completeness; and BC4, polynomial semantic-state and transition-structure control. These conditions do not replace the six-component runtime analysis of Chapter 4.

Certification versus realization. A *structural realization* identifies the CPR objects for a problem family. A *tractability certification* additionally proves BC1–BC4 and polynomial bounds for all six runtime components. A small CPR-Width, a compact semantic quotient, or a successful finite example is therefore insufficient by itself.

5.2 Boundary Condition BC1

Definition 5.1 (BC1 – Explicit Uniform Structural Representation)**.** A uniformly encoded family $\mathcal{F} = \{P_x : x \in \{0,1\}^*\}$ satisfies BC1 for task τ and reconstruction model $\mathfrak{M}$ if the model explicitly specifies, for every encoded instance P_x: the finite reconstruction-component representation $V(P_x)$ and local domains D_v; the admissibility predicate for reconstruction orderings; the representation of an admissible ordering when supplied as part of the instance, or one uniform procedure for constructing one; the reachable prefix-state spaces; the task-sufficient active reconstruction interfaces; the reconstruction-safe signature family $\mathrm{Sig}^{\pi_x}_{\tau,i}$; the reachable interface-state spaces; the semantic equivalence relation and semantic quotient; the quotient transition rule; the reconstruction operator; the verification rule; and the required output representation. All objects must be specified by one uniform finite description that applies to the complete encoded family.

BC1 is a representation condition: it requires the mathematical objects underlying the CPR analysis to be explicitly declared. It does not assert that these objects are inexpensive to construct, nor that the resulting reconstruction process is sound, complete, or polynomially bounded.

5.3 Boundary Condition BC2

Definition 5.2 (BC2 – Polynomial Uniform Constructibility)**.** A uniformly encoded family $\mathcal{F}$ satisfies BC2 for task τ and model $\mathfrak{M}$ if there exists one deterministic preprocessing algorithm $\mathcal{A}_{\mathrm{Preprocess}}$ and one polynomial $p_{\mathrm{BC2}} : \mathbb{N} \to \mathbb{N}$ such that, for every encoding x,

$$N_{\mathcal{A}_{\mathrm{Preprocess}}}(x) \le p_{\mathrm{BC2}}(|x|), \tag{5.1}$$

where $N_{\mathcal{A}_{\mathrm{Preprocess}}}(x)$ is the number of elementary operations executed by the preprocessing algorithm, which constructs or reads all structural data required before dynamic semantic-state processing begins.

The constructed or supplied data include, whenever applicable, the finite component representation, the local-domain representation, an admissible reconstruction ordering, the task-sufficient active-interface representation, the static reconstruction-safe signature data, auxiliary incidence and indexing structures, and all other data assigned to N_{Build} or the static portion of N_{Reduce}. The ordering-supply procedure is part of $\mathcal{A}_{\mathrm{Preprocess}}$: it either reads an admissible ordering from the declared input or constructs one uniformly from x; in both cases $\pi_x = \mathcal{A}_\pi(x) \in \Pi_{\mathrm{adm}}(P_x, \tau; \mathfrak{M})$, and the operation count of $\mathcal{A}_\pi$ is included in N_{Build}, not counted as an additional runtime component. The quantity $N_{\mathcal{A}_{\mathrm{Preprocess}}}$ is an auxiliary preprocessing count and does not introduce a seventh component into the complete runtime model; every elementary preprocessing operation is assigned either to N_{Build} or to N_{Reduce}, following the disjoint accounting convention of Chapter 4, but not to both. BC2 is uniform: the same preprocessing algorithm and polynomial bound must apply to every encoded instance. It does not control semantic transitions, reconstruction, verification, or output, whose costs remain separate runtime components. BC2 also forbids hiding an exponentially large continuation-space enumeration inside an elementary preprocessing step. Any such enumeration must be fully charged to the appropriate runtime component and is admissible only when its total bit-cost is polynomial in $|x|$. Direct or incremental alternatives are permitted, but their construction cost must be proved for the declared family.

5.4 Boundary Condition BC3

Definition 5.3 (BC3 – Task-Relative Soundness and Completeness)**.** A CPR reconstruction model satisfies BC3 for a declared task τ if its reconstructed result is both sound and complete with respect to the exact output requirement of τ.

The precise condition depends on the declared task. For every task τ, let $\mathrm{Sol}_\tau(P_x)$ denote the complete set of feasible objects relevant to that task, whenever such a representation is applicable. For **decision**, let $\mathrm{Dec}_\tau(P_x) \in \{0,1\}$ denote the correct task-relative decision value and $\mathrm{CPRDecision}(P_x, \pi_x, \tau; \mathfrak{M}) \in \{0,1\}$ the CPR-computed value; soundness and completeness require $\mathrm{CPRDecision}(P_x, \pi_x, \tau; \mathfrak{M}) = \mathrm{Dec}_\tau(P_x)$, which for an existential feasibility decision task specializes to $\mathrm{Sol}_\tau(P_x) \neq \varnothing \Leftrightarrow \mathrm{CPRDecision}(P_x, \pi_x, \tau; \mathfrak{M}) = 1$. The CPR procedure must return the correct yes-or-no answer; it need not preserve every individual feasible witness unless witness reconstruction is part of the declared task. For **enumeration**, soundness and completeness require $\mathrm{Output}_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) = \mathrm{Sol}_\tau(P_x)$, equivalently soundness $\mathrm{Output}_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) \subseteq \mathrm{Sol}_\tau(P_x)$ and completeness $\mathrm{Sol}_\tau(P_x) \subseteq \mathrm{Output}_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M})$. For **counting**, soundness and completeness require $\mathrm{CPRCount}(P_x, \pi_x, \tau; \mathfrak{M}) = |\,\mathrm{Sol}_\tau(P_x)|$. For **optimization**, let $\mathrm{Opt}_\tau(P_x)$ denote the exact task-required optimum object, which may be the optimum value, one optimal witness, all optimal witnesses, or another explicitly declared optimality certificate. The codomain of Opt_τ is part of the declaration of the task τ. BC3 requires $\mathrm{CPROpt}(P_x, \pi_x, \tau; \mathfrak{M}) = \mathrm{Opt}_\tau(P_x)$. No weaker output object may silently replace the declared task requirement. More generally, if the task is represented by the evaluation map Θ_τ, BC3 requires exact preservation of the declared task value,

$$\Theta^{\mathrm{CPR}}_\tau(P_x, \pi_x; \mathfrak{M}) = \Theta_\tau(P_x). \tag{5.2}$$

The decision, enumeration, counting, and optimization conditions above are special cases of this identity. BC3 is a correctness condition and does not by itself provide a polynomial runtime bound.

5.5 Boundary Condition BC4

For each encoded instance, define the total number of semantic-state occurrences by $V_{\mathrm{sem}}(P_x, \pi_x, \tau; \mathfrak{M}) = \sum_{i=0}^{m_{\mathrm{stage}}(P_x,\pi_x)} |S_i(P_x, \pi_x, \tau; \mathfrak{M})|$; the same abstract state occurring at two different stages is counted twice, since the two occurrences belong to different stage-specific semantic-state spaces. Let $E_{\mathrm{sem}}(P_x, \pi_x, \tau; \mathfrak{M})$ denote the total number of reachable semantic transition edges processed by the declared CPR reconstruction system.

Definition 5.4 (BC4 – Polynomial Semantic-State and Transition-Structure Control)**.** A uniformly encoded family $\mathcal{F}$ satisfies BC4 for task τ, model $\mathfrak{M}$, and admissible orderings π_x if there exist polynomials $p_{\mathrm{SemanticVertex}}, p_{\mathrm{SemanticEdge}} : \mathbb{N} \to \mathbb{N}$, independent of x, such that for every encoded instance P_x,

$$V_{\mathrm{sem}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_{\mathrm{SemanticVertex}}(|x|), \tag{5.3}$$

$$E_{\mathrm{sem}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_{\mathrm{SemanticEdge}}(|x|). \tag{5.4}$$

The first inequality controls the complete number of stage-indexed reachable semantic states, and the second controls the complete number of reachable semantic transition edges. Since $S_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) = \max_i |S_i(P_x, \pi_x, \tau; \mathfrak{M})| \leq V_{\mathrm{sem}}(P_x, \pi_x, \tau; \mathfrak{M})$, BC4 also implies polynomial control of the maximum stage-wise semantic-state count. A short binary description of one semantic state does not imply that only polynomially many semantic states occur, and polynomially many semantic states do not imply polynomially many semantic transition edges when the branching structure is not polynomially controlled. BC4 is a structural cardinality condition; it does not by itself imply that processing one transition edge is inexpensive. The complete elementary-operation cost of generating, identifying, merging, storing, and processing the semantic transitions remains $N_{\mathrm{Transition}}(P_x, \pi_x, \tau; \mathfrak{M})$, which must be controlled separately by the complete runtime model, and BC4 also does not eliminate the need to control reconstruction, verification, and output costs separately. BC4 is retained as an independent structural certification condition, although the complete polynomial-time conclusion is ultimately obtained from the six component-wise operation-count bounds. BC4 also does not identify semantic-state cardinality with memory consumption. A family may contain polynomially many semantic states while individual signatures, transition records, or reconstructed objects require large encodings. Therefore, any polynomial-resource claim must additionally account for the encoded lengths of all stored objects in the fixed bit-cost model. In particular, the representation size of reconstruction-safe signatures may not be treated as constant unless such a bound follows from the declared reconstruction model. Thus, polynomial semantic-state cardinality is a necessary structural control condition within the CPR certificate, but it is not a substitute for explicit construction-cost, transition-cost, and representation-size analysis.

5.6 CPR-Tractable Families and the Language Class

The expression "CPR-tractable" is used only when the four boundary conditions and the complete polynomial-cost requirements are satisfied uniformly.

Definition 5.5 (CPR-Tractable Decision Family)**.** Let $\mathcal{F} = \{P_x : x \in \{0,1\}^*\}$ be a uniformly encoded family of finite decision-problem instances. The family is CPR-tractable if there exist one declared decision task τ, one uniform CPR reconstruction model $\mathfrak{M}$, one uniform deterministic preprocessing algorithm $\mathcal{A}_{\mathrm{Preprocess}}$, containing an ordering-supply component $\mathcal{A}_\pi$ with $\pi_x = \mathcal{A}_\pi(x) \in \Pi_{\mathrm{adm}}(P_x, \tau; \mathfrak{M})$, and six fixed polynomial functions $p_j : \mathbb{N} \to \mathbb{N}$, $j \in J = \{\mathrm{Build}, \mathrm{Reduce}, \mathrm{Transition}, \mathrm{Reconstruct}, \mathrm{Verify}, \mathrm{Output}\}$, such that BC1–BC4 hold uniformly and, for every encoding x, $N_j(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_j(|x|)$ for every $j \in J$.

The notation $\mathcal{A}_\pi(x)$ also covers the case in which the ordering is explicitly included in the declared input representation and is read and validated by the ordering-supply component. Let the decision language associated with the family be $L_{\mathcal{F}} = \{x \in \{0,1\}^* : P_x \text{ is a yes-instance for the declared task } \tau\}$.

Definition 5.6 (The Language Class $\mathbf{P}_{\mathrm{CPR}}$)**.** Define

$$\mathbf{P}_{\mathrm{CPR}} = \{L \subseteq \{0,1\}^* : L = L_{\mathcal{F}} \text{ for some uniformly encoded CPR-tractable decision family } \mathcal{F}\}.$$

The notation $\mathbf{P}_{\mathrm{CPR}}$ is introduced specifically for the present framework and does not denote a standard complexity class from the literature. Membership in $\mathbf{P}_{\mathrm{CPR}}$ therefore requires more than satisfaction of BC1–BC4 as isolated statements: it requires a uniform CPR decision procedure with polynomial bounds for all six components of the complete operation-count model. The class is framework-dependent because certification depends on the declared component representation, the task, the reconstruction model, the admissible-ordering rule, the reconstruction-safe semantic signatures, and the complete runtime-accounting convention.

5.7 Uniform Tractability

Theorem 5.1 (Uniform CPR Tractability)**.** *Let $\mathcal{F} = \{P_x : x \in \{0,1\}^*\}$ be a uniformly encoded family of finite decision-problem instances with associated decision language $L_{\mathcal{F}}$. Assume there exist one declared decision task τ, one uniform CPR reconstruction model $\mathfrak{M}$, one uniform deterministic preprocessing algorithm $\mathcal{A}_{\mathrm{Preprocess}}$ with ordering-supply component $\mathcal{A}_\pi$ producing $\pi_x = \mathcal{A}_\pi(x) \in \Pi_{\mathrm{adm}}(P_x, \tau; \mathfrak{M})$, and six*

fixed polynomial functions $p_j : \mathbb{N} \to \mathbb{N}$, $j \in J$, such that BC1, BC2, BC3, and BC4 hold uniformly and, for every encoding x, $N_j(P_x, \pi_x, \tau; \mathfrak{M}) \le p_j(|x|)$ for every $j \in J$. Then $L_{\mathcal{F}} \in \mathbf{P}$.

Proof. By BC1, the complete CPR decision model is explicitly and uniformly specified. By BC2, the reconstruction representation, all required structural data, and an admissible ordering are read or constructed using a number of elementary operations polynomial in $|x|$. By BC3, the CPR decision procedure is sound and complete: $x \in L_{\mathcal{F}} \Leftrightarrow \mathrm{CPRDecision}(P_x, \pi_x, \tau; \mathfrak{M}) = 1$. By BC4, the complete stage-indexed semantic-state system and its semantic transition graph have polynomially bounded cardinality; the complete cost of processing that graph is controlled separately by the assumed polynomial bound on $N_{\mathrm{Transition}}$. By the complete operation-count decomposition of Chapter 4, $N_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) = N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} + N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}$; using the six assumed uniform polynomial bounds gives $N_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) \le p_{\mathrm{Build}}(|x|) + p_{\mathrm{Reduce}}(|x|) + p_{\mathrm{Transition}}(|x|) + p_{\mathrm{Reconstruct}}(|x|) + p_{\mathrm{Verify}}(|x|) + p_{\mathrm{Output}}(|x|)$, a finite sum of fixed polynomials and hence itself polynomially bounded in $|x|$. Because the operation counts are interpreted in the fixed deterministic bit-cost model declared at the start of this chapter, the complete CPR procedure is executable in deterministic polynomial time, so membership of x in $L_{\mathcal{F}}$ is decided by one deterministic polynomial-time procedure. Hence $L_{\mathcal{F}} \in \mathbf{P}$. □

Theorem 5.1 is a certification theorem, not a width-only tractability theorem. BC1–BC4 certify the declared model structure, constructibility, correctness, and semantic-state control; deterministic polynomial time follows only after all six runtime components are uniformly polynomially bounded. Accordingly,

$$\text{small CPR-Width} \not\Longrightarrow \text{polynomial construction cost},$$

$$\text{polynomially many semantic states} \not\Longrightarrow \text{polynomial transition cost},$$

and

$$\text{polynomial internal reconstruction} \not\Longrightarrow \text{polynomial total runtime}$$

when verification or required output is superpolynomial.

Corollary 5.1 (Certified CPR Languages Are Polynomial-Time Languages)**.** $\mathbf{P}_{\mathrm{CPR}} \subseteq \mathbf{P}$.

Role of the class $\mathbf{P}_{\mathrm{CPR}}$. *The purpose of $\mathbf{P}_{\mathrm{CPR}}$ is not to define a complexity class larger than, smaller than, or independent of $\mathbf{P}$ by means of a new machine model. Its role is certificational. Membership records that deterministic polynomial-time solvability has been established through a specific CPR decomposition: a uniformly constructible reconstruction model, task-sufficient semantic compression, polynomially bounded semantic state and transition structure, and complete polynomial accounting of all reconstruction, verification, and output costs. Thus the content of a CPR certificate lies in the structural explanation of a polynomial-time computation, rather than in the set-theoretic inclusion $\mathbf{P}_{\mathrm{CPR}} \subseteq \mathbf{P}$ itself.*

Proof. Let $L \in \mathbf{P}_{\mathrm{CPR}}$. By Definition 5.6, L is associated with a uniformly encoded CPR-tractable decision family, so the hypotheses of Theorem 5.1 hold, and hence $L \in \mathbf{P}$. □

This inclusion is a consequence of the certification requirements built into the definition of $\mathbf{P}_{\mathrm{CPR}}$; it is not a complexity-class collapse statement.

5.8 Conditional Complexity Consequence

Corollary 5.2 (Conditional NP-Complete Consequence)**.** *Let $L^* \subseteq \{0,1\}^*$ be an NP-complete language. If $L^* \in \mathbf{P}_{\mathrm{CPR}}$, then $\mathbf{P} = \mathbf{NP}$.*

Proof. By Corollary 5.1, $\mathbf{P}_{\mathrm{CPR}} \subseteq \mathbf{P}$; therefore $L^* \in \mathbf{P}_{\mathrm{CPR}}$ implies $L^* \in \mathbf{P}$. Since L^* is NP-complete, every language in $\mathbf{NP}$ is polynomial-time many-one reducible to L^*, and because $L^* \in \mathbf{P}$, it follows that $\mathbf{NP} \subseteq \mathbf{P}$. The standard inclusion $\mathbf{P} \subseteq \mathbf{NP}$ also holds. Consequently $\mathbf{P} = \mathbf{NP}$. □

Remark 5.1 (Unproved Premise)**.** The manuscript does not establish the hypothesis $L^* \in \mathbf{P}_{\mathrm{CPR}}$ for any NP-complete language L^*; in particular, no NP-complete language is proved in this work to possess a uniform CPR reconstruction model satisfying BC1–BC4 together with uniform polynomial bounds for every component of the complete runtime model. The preceding corollary is therefore a conditional statement only and does not establish $\mathbf{P} = \mathbf{NP}$.

5.9 Conclusion

BC1–BC4 certify representation, constructibility, correctness, and semantic-state control. Polynomial-time CPR tractability additionally requires uniform polynomial bounds for all six runtime components. The complete sufficient condition is

$$\Big[\exists\tau\,\exists\mathfrak{M}\,\exists\mathcal{A}_{\mathrm{Preprocess}}\,\exists\{p_j\}_{j\in J}\,\forall x\in\{0,1\}^*:$$
$$\pi_x=\mathcal{A}_\pi(x)\in\Pi_{\mathrm{adm}}(P_x,\tau;\mathfrak{M})\wedge\mathrm{BC1}\wedge\mathrm{BC2}\wedge\mathrm{BC3}\wedge\mathrm{BC4}$$
$$\wedge\bigwedge_{j\in J}\left[N_j(P_x,\pi_x,\tau;\mathfrak{M})\le p_j(|x|)\right]\Big]\Longrightarrow L_{\mathcal{F}}\in\mathbf{P},$$

The quantifier order is essential: the task, reconstruction model, preprocessing algorithm, ordering-supply rule, and six polynomial bounds are fixed for the complete encoded family before an instance x is selected. Hence

$$\mathbf{P}_{\mathrm{CPR}}\subseteq\mathbf{P},$$

while the NP-complete consequence remains strictly conditional:

$$L^*\text{ NP-complete}\wedge L^*\in\mathbf{P}_{\mathrm{CPR}}\Longrightarrow\mathbf{P}=\mathbf{NP}.$$

No such NP-complete membership premise is established in this manuscript.

Family-level scaling requirement. Finite verification at a single parameter value is insufficient for BC2 or BC4. A family-level certification must bound semantic representation construction, semantic-state growth, and transition processing as functions of the input encoding length. In particular, constant or bounded CPR-Width does not by itself imply polynomial construction of the semantic quotient. The following chapters provide representative structural realizations; they do not certify the corresponding unrestricted problem classes as CPR-tractable.

6 Problem-Class Realizations of CPR-Width

The mathematical theory developed in the preceding chapters is intended to apply to declared reconstruction models rather than to one isolated problem domain. This chapter studies representative realizations of Controlled Partial Reconstruction Width for Boolean satisfiability, finite constraint satisfaction, graph coloring, tensor representations, and digital arithmetic. Throughout, let P denote a finite problem instance, let τ be a fixed declared computational task, let $\mathfrak{M}$ be a fixed reconstruction model, and let $\pi\in\Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})$ be an admissible reconstruction ordering; the dependence on τ and $\mathfrak{M}$ may be suppressed when unambiguous.

A central distinction is required throughout. For every problem-class realization, the visibly defined structural boundary is first treated as a *candidate interface* $\widetilde{B}_i^{\pi,\tau}$, to be distinguished from the cardinality-minimal task-sufficient interface $B_i^{\pi,\tau}$ of Chapter 2. Unless task sufficiency and cardinality minimality have both been proved, one may conclude only the general candidate bound

$$b_i^{\pi,\tau}(P)\le\big|\widetilde{B}_i^{\pi,\tau}\big|,\qquad\text{hence}\qquad\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})\le\max_i\big|\widetilde{B}_i^{\pi,\tau}\big|,\tag{6.1}$$

with equality requiring an additional cardinality-minimality proof. Each section below instantiates $\widetilde{B}_i^{\pi,\tau}$ for a specific representation and applies Bound (6.1) rather than re-deriving it.

6.1 Boolean Satisfiability

Let $\Phi=C_1\wedge C_2\wedge\cdots\wedge C_m$ be a Boolean formula over $V(\Phi)=\{x_1,\dots,x_n\}$, let $\pi=(x_{\pi(1)},\dots,x_{\pi(n)})$ be an admissible variable ordering, with processed prefix $V_i^\pi=\{x_{\pi(1)},\dots,x_{\pi(i)}\}$ and unresolved suffix $\overline{V}_i^\pi=V(\Phi)\setminus V_i^\pi$. Define the structural SAT candidate interface by

$$\widetilde{B}_i^{\Phi,\pi}=\big\{x\in V_i^\pi:x\text{ occurs in a clause whose residual status is not yet fixed}\big\}.\tag{6.2}$$

This is the set of processed variables whose assigned values may still influence residual clauses; it is a natural structural candidate, not automatically the unique or cardinality-minimal task-sufficient interface. For a reachable prefix assignment $r\in\mathcal{R}_i^\pi(\Phi)$, the exact satisfying-extension set is $E_i^{\Phi,\pi}(r)=\big\{s\in\prod_{x\in\overline{V}_i^\pi}\{0,1\}:r\cup s\models\Phi\big\}$. This extension set is a strong task-relevant residual object, but under the general reachability convention of Chapter 2 it is not automatically a reconstruction-safe signature: reachability does not imply extendability, and two reachable prefixes may share $E_i^{\Phi,\pi}(r)=E_i^{\Phi,\pi}(r')=\varnothing$ while still admitting different locally admissible next values.

Accordingly, let $\mathrm{Sig}_{\mathrm{SAT},i}^\pi(r)$ denote a declared SAT prefix signature satisfying conditions S1–S3 of Chapter 2; such a signature may contain $E_i^{\Phi,\pi}(r)$ together with exactly the additional transition information required for reconstruction safety. Alternatively, the extension-set signature $\mathrm{Sig}_{\mathrm{SAT},i}^\pi(r)=E_i^{\Phi,\pi}(r)$ may be used only for a declared SAT reachability rule for which S1–S3 have been proved separately. If the candidate interface $\widetilde{B}_i^{\Phi,\pi}$ determines the selected reconstruction-safe signature, then it is task-sufficient and Bound (6.1) gives

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(\Phi,\pi,\mathrm{SAT};\mathfrak{M})\le\max_i\big|\widetilde{B}_i^{\Phi,\pi}\big|,\qquad\boldsymbol{\omega}_{\mathrm{rec}}(\Phi,\pi,\mathrm{SAT};\mathfrak{M})=\max\{0,\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(\Phi,\pi,\mathrm{SAT};\mathfrak{M})-1\}.\tag{6.3}$$

An equality claim in (6.3) requires a proof that no smaller processed-variable set determines the same reconstruction-safe signature. This realization does not claim that unrestricted SAT has bounded CPR-Width or admits polynomial CPR reconstruction.

6.2 Constraint Satisfaction Problems

Let $P_{\mathrm{CSP}} = (V, D, \mathcal{C})$ be a finite CSP instance with $V = \{x_1, \dots, x_n\}$, domain family $D = (D_{x_1}, \dots, D_{x_n})$, and constraint set $\mathcal{C} = \{C_1, \dots, C_m\}$, where each constraint C has variable scope $\mathrm{scope}(C) \subseteq V$. For an admissible ordering π, define the structural CSP candidate interface by

$$\widetilde{B}_i^{P_{\mathrm{CSP}},\pi} = \{x \in V_i^\pi : \exists C \in \mathcal{C},\ x \in \mathrm{scope}(C),\ \mathrm{scope}(C) \cap \overline{V}_i^\pi \neq \varnothing\}, \tag{6.4}$$

i.e. the processed variables participating in constraints whose scopes still contain unresolved variables. For a reachable prefix assignment r, the exact globally satisfying extension set is $E_i^{\mathrm{CSP},\pi}(r) = \{s \in \prod_{x \in \overline{V}_i^\pi} D_x : r \cup s \text{ satisfies every constraint in } \mathcal{C}\}$. As in the SAT realization, this is a strong residual object but does not automatically establish S2 under a reachability rule that permits prefixes with no complete satisfying extension; a reconstruction-safe CSP signature must either augment $E_i^{\mathrm{CSP},\pi}(r)$ with the transition information required by S1–S3, or the declared reachability rule must be accompanied by a separate proof that the extension-set signature itself satisfies S1–S3. For counting, optimization, or another declared task, the signature must analogously preserve the corresponding task value together with all transition information required by reconstruction safety.

If $\widetilde{B}_i^{P_{\mathrm{CSP}},\pi}$ determines the declared safe signature, Bound (6.1) gives $b_i^{\pi,\tau}(P_{\mathrm{CSP}}) \le |\widetilde{B}_i^{P_{\mathrm{CSP}},\pi}|$, hence

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P_{\mathrm{CSP}}, \pi, \tau; \mathfrak{M}) \le \max_i |\widetilde{B}_i^{P_{\mathrm{CSP}},\pi}|, \qquad \boldsymbol{\omega}_{\mathrm{rec}}(P_{\mathrm{CSP}}, \pi, \tau; \mathfrak{M}) = \max\{0, \widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P_{\mathrm{CSP}}, \pi, \tau; \mathfrak{M}) - 1\}. \tag{6.5}$$

The structural boundary is therefore a certified upper bound once task sufficiency has been established; it becomes the exact CPR interface size only after a cardinality-minimality proof.

6.3 Graph Coloring

Let $G = (V, E)$ be a finite graph and $K = \{1, \dots, k\}$ the available color set. For an ordering $\pi = (v_{\pi(1)}, \dots, v_{\pi(|V|)})$, define the structural graph-coloring candidate interface by

$$\widetilde{B}_i^{G,\pi} = \{v \in V_i^\pi : \exists w \in \overline{V}_i^\pi,\ \{v, w\} \in E\}, \qquad \mathrm{vs}(G, \pi) = \max_i |\widetilde{B}_i^{G,\pi}|, \tag{6.6}$$

which is exactly the classical vertex-separation boundary associated with π [5]. The equality $b_i^{\pi,\tau}(G) = |\widetilde{B}_i^{G,\pi}|$ does not follow solely from this structural observation, since semantic redundancy may permit a smaller task-sufficient coordinate set; the generally valid relation is $b_i^{\pi,\tau}(G) \le |\widetilde{B}_i^{G,\pi}|$ whenever the structural boundary determines the selected reconstruction-safe signature. Consequently, by Bound (6.1),

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(G, \pi, \tau; \mathfrak{M}) \le \mathrm{vs}(G, \pi), \qquad \boldsymbol{\omega}_{\mathrm{rec}}(G, \pi, \tau; \mathfrak{M}) \le \max\{0, \mathrm{vs}(G, \pi) - 1\}. \tag{6.7}$$

The right-hand side of (6.7) is the ordering-specific vertex-separation value; the left-hand side is the minimum task-sufficient operational CPR interface width. They may coincide for a particular realization, but equality requires a proof. The purpose of this realization is therefore not to rename vertex separation, but to embed a classical structural boundary into a task-relative reconstruction model with reconstruction-safe signatures, semantic quotienting, and complete runtime accounting.

Preliminary width-comparison hypothesis. Whenever the structural candidate boundary is proved task-sufficient for the selected reconstruction-safe signature, Bound (6.7) motivates the preliminary hypothesis that, for suitable graph tasks and representations, CPR-Width may be upper-controlled by a classical separator-based width parameter. The statement is deliberately one-sided: the minimum task-sufficient CPR interface may be strictly smaller than the structural separator. A formal theorem relating minimum CPR-Width to pathwidth, treewidth, or another classical width parameter requires a fixed representation, task, signature family, admissibility convention, and normalization rule, and is developed further as an open problem in Chapter 9.

6.3.1 Quantitative Verification for C_4

Consider the four-vertex cycle $C_4 = (V, E)$, where $V = \{v_1, v_2, v_3, v_4\}$ and

$$E = \{\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}, \{v_4, v_1\}\}.$$

Three colors are available. The complete raw assignment count is $N_{\mathrm{raw}} = 3^4 = 81$. Let $\Phi(C_4) = \{c : V \to \{1, 2, 3\} : c(u) \neq c(v) \text{ for every } \{u, v\} \in E\}$ be the set of proper three-colorings, so the number of admissible colorings is $N_{\mathrm{adm}} = |\Phi(C_4)| = 18$. Fixing the partial coloring $f(v_1) = 1,\ f(v_2) = 2$, the admissible completions are $(c(v_3), c(v_4)) \in \{(1, 2), (1, 3), (3, 2)\}$, so $N_{\mathrm{rec}}(f) = 3$. The three quantities describe different spaces,

$$N_{\mathrm{raw}} = 81 \neq N_{\mathrm{adm}} = 18 \neq N_{\mathrm{rec}}(f) = 3, \tag{6.8}$$

namely the cardinality of the complete raw assignment space, of the admissible proper-coloring space, and of the reconstruction fiber induced by the fixed partial coloring f, respectively. These are finite

cardinalities; they are not CPR-Width values and must not be identified with $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}$, $\boldsymbol{\omega}_{\mathrm{rec}}$, S_{CPR}, or a runtime quantity. The admissible-to-fiber and raw-to-fiber cardinality ratios, $K_{\mathrm{adm/fiber}} = 18/3 = 6$ and $K_{\mathrm{raw/fiber}} = 81/3 = 27$, compare the conditional reconstruction-fiber cardinality obtained after fixing f with unconditional finite cardinalities; they are descriptive only and must not be interpreted as algorithmic speedup factors, compression factors, CPR-Width values, semantic-state counts, or runtime gains.

6.3.2 Stage-by-Stage Interface Tracking and Ordering Dependence for C_5

The C_4 example compares raw, admissible, and fiber cardinalities but does not track the structural candidate interface stage by stage. The five-vertex cycle $C_5 = (V, E)$ with $V = \{v_1, \ldots, v_5\}$ and $E = \{\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}, \{v_4, v_5\}, \{v_5, v_1\}\}$ provides a second finite structural example.

Ordering A: consecutive traversal. For $\pi_A = (v_1, v_2, v_3, v_4, v_5)$, tracking the structural candidate boundary gives $\widetilde{B}_1^{G,\pi_A} = \{v_1\}$, $\widetilde{B}_2^{G,\pi_A} = \{v_1, v_2\}$, $\widetilde{B}_3^{G,\pi_A} = \{v_1, v_3\}$, $\widetilde{B}_4^{G,\pi_A} = \{v_1, v_4\}$, $\widetilde{B}_5^{G,\pi_A} = \varnothing$, so $\max_i |\widetilde{B}_i^{G,\pi_A}| = 2$. If these candidate boundaries are task-sufficient for the selected reconstruction-safe signature, then $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(C_5, \pi_A, \tau; \mathfrak{M}) \leq 2$ and $\boldsymbol{\omega}_{\mathrm{rec}}(C_5, \pi_A, \tau; \mathfrak{M}) \leq 1$.

Ordering B: interleaved traversal. For $\pi_B = (v_1, v_3, v_5, v_2, v_4)$, tracking the same boundary gives $\widetilde{B}_1^{G,\pi_B} = \{v_1\}$, $\widetilde{B}_2^{G,\pi_B} = \{v_1, v_3\}$, $\widetilde{B}_3^{G,\pi_B} = \{v_1, v_3, v_5\}$, $\widetilde{B}_4^{G,\pi_B} = \{v_3, v_5\}$, $\widetilde{B}_5^{G,\pi_B} = \varnothing$, so $\max_i |\widetilde{B}_i^{G,\pi_B}| = 3$. If task-sufficient, then $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(C_5, \pi_B, \tau; \mathfrak{M}) \leq 3$ and $\boldsymbol{\omega}_{\mathrm{rec}}(C_5, \pi_B, \tau; \mathfrak{M}) \leq 2$.

Reading the comparison. The same graph, task, and reconstruction model exhibit different tracked candidate-boundary maxima under the two orderings, 2 versus 3, demonstrating ordering dependence of the structural candidate boundary. This does not by itself prove $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(C_5, \pi_A, \tau; \mathfrak{M}) < \widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(C_5, \pi_B, \tau; \mathfrak{M})$, because cardinality minimality of the candidate boundary has not been proved at every stage, and no exact separation of the corresponding normalized values $\boldsymbol{\omega}_{\mathrm{rec}}(C_5, \pi_A, \tau; \mathfrak{M})$ and $\boldsymbol{\omega}_{\mathrm{rec}}(C_5, \pi_B, \tau; \mathfrak{M})$ is claimed here. The calculation nevertheless illustrates why optimization over admissible orderings is structurally meaningful. The larger random 3-SAT candidate-interface experiment is reported separately in Chapter 8 and is likewise interpreted as structural pattern evidence rather than an exact minimum-CPR-Width computation.

6.4 Common Reconstruction Pattern

The representative problem classes considered in this chapter possess different candidate interfaces, semantic interpretations, and instance-specific width expressions; for every realization, one must distinguish $\widetilde{B}_i^{\pi,\tau}$ from $B_i^{\pi,\tau}$. Table 6.1 records the corresponding structural realizations.

Table 6.1: Problem-class realizations of Controlled Partial Reconstruction Width.

Problem class	Structural/interface realization	Semantic interpretation	Width or finite quotient quantity
SAT	Processed variables still affecting unresolved clauses.	Declared reconstruction-safe residual SAT signature satisfying S1–S3.	$\boldsymbol{\omega}_{\mathrm{rec}}(\Phi, \pi)$
CSP	Processed variables shared with unresolved constraints.	Declared reconstruction-safe residual CSP signature satisfying S1–S3.	$\boldsymbol{\omega}_{\mathrm{rec}}(P_{\mathrm{CSP}}, \pi)$
Graph coloring	Processed vertices adjacent to unresolved vertices.	Declared reconstruction-safe residual coloring signature satisfying S1–S3.	$\boldsymbol{\omega}_{\mathrm{rec}}(G, \pi)$
Tensor	Active indices affecting unresolved structural interactions.	Declared reconstruction-safe residual tensor signature satisfying S1–S3.	$\boldsymbol{\omega}_{\mathrm{rec}}(T_P, \pi)$
Full adder (isolated)	No staged CPR interface is asserted here; the finite Hamming-weight output quotient preserves the exact output.	Finite output-equivalence classes; not a staged semantic quotient.	$S_{\mathrm{out}}(P_{\mathrm{FA}}) = 4$

For SAT, CSP, graph coloring, and tensor representations, the displayed structural interfaces are candidate-interface realizations: their use as certified CPR interfaces requires task sufficiency, and as exact minimum CPR interfaces additionally requires cardinality minimality. For the isolated full adder, Table 6.1 records a finite output quotient rather than a stage-wise CPR semantic quotient.

6.5 Theory of Controlled and Constant CPR-Width

The width expressions in Table 6.1 are problem-class-specific instantiations of the general CPR-Width definition, not universal numerical constants for unrestricted problem classes: $\boldsymbol{\omega}_{\mathrm{rec}}(\Phi, \pi)$, $\boldsymbol{\omega}_{\mathrm{rec}}(P_{\mathrm{CSP}}, \pi)$,

$\boldsymbol{\omega}_{\mathrm{rec}}(G,\pi)$, and $\boldsymbol{\omega}_{\mathrm{rec}}(T_P,\pi)$ denote widths of particular represented instances, declared tasks, reconstruction models, and admissible orderings. The isolated full adder is treated differently: its row records the finite output-quotient state count $S_{\mathrm{out}}(P_{\mathrm{FA}}) = 4$, not a stage-wise CPR-Width. A staged CPR realization for arithmetic is introduced separately for the ripple-adder family in Chapter 7.

6.5.1 Conditional Framework for a Partially Reconstructible Subclass

Let $\mathcal{C}$ be a uniformly encoded combinatorial problem class. The following construction gives a conditional family-level framework rather than asserting that such a subclass exists for every problem class. A controlled partially reconstructible subclass $\varnothing \neq \mathcal{C}' \subseteq \mathcal{C}$ is meaningful for asymptotic analysis only when it is itself uniformly encoded and contains instances of unbounded encoding length; if membership in $\mathcal{C}'$ is used as part of an ordinary language-level tractability claim rather than as a promise restriction, membership in the subclass must itself be decidable within the declared polynomial resource model.

Assume one fixed task τ, one fixed uniform reconstruction model $\mathfrak{M}$, and one deterministic uniform preprocessing algorithm $\mathcal{A}_{\mathrm{Preprocess}}$ are declared for the subclass, whose ordering-supply component $\mathcal{A}_\pi$ either reads an admissible ordering from the declared input or constructs one uniformly from the encoding, so that for every $P_x \in \mathcal{C}'$ the supplied ordering satisfies $\pi_x = \mathcal{A}_\pi(x) \in \Pi_{\mathrm{adm}}(P_x, \tau; \mathfrak{M})$; the operation count of $\mathcal{A}_\pi$ is included in N_{Build}. The subclass has *controlled CPR-Width* if there exists one fixed width-bound function $w_{\mathcal{C}'} : \mathbb{N} \to \mathbb{N}_0$ such that $\boldsymbol{\omega}_{\mathrm{rec}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq w_{\mathcal{C}'}(\ell(P_x))$ for every $P_x \in \mathcal{C}'$. The designation is informative only relative to a declared asymptotic growth class for $w_{\mathcal{C}'}$ (typical regimes: constant, logarithmic, polynomial); the mere existence of an unrestricted instance-length bound is not by itself a tractability statement, and the width condition alone does not establish polynomial reconstructibility.

The complete family-level condition additionally requires the four CPR boundary conditions BC1–BC4 to hold uniformly, and, writing $J = \{\mathrm{Build}, \mathrm{Reduce}, \mathrm{Transition}, \mathrm{Reconstruct}, \mathrm{Verify}, \mathrm{Output}\}$, fixed polynomial functions $p_j : \mathbb{N} \to \mathbb{N}$ $(j \in J)$ such that $N_j(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_j(\ell(P_x))$ for every $P_x \in \mathcal{C}'$ and every $j \in J$. In summary, the complete family-level condition fixes one τ, one $\mathfrak{M}$, one $\mathcal{A}_{\mathrm{Preprocess}}$, one width-bound function $w_{\mathcal{C}'}$, BC1–BC4 uniformly for $\mathcal{C}'$, and six fixed polynomial functions $\{p_j\}_{j\in J}$; under these assumptions, for every $P_x \in \mathcal{C}'$,

$$N_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_{\mathrm{CPR}}\big(\ell(P_x)\big) \tag{6.9}$$

for one fixed polynomial p_{CPR}. This formulation keeps the quantifier order uniform and avoids instance-dependent polynomial bounds.

6.5.2 Constant-Width Special Case

A constant-width subclass is a special case of the controlled-width theory. If there exists a constant $k_{\mathcal{C}'} \in \mathbb{N}_0$, independent of the instance size, such that $\boldsymbol{\omega}_{\mathrm{rec}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq k_{\mathcal{C}'}$ for every $P_x \in \mathcal{C}'$, then $\mathcal{C}'$ has uniformly bounded or constant CPR-Width under the declared task, model, and ordering rule, and the corresponding unnormalized quantity satisfies $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq k_{\mathcal{C}'} + 1$. If the local domains are bounded by a constant $1 \leq q(P_x) \leq q_0$, the Width–State Bound gives

$$S_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq q_0^{\,k_{\mathcal{C}'}+1}, \tag{6.10}$$

so the width-dependent state factor is constant with respect to the encoding length. This does not by itself establish polynomial runtime: ordering construction, interface construction, signature construction, semantic quotienting, transition processing, reconstruction, verification, and output must still satisfy the complete polynomial-cost requirements.

6.5.3 Meaning of the Theory

The theory does not claim that every unrestricted problem class has one fixed constant CPR-Width. It states that subclasses may exist whose instances (1) admit a suitable admissible reconstruction ordering, (2) have controlled or constant CPR-Width, (3) satisfy all four CPR boundary conditions, and (4) admit complete polynomial CPR execution for the declared task. Thus the correct statement is not that every problem class has one universal constant width, but rather that a problem class may contain subclasses with controlled or constant CPR-Width. For Table 6.1, this means that $\boldsymbol{\omega}_{\mathrm{rec}}(\Phi, \pi)$, $\boldsymbol{\omega}_{\mathrm{rec}}(P_{\mathrm{CSP}}, \pi)$, $\boldsymbol{\omega}_{\mathrm{rec}}(G, \pi)$, and $\boldsymbol{\omega}_{\mathrm{rec}}(T_P, \pi)$ are the corresponding problem-specific width expressions, which for suitable subclasses must be calculated, estimated, or bounded; the isolated full adder is the exception, its row recording $S_{\mathrm{out}}(P_{\mathrm{FA}})$ rather than a stage-wise CPR-Width.

Width as a resource-separation parameter. The role of CPR-Width is broader than predicting whether a complete candidate space is large or small. Its principal function is to isolate one specific computational resource: the amount of task-relevant processed information that must remain simultaneously available during reconstruction. This permits width to be compared with, but not identified with, input size, search-space size, semantic-state count, reconstruction depth, memory representation size, or runtime. Consequently, small $\boldsymbol{\omega}_{\mathrm{rec}}$ does not imply small runtime, and growing $\boldsymbol{\omega}_{\mathrm{rec}}$ alone does not imply computational intractability: a reconstruction process may have a small active interface while requiring many stages, expensive signature construction, large encoded semantic states, or costly output materialization, while conversely a comparatively large structural interface may coexist with efficient processing when its associated transitions or semantic encodings are inexpensive. For a family of instances, one may therefore study whether runtime growth is associated primarily with growth of the active reconstruction interface, growth

of the semantic quotient, increasing reconstruction depth, increasing cost of computing or updating the reconstruction-safe signature, increasing verification or output cost, or another resource not controlled by CPR-Width. This diagnostic interpretation is why the framework retains the dependency chain $\boldsymbol{\omega}_{\mathrm{rec}} \to S_{\mathrm{CPR}} \to N_{\mathrm{CPR}}$ rather than treating the quantities as equivalent.

6.6 Common Reconstruction Chain

Despite their structural differences, the representative staged realizations follow the common pattern

$$\pi \longrightarrow \widetilde{B}_i^{\pi,\tau} \longrightarrow B_i^{\pi,\tau} \longrightarrow U_i^{\mathrm{reach}} \longrightarrow \widehat{\mathrm{Sig}}_{\tau,i}^{\pi} \longrightarrow S_i \longrightarrow \overline{\delta}_i^{\pi,\tau} \longrightarrow \mathrm{Output}_\tau, \tag{6.11}$$

i.e. ordering → structural candidate interface → task-sufficient interface → reachable interface states → induced reconstruction-safe signature → semantic quotient → quotient transitions → reconstruction/verification/output. The candidate interface is problem-class dependent; task sufficiency, cardinality minimality, reconstruction-safe semantic equivalence, quotient transitions, and complete runtime accounting are governed by the general CPR-WIDTH theory.

6.7 Tensor Representations

Let $T_P : I(P)\times J(P)\times K(P) \to \mathcal{A}$ be a finite structural tensor associated with a problem instance P. For the CPR realization, fix explicitly a finite reconstruction-component carrier $V(T_P) = \{\alpha_1, \ldots, \alpha_m\}$, where each component $\alpha \in V(T_P)$ has a finite nonempty domain D_α; depending on the representation, α may denote a tensor index, a tensor fragment, a local compatibility relation, or another explicitly encoded finite structural component. An admissible reconstruction ordering is a permutation $\pi = (\alpha_{\pi(1)}, \ldots, \alpha_{\pi(m)})$ of $V(T_P)$, with processed prefix $V_i^\pi = \{\alpha_{\pi(1)}, \ldots, \alpha_{\pi(i)}\}$. The use of "tensor" here is structural: it denotes a finite multi-indexed array serving as a reconstruction representation, not an assertion about multilinear-algebraic tensor rank. Define a structural tensor candidate interface by

$$\widetilde{B}_i^{T_P,\pi} = \big\{\alpha \in V_i^\pi : \alpha \text{ still influences an unresolved tensor interaction}\big\}. \tag{6.12}$$

A reconstruction-safe tensor signature must preserve the task-relevant future behavior required by S1–S3; depending on the declared task, it may encode a certified residual output class, a task-relevant structural output vector, an exact continuation object, or another transition-congruent finite signature, and reconstruction safety must be proved for whatever signature is selected. If $\widetilde{B}_i^{T_P,\pi}$ determines the selected reconstruction-safe signature, Bound (6.1) gives $b_i^{\pi,\tau}(T_P) \le |\widetilde{B}_i^{T_P,\pi}|$, hence $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(T_P,\pi,\tau;\mathfrak{M}) \le \max_i |\widetilde{B}_i^{T_P,\pi}|$ and $\boldsymbol{\omega}_{\mathrm{rec}}(T_P,\pi,\tau;\mathfrak{M}) = \max\{0, \widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(T_P,\pi,\tau;\mathfrak{M})-1\}$. Tensor representation therefore supplies a structural carrier; it does not by itself prove small CPR-Width, efficient semantic quotienting, or polynomial runtime.

6.8 Full Adder

The isolated full adder provides a finite exact output-preserving quotient. Let $\mathbb{B} = \{0,1\}$ and $X_{\mathrm{FA}} = \mathbb{B}^3$, with input states $x = (a, b, c_{\mathrm{in}})$ and full-adder output map $F_{\mathrm{FA}} : \mathbb{B}^3 \to \mathbb{B}^2$. Define the Hamming-weight projection $C_{\mathrm{FA}}(a, b, c_{\mathrm{in}}) = a + b + c_{\mathrm{in}}$, so $C_{\mathrm{FA}} : \mathbb{B}^3 \to \{0,1,2,3\}$, and $Q_{\mathrm{FA}}(h) = \big(h \bmod 2, \lfloor h/2 \rfloor\big)$. Then $F_{\mathrm{FA}} = Q_{\mathrm{FA}} \circ C_{\mathrm{FA}}$, so the eight input states collapse into four exact output-equivalence classes, $8 \to 4$, and for the isolated full-adder output task $S_{\mathrm{out}}(P_{\mathrm{FA}}) = 4$.

This is an exact finite output quotient. It must not automatically be identified with $S_{\mathrm{CPR}}(P_{\mathrm{FA}}) = 4$, because no staged CPR reconstruction model for the isolated full-adder block is declared in this section, and likewise no stage-wise value of $\boldsymbol{\omega}_{\mathrm{rec}}(P_{\mathrm{FA}}, \pi)$ is claimed here. Chapters 7 and 8 preserve this distinction: the isolated full-adder factorization is treated as a finite output quotient, while the ripple-adder family receives a separate staged CPR reconstruction model with its own interface, semantic-state, transition, and runtime bounds.

6.9 Synthesis, Scope, and Non-Claims

The problem classes considered in this chapter possess different structural realizations: for SAT the candidate boundary is induced by residual clauses, for CSP by unresolved constraint scopes, for graph coloring by edges crossing the processed and unresolved vertex sets, and for tensor representations by unresolved structural interactions. The isolated full adder is treated separately through its finite output-preserving quotient, and its staged arithmetic CPR realization is deferred to the ripple-adder model of Chapter 7. For the staged structural realizations, a candidate interface becomes a certified CPR interface only after task sufficiency has been proved, and an exact minimum CPR interface only after cardinality minimality has also been proved. Accordingly, every problem-class realization should establish, in order: (1) an explicit component representation; (2) a declared task; (3) an admissible reconstruction ordering; (4) a structural candidate interface; (5) a reconstruction-safe signature satisfying S1–S3; (6) task sufficiency of the candidate interface; (7) cardinality minimality if exact width equality is claimed; (8) reachable interface-state and semantic quotient definitions; (9) semantic transition control; and (10) all six complete runtime-cost bounds whenever a tractability statement is intended.

The realizations in this chapter do not establish that unrestricted SAT, CSP, graph coloring, or tensor reconstruction has small CPR-Width; that every structural candidate interface is minimal; that reconstruction-safe signatures are always polynomially computable; that a favorable ordering is always

polynomially constructible; or that polynomial runtime follows from width alone. The supported conclusion is narrower:

> The combinatorial realizations provide problem-specific candidate interface models that can be analyzed within CPR-WIDTH, while the isolated full adder provides an exact finite output-preserving quotient that motivates the staged arithmetic realization developed in Chapter 7.

A tractability conclusion is permitted only for explicitly defined uniform families or subclasses satisfying BC1–BC4 and fixed polynomial bounds for all six runtime components.

6.10 Transfer to Width-Controlled Reconstruction Beyond Finite Combinatorics

The formal CPR-WIDTH framework developed in this manuscript is defined for finite combinatorial reconstruction. The underlying design principle may nevertheless motivate future domain-specific reconstruction parameters outside this setting, but such a transfer must not be interpreted as a direct reuse of $\boldsymbol{\omega}_{\mathrm{rec}}$ or $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}$ without a separate derivation. The transferable idea is only the structural principle that reconstruction stages determine active task-relevant information, which in turn controls the continuation and hence the reconstruction cost. A different mathematical domain may introduce a distinct width-controlled reconstruction parameter only after specifying its own reconstruction objects, stage structure, admissibility rules, task-relative state or signature notion, correctness conditions, reconstruction operator, and computational cost model; no CPR tractability theorem, approximation guarantee, numerical accuracy result, or complexity conclusion transfers automatically from the finite combinatorial framework to such a domain. This section therefore records only a possible methodological direction. All formal results of the present manuscript remain restricted to the finite reconstruction models explicitly defined in Chapters 2–9.

6.11 Conclusion

This chapter developed representative realizations of CPR-Width for SAT, CSP, graph coloring, tensor structures, and the isolated full adder. For the staged combinatorial realizations, the principal width expressions $\boldsymbol{\omega}_{\mathrm{rec}}(\Phi,\pi)$, $\boldsymbol{\omega}_{\mathrm{rec}}(P_{\mathrm{CSP}},\pi)$, $\boldsymbol{\omega}_{\mathrm{rec}}(G,\pi)$, and $\boldsymbol{\omega}_{\mathrm{rec}}(T_P,\pi)$ do not denote one universal width for an unrestricted problem class; they denote task-, model-, representation-, and ordering-relative quantities that must be determined for the declared reconstruction setting, subject to the general candidate bound (6.1) and equality only after cardinality minimality. For the isolated full adder, the relevant finite quantity is instead $S_{\mathrm{out}}(P_{\mathrm{FA}}) = 4$, not a stage-wise CPR semantic-state count. A problem class may contain a controlled partially reconstructible subclass $\mathcal{C}' \subseteq \mathcal{C}$ whose uniformly supplied orderings satisfy $\boldsymbol{\omega}_{\mathrm{rec}}(P_x,\pi_x,\tau;\mathfrak{M}) \le w_{\mathcal{C}'}(\ell(P_x))$, with a constant-width subclass obtained when $w_{\mathcal{C}'}$ may be taken as a constant $k_{\mathcal{C}'}$ independent of instance size. Neither controlled width nor constant width alone implies polynomial runtime; the complete family-level tractability requirement remains

$$\boxed{\mathrm{BC1} \wedge \mathrm{BC2} \wedge \mathrm{BC3} \wedge \mathrm{BC4} \ \wedge \ \bigwedge_{j\in J}\Big[N_j(P_x,\pi_x,\tau;\mathfrak{M}) \le p_j\big(\ell(P_x)\big)\Big].} \tag{6.13}$$

Thus, Chapter 6 identifies structural realizations of CPR-Width across several representative problem classes and formulates the conditions under which controlled or constant-width subclasses may become fully CPR-tractable. The following chapter develops the fully staged ripple-adder reconstruction model.

7 Adder and Arithmetic Reconstruction

Digital addition provides an exactly verifiable CPR-WIDTH realization. The purpose is not a new addition algorithm, but a structural and operation-count analysis of standard binary addition.

Two constructions are distinguished. The isolated full adder has an exact output-preserving quotient from eight inputs to four Hamming-weight classes; this output quotient is not automatically a stage-wise CPR semantic-state count. The n-bit ripple-adder family is represented below by a uniform all-input staged transducer whose future behavior is controlled by one Boolean carry state.

The declared CPR task of this staged schema is denoted by

$$\tau_{\mathrm{sch}},$$

and consists of preserving and materializing the complete stage-wise labeled carry-transition relation of the ripple-adder transducer.

For a fixed encoded input

$$x = (A, B, c_0),$$

the corresponding bits select one deterministic path through this schema. Exact forward addition along that selected path yields

$$(s_0, s_1, \ldots, s_{n-1}, c_n),$$

but this fixed-input evaluation is not introduced as a separate CPR-Width task or reconstruction model. All operation-count statements in this chapter use the fixed elementary-operation accounting model of Chapter 4. This chapter develops the arithmetic reconstruction model and proves its structural and asymptotic properties; Chapter 8 records the exhaustive finite verification of the isolated full-adder quotient.

7.1 Full-Adder Map and Hamming-Weight Factorization

Let $\mathbb{B} = \{0, 1\}$. A local full-adder input state is $x = (a, b, c_{\text{in}}) \in \mathbb{B}^3$, where a and b are input bits and c_{in} is the incoming carry bit; the corresponding output state is $F_{\text{FA}}(x) = (s, c_{\text{out}}) \in \mathbb{B}^2$, where s is the sum bit and c_{out} the outgoing carry bit. The full-adder map $F_{\text{FA}} : \mathbb{B}^3 \to \mathbb{B}^2$ is defined by $s = a \oplus b \oplus c_{\text{in}}$ and $c_{\text{out}} = ab \vee ac_{\text{in}} \vee bc_{\text{in}}$, equivalently $a + b + c_{\text{in}} = s + 2c_{\text{out}}$; the complete truth table is shown in Table 7.1.

Table 7.1: Full-adder truth table.

a	b	c_{in}	s	c_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Define the Hamming-weight projection $C_{\text{FA}} : \mathbb{B}^3 \to \{0, 1, 2, 3\}$ by $C_{\text{FA}}(a, b, c_{\text{in}}) = a + b + c_{\text{in}}$; the eight Boolean input states are mapped to four Hamming-weight states, $8 \to 4$. Let $Z_{\text{FA}} = \{0, 1, 2, 3\}$ be the reduced Hamming-weight space and define $Q_{\text{FA}} : Z_{\text{FA}} \to \mathbb{B}^2$ by $Q_{\text{FA}}(h) = (h \bmod 2, \lfloor h/2 \rfloor)$; then $F_{\text{FA}} = Q_{\text{FA}} \circ C_{\text{FA}}$.

Theorem 7.1 (Exact Full-Adder Factorization)**.** *For every input state $x = (a, b, c_{\text{in}}) \in \mathbb{B}^3$, the complete full-adder output is determined uniquely by the Hamming weight $h = C_{\text{FA}}(x)$; consequently $F_{\text{FA}}(x) = Q_{\text{FA}}(C_{\text{FA}}(x))$.*

Proof. Let $h = a + b + c_{\text{in}}$; since $h \in \{0, 1, 2, 3\}$, its parity determines the sum bit, $s = h \bmod 2$, and its integer quotient by two determines the outgoing carry, $c_{\text{out}} = \lfloor h/2 \rfloor$. Therefore the output pair (s, c_{out}) is uniquely determined by h, so $F_{\text{FA}} = Q_{\text{FA}} \circ C_{\text{FA}}$. □

The factorization preserves the complete declared output; it does not preserve the original input state uniquely. For every $h \in Z_{\text{FA}}$, the Hamming-weight fiber is $\Omega_{\text{FA}}(h) = C_{\text{FA}}^{-1}(\{h\})$; the four fibers are $\Omega_{\text{FA}}(0) = \{(0, 0, 0)\}$, $\Omega_{\text{FA}}(1) = \{(1, 0, 0), (0, 1, 0), (0, 0, 1)\}$, $\Omega_{\text{FA}}(2) = \{(1, 1, 0), (1, 0, 1), (0, 1, 1)\}$, and $\Omega_{\text{FA}}(3) = \{(1, 1, 1)\}$, with cardinalities $(1, 3, 3, 1)$, so the maximum fiber size is $d_{\text{fiber}}^{\max}(P_{\text{FA}}) = 3$. The map Q_{FA} is injective on Z_{FA} ($0 \mapsto (0, 0)$, $1 \mapsto (1, 0)$, $2 \mapsto (0, 1)$, $3 \mapsto (1, 1)$), so $C_{\text{FA}}(x) = C_{\text{FA}}(y) \Leftrightarrow F_{\text{FA}}(x) = F_{\text{FA}}(y)$: the Hamming-weight fibers are exactly the output-equivalence classes of the isolated full adder.

For the isolated full-adder output task, define $x \equiv_{\text{out}} y \Leftrightarrow F_{\text{FA}}(x) = F_{\text{FA}}(y)$; the output quotient $\mathbb{B}^3/\equiv_{\text{out}} = \{\Omega_{\text{FA}}(0), \ldots, \Omega_{\text{FA}}(3)\}$ has cardinality $S_{\text{out}}(P_{\text{FA}}) = |\mathbb{B}^3/\equiv_{\text{out}}| = 4$, with binary output-state dimension $D_{\text{out}}(P_{\text{FA}}) = \lceil \log_2 S_{\text{out}}(P_{\text{FA}}) \rceil = 2$. The quantities S_{out} and D_{out} refer to the isolated output quotient and are not identified automatically with S_{CPR} and D_{CPR}.

7.2 Ripple-Adder Function Family

For every $n \geq 1$, define the n-bit ripple-adder function $F_{\text{add},n} : \mathbb{B}^n \times \mathbb{B}^n \times \mathbb{B} \to \mathbb{B}^{n+1}$, with input (A, B, c_0), $A = (a_{n-1} \cdots a_1 a_0)_2$, $B = (b_{n-1} \cdots b_1 b_0)_2$, and output $(s_0, s_1, \ldots, s_{n-1}, c_n)$. For every position $i \in \{0, \ldots, n-1\}$, define $h_i = a_i + b_i + c_i$, $s_i = h_i \bmod 2$, and $c_{i+1} = \lfloor h_i/2 \rfloor$; the complete arithmetic identity is $A + B + c_0 = \sum_{i=0}^{n-1} s_i 2^i + c_n 2^n$. The CPR model developed below is a uniform transducer model for the complete function $F_{\text{add},n}$: a fixed encoded input (A, B, c_0) determines one deterministic path through this transducer.

Define the carry-state alphabet $\mathcal{Q}_{\text{carry}} = \{0, 1\}$ and local input alphabet $\mathcal{X}_{\text{local}} = \mathbb{B}^2$, with local input label $\alpha_i = (a_i, b_i) \in \mathcal{X}_{\text{local}}$. Define the carry transition map $\delta_{\text{carry}} : \mathcal{Q}_{\text{carry}} \times \mathcal{X}_{\text{local}} \to \mathcal{Q}_{\text{carry}}$ by $\delta_{\text{carry}}(c, (a, b)) = \lfloor (a + b + c)/2 \rfloor$, and the local output map $\mu_{\text{carry}} : \mathcal{Q}_{\text{carry}} \times \mathcal{X}_{\text{local}} \to \mathbb{B}$ by $\mu_{\text{carry}}(c, (a, b)) = (a + b + c) \bmod 2$; thus every local step is $(c_i, (a_i, b_i)) \mapsto (s_i, c_{i+1})$. The transducer has two carry states, and for every carry state four local input labels are possible, so its complete local labeled transition table contains at most $2 \cdot 4 = 8$ labeled transitions – a property of the uniform family-level transducer, not the number of transitions executed during one fixed addition run.

7.3 CPR Component Representation and Reachability

From this point onward, the structural CPR object is a single all-input transduction schema. A fixed encoded addition input selects one deterministic execution path through that schema, but it does not define a second CPR-width object.

Declared schema object. Let

$$P_{\text{add},n}^{\text{sch}}$$

denote the n-stage ripple-adder transduction schema. Its declared task

$$\tau_{\mathrm{sch}}$$

is to preserve the complete stage-wise carry-transition relation of the ripple-adder transducer, including the emitted sum bit associated with each local transition.
For bit position $i = 0$, define the first stage record

$$\rho_0 = (c_0, a_0, b_0, s_0, c_1)$$

with domain

$$D_{\rho_0} = \left\{(c, a, b, s, c') \in \mathbb{B}^5 : a + b + c = s + 2c'\right\}.$$

For $1 \le i \le n-1$, define

$$\rho_i = (a_i, b_i, s_i, c_{i+1}), \qquad D_{\rho_i} = \mathbb{B}^4.$$

The reconstruction-component set is

$$V(P^{\mathrm{sch}}_{\mathrm{add},n}) = \{\rho_0, \rho_1, \ldots, \rho_{n-1}\},$$

with the admissible ordering

$$\pi_{\mathrm{add}} = (\rho_0, \rho_1, \ldots, \rho_{n-1}).$$

Declared reconstruction model. The structural model used in this chapter is

$$\mathfrak{M}_{\mathrm{sch}} = (\mathrm{Adm}_{\mathrm{sch}}, \mathcal{R}_{\mathrm{sch}}, \mathrm{Sig}_{\mathrm{sch}}),$$

in the sense of Definition 2.1. Here $\mathrm{Adm}_{\mathrm{sch}}$ accepts the declared stage order π_{add}; $\mathcal{R}_{\mathrm{sch}}$ consists exactly of locally consistent prefixes of stage records satisfying the ripple-adder identity; and $\mathrm{Sig}_{\mathrm{sch}}$ is the carry signature defined in Section 7.4. The admissible next-value and transition maps are the canonical objects derived from this reachable-prefix rule.
For a reachable schema prefix

$$r = (\rho_0, \ldots, \rho_{i-1}) \in \mathcal{R}_i^{\pi_{\mathrm{add}}}(P^{\mathrm{sch}}_{\mathrm{add},n}), \qquad 1 \le i < n,$$

let $c_i(r)$ denote the outgoing carry contained in the last processed record. A candidate next component value

$$z = (a_i, b_i, s_i, c_{i+1}) \in D_{\rho_i}$$

is admissible exactly when

$$a_i + b_i + c_i(r) = s_i + 2c_{i+1}.$$

Thus

$$\Lambda_i^{\pi_{\mathrm{add}}}(r) = \left\{(a_i, b_i, s_i, c_{i+1}) \in \mathbb{B}^4 : a_i + b_i + c_i(r) = s_i + 2c_{i+1}\right\},$$

and

$$\delta_i^{\pi_{\mathrm{add}}}(r, z) = r \cup \{\rho_i \mapsto z\}.$$

At stage zero,

$$\Lambda_0^{\pi_{\mathrm{add}}}(\varnothing) = D_{\rho_0}.$$

For every internal stage $1 \le i < n$, both carry values occur among reachable schema prefixes:

$$\{0, 1\} \subseteq \left\{c_i(r) : r \in \mathcal{R}_i^{\pi_{\mathrm{add}}}(P^{\mathrm{sch}}_{\mathrm{add},n})\right\}.$$

Fixed-input evaluation path. For a fixed encoded input

$$x = (A, B, c_0),$$

the bits a_i, b_i select one deterministic path through $P^{\mathrm{sch}}_{\mathrm{add},n}$. The path is used below to prove exact forward evaluation and its linear execution cost. No separate fixed-input CPR-Width is introduced: the structural CPR-Width theorem of this chapter refers only to the declared schema object $P^{\mathrm{sch}}_{\mathrm{add},n}$.

7.4 Carry Signature, Interface Sufficiency, and Minimality

For stage $i = 0$, define

$$\mathrm{Sig}^{\pi_{\mathrm{add}}}_{\tau_{\mathrm{sch}},0}(\varnothing) = \textsc{start}.$$

For every internal stage $1 \le i < n$, define

$$\mathrm{Sig}^{\pi_{\mathrm{add}}}_{\tau_{\mathrm{sch}},i}(r) = c_i(r).$$

At the terminal stage define

$$\mathrm{Sig}^{\pi_{\mathrm{add}}}_{\tau_{\mathrm{sch}},n}(r) = \textsc{accept}.$$

Stage-relative schema task value. For every internal stage $1 \le i < n$, define

$$\mathrm{Ans}^{\pi_{\mathrm{add}}}_{\tau_{\mathrm{sch}},i}(r)$$

to be the residual labeled carry-transition relation of the unresolved suffix, conditional on the current carry entering stage i. Thus the stage-relative evaluation map $\Theta^{\pi_{\mathrm{add}}}_{\tau_{\mathrm{sch}},i}$ for this task returns that residual transducer

relation. At stage 0, the answer is the complete n-stage relation, and at stage n the unresolved relation is empty and hence constant. This explicitly instantiates the task-relative value required by Section 2.5.

Proposition 7.1 (Carry Signature Is Reconstruction-Safe)**.** *The preceding signature family is reconstruction-safe for* $P^{\mathrm{sch}}_{\mathrm{add},n}$ *and* τ_{sch}.

Proof. At stage 0, the reachable-prefix space is a singleton, so S1–S3 hold vacuously.
For every internal stage $1 \le i < n$, equal carry signatures mean

$$c_i(r) = c_i(r').$$

By the definition of the stage-relative schema task value, the residual labeled carry-transition relation of the unresolved suffix depends on the processed prefix only through this incoming carry. Therefore

$$\mathrm{Ans}^{\pi_{\mathrm{add}}}_{\tau_{\mathrm{sch}},i}(r) = \mathrm{Ans}^{\pi_{\mathrm{add}}}_{\tau_{\mathrm{sch}},i}(r'),$$

so S1 holds.
For S2, the defining condition

$$a_i + b_i + c_i(r) = s_i + 2c_{i+1}$$

is identical for r and r', so

$$\Lambda^{\pi_{\mathrm{add}}}_i(r) = \Lambda^{\pi_{\mathrm{add}}}_i(r').$$

For S3, let the same admissible next stage record

$$z = (a_i, b_i, s_i, c_{i+1})$$

be applied to both prefixes. If $i < n-1$, both successor signatures equal the outgoing-carry field c_{i+1} contained in z. If $i = n-1$, both successors are terminal and receive the constant signature ACCEPT. Thus S3 holds. □

For every internal stage $1 \le i < n$, define the schema candidate interface

$$\widetilde{B}^{\pi_{\mathrm{add}},\tau_{\mathrm{sch}}}_i = \{\rho_{i-1}\},$$

with

$$\widetilde{B}^{\pi_{\mathrm{add}},\tau_{\mathrm{sch}}}_0 = \widetilde{B}^{\pi_{\mathrm{add}},\tau_{\mathrm{sch}}}_n = \varnothing.$$

Proposition 7.2 (Carry-Interface Sufficiency)**.** *For every internal stage* $1 \le i < n$,

$$\widetilde{B}^{\pi_{\mathrm{add}},\tau_{\mathrm{sch}}}_i = \{\rho_{i-1}\}$$

is task-sufficient for the carry signature.

Proof. If two reachable schema prefixes agree on ρ_{i-1}, then they agree on its outgoing-carry field and therefore have the same carry signature. □

Proposition 7.3 (Carry-Interface Minimality)**.** *For every internal stage* $1 \le i < n$, *the empty interface is not task-sufficient. Consequently,*

$$b^{\pi_{\mathrm{add}},\tau_{\mathrm{sch}}}_i(P^{\mathrm{sch}}_{\mathrm{add},n}) = 1$$

for every internal stage when $n \ge 2$.

Proof. At every internal stage, there exist reachable schema prefixes r_0, r_1 with

$$c_i(r_0) = 0, \qquad c_i(r_1) = 1.$$

They agree on the empty interface but have different carry signatures, so the empty interface is not task-sufficient. The preceding proposition supplies a task-sufficient singleton interface, proving minimality. □

The unnormalized schema width is therefore

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P^{\mathrm{sch}}_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{sch}}; \mathfrak{M}_{\mathrm{sch}}) = \begin{cases} 0, & n = 1, \\ 1, & n \ge 2. \end{cases} \tag{7.1}$$

By the normalization convention,

$$\boldsymbol{\omega}_{\mathrm{rec}}(P^{\mathrm{sch}}_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{sch}}; \mathfrak{M}_{\mathrm{sch}}) = 0 \qquad (n \ge 1). \tag{7.2}$$

The schema-width statement does not claim that a deterministic fixed-input execution requires zero working information. During such an execution the current carry is the stage-local state used to select the next transition; it is precisely the information represented by the one-component schema interface.

7.5 Evaluation, Semantic-State Control, and Runtime

Theorem 7.2 (Exact Ripple-Adder Evaluation)**.** *For every* $n \ge 1$ *and every fixed encoded input* $x = (A, B, c_0)$, *the declared schema* $P^{\mathrm{sch}}_{\mathrm{add},n}$ *has one deterministic execution path selected by* x, *and that path produces uniquely the complete output*

$$(s_0, s_1, \ldots, s_{n-1}, c_n).$$

Proof. At stage zero, fixed a_0, b_0, c_0 determine uniquely s_0, c_1 through

$$a_0 + b_0 + c_0 = s_0 + 2c_1.$$

Assume inductively that c_i has been determined. The fixed bits a_i, b_i then determine uniquely s_i, c_{i+1} from

$$a_i + b_i + c_i = s_i + 2c_{i+1}.$$

Induction gives all n sum bits and the final carry. □

Output materialization on a selected path. When a fixed encoded input selects one path through the schema, the sum bit s_i produced by the chosen transition is emitted irreversibly after stage i, and the final carry c_n is emitted at the terminal stage. The current carry c_i remains the stage-local transition state until the next transition has been processed.

Schema semantic quotient. The semantic quotient induced by the carry signature has the constant carry alphabet

$$\mathcal{Q}_{\text{carry}} = \{0, 1\}, \qquad |\mathcal{Q}_{\text{carry}}| = 2. \tag{7.3}$$

Hence every internal schema stage has at most two semantic states.

Definition 7.1 (Total Stage-Indexed Semantic-State Count)**.** Define

$$V_{\text{sem}}(P, \pi, \tau; \mathfrak{M}) = \sum_{i=0}^{m_{\text{stage}}(P,\pi)} |S_i(P, \pi, \tau; \mathfrak{M})|.$$

For the schema,

$$V_{\text{sem}}(P^{\text{sch}}_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{sch}}; \mathfrak{M}_{\text{sch}}) \le 2(n + 1). \tag{7.4}$$

Remark 7.1. The quantity

$$V_{\text{sem}}(P, \pi, \tau; \mathfrak{M})$$

is a total stage-indexed semantic-state count obtained by summing the cardinalities of the semantic quotients over all reconstruction stages. It must not be identified with

$$S_{\text{CPR}}(P, \pi, \tau; \mathfrak{M}),$$

which denotes the maximum semantic-state cardinality attained at any single reconstruction stage.

Definition 7.2 (Total Semantic-Transition Count)**.** Let

$$E^{\text{sem}}_i(P, \pi, \tau; \mathfrak{M}) \subseteq S_i(P, \pi, \tau; \mathfrak{M}) \times D_{w^{\pi}_i} \times S_{i+1}(P, \pi, \tau; \mathfrak{M})$$

be the set of reachable labeled quotient transitions (σ, a, σ'), and define

$$E_{\text{sem}}(P, \pi, \tau; \mathfrak{M}) = \sum_{i=0}^{m_{\text{stage}}(P,\pi)-1} |E^{\text{sem}}_i(P, \pi, \tau; \mathfrak{M})|.$$

For a fixed current carry value, four input pairs (a_i, b_i) are possible and each determines one valid (s_i, c_{i+1}). Thus the schema has at most eight labeled quotient transitions per bit position:

$$E_{\text{sem}}(P^{\text{sch}}_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{sch}}; \mathfrak{M}_{\text{sch}}) \le 8n. \tag{7.5}$$

7.5.1 Construction Complexity of the Semantic Quotient

The incremental signature update is

$$\text{upd}_i(c_i, (a_i, b_i, s_i, c_{i+1})) = c_{i+1}. \tag{7.6}$$

For a fixed input path, the selected next record is computed by

$$c_{i+1} = \left\lfloor \frac{a_i + b_i + c_i}{2} \right\rfloor, \qquad s_i = (a_i + b_i + c_i) \bmod 2. \tag{7.7}$$

Proposition 7.4 (Linear Semantic-Quotient Construction)**.** *For the n-stage schema $P^{\text{sch}}_{\text{add},n}$, the stage-indexed semantic quotient and its reachable labeled transition structure have $O(n)$ total size and can be instantiated in $O(n)$ elementary operations.*

Proof. The semantic alphabet has two values and each stage has at most eight labeled quotient transitions. Equations 7.4 and 7.5 therefore give linear total vertex and edge counts. The local admissibility test and carry update use constant-size Boolean records, so each stage requires $O(1)$ work. □

Complete CPR operation count for the declared schema task. Use as the executing CPR procedure the uniform construction and materialization of the stage-indexed carry-transition relation declared by τ_{sch}. Since each stage descriptor, carry state, local transition record, verification identity, and output record has constant local type, there exist fixed constants $C_{\text{Build}}, C_{\text{Reduce}}, C_{\text{Transition}}, C_{\text{Reconstruct}}, C_{\text{Verify}}, C_{\text{Output}} > 0$, independent of n, such that

$$N_j(P^{\text{sch}}_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{sch}}; \mathfrak{M}_{\text{sch}}) \le C_j(n + 1)$$

for every $j \in \{\text{Build}, \text{Reduce}, \text{Transition}, \text{Reconstruct}, \text{Verify}, \text{Output}\}$. In particular,

$$N_{\text{Build}} = O(n), \quad N_{\text{Reduce}} = O(n), \quad N_{\text{Transition}} = O(n),$$
$$N_{\text{Reconstruct}} = O(n), \quad N_{\text{Verify}} = O(n), \quad N_{\text{Output}} = \Theta(n).$$

The output lower bound follows because the declared schema output contains a constant-size transition record for each of the n stages. Therefore

$$N_{\text{CPR}}(P^{\text{sch}}_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{sch}}; \mathfrak{M}_{\text{sch}}) = \Theta(n). \tag{7.8}$$

Thus the six component-wise polynomial bounds and the width theorem (7.1) refer to exactly the same declared CPR object, task, ordering, and reconstruction model.

Per-input evaluation cost. Once the schema has been declared, a fixed encoded input $x = (A, B, c_0)$ selects one transition at each stage. The standard forward evaluation therefore performs n constant-size local updates and writes $n + 1$ output bits, so its separate evaluation cost is

$$N_{\text{eval}}(n, x) = \Theta(n).$$

This per-input evaluation statement is not substituted for the schema-level quantity N_{CPR} in Equation 7.8.

7.6 Boundary-Condition Record

Define the uniformly encoded schema family

$$\mathcal{F}^{\text{sch}}_{\text{add}} = \left\{ P^{\text{sch}}_{\text{add},n} : n \geq 1 \right\}.$$

A schema instance contains n stage descriptors of constant local type, so

$$\ell(P^{\text{sch}}_{\text{add},n}) = \Theta(n).$$

BC1 – Explicit Uniform Structural Representation. For every n, the component representation, local domains, admissible ordering, reachable-prefix rule, carry signature, semantic quotient, labeled transition relation, verification identity, and declared schema output representation are explicitly specified.

BC2 – Polynomial Uniform Constructibility. The n stage descriptors, carry-signature data, and complete constant-size transition table per stage are constructed uniformly in $O(n)$ elementary operations.

BC3 – Task-Relative Soundness and Completeness. The declared task is the complete stage-wise labeled carry-transition relation. By construction, every locally valid ripple-adder transition is present and every listed transition satisfies

$$a_i + b_i + c_i = s_i + 2c_{i+1}.$$

Hence the materialized relation is sound and complete for τ_{sch}.

BC4 – Polynomial Semantic-State and Transition-Structure Control. Equations 7.4 and 7.5 give

$$V_{\text{sem}} \leq 2(n + 1) = O(n), \qquad E_{\text{sem}} \leq 8n = O(n).$$

Together with the six component-wise bounds above, $\mathcal{F}^{\text{sch}}_{\text{add}}$ satisfies the Chapter 5 certification conditions for its declared schema task and obeys

$$N_{\text{CPR}}(P^{\text{sch}}_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{sch}}; \mathfrak{M}_{\text{sch}}) = \Theta(n).$$

The conclusion concerns this explicitly declared schema task. It is not inferred from constant width alone and it is not a claim that an unrelated counting or search task inherits the same CPR certificate.

Theorem 7.3 (Uniform CPR Certificate for the Ripple-Adder Schema Family)**.** *Let*

$$\mathcal{F}^{\text{sch}}_{\text{add}} = \{P^{\text{sch}}_{\text{add},n} : n \geq 1\}$$

be the uniformly encoded ripple-adder schema family, equipped with the declared schema task τ_{sch}, the uniform reconstruction model $\mathfrak{M}_{\text{sch}}$, and the least-significant-bit to most-significant-bit ordering π_{add}. Then the following statements hold uniformly for every $n \geq 1$.

1. *The carry-signature family is reconstruction-safe and satisfies S1–S3.*
2. *For every internal stage $1 \leq i \leq n - 1$, the cardinality-minimal task-sufficient interface is $\{\rho_{i-1}\}$, the singleton containing the most recently processed stage record. For $n = 1$, no internal stage $1 \leq i \leq n - 1$ exists; hence the maximum minimum-interface size over the complete one-stage schema is zero. Consequently,*

$$\widehat{\boldsymbol{\omega}}_{\text{rec}}(P^{\text{sch}}_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{sch}}; \mathfrak{M}_{\text{sch}}) = \begin{cases} 0, & n = 1, \\ 1, & n \geq 2, \end{cases}$$

   and

$$\boldsymbol{\omega}_{\text{rec}}(P^{\text{sch}}_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{sch}}; \mathfrak{M}_{\text{sch}}) = 0.$$

3. *The stage-indexed semantic quotient and transition graph satisfy*

$$V_{\text{sem}}(P^{\text{sch}}_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{sch}}; \mathfrak{M}_{\text{sch}}) \leq 2(n + 1)$$

   and

$$E_{\text{sem}}(P^{\text{sch}}_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{sch}}; \mathfrak{M}_{\text{sch}}) \leq 8n.$$

4. *BC1–BC4 hold uniformly for the declared schema task.*

5. *All six runtime components admit uniform linear upper bounds. More precisely, there exist fixed linear polynomials $p_j(n) = C_j(n+1)$, one for each of the six runtime components, such that*

$$N_j(P^{\mathrm{sch}}_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{sch}}; \mathfrak{M}_{\mathrm{sch}}) \le p_j(n)$$

for every $j \in \{\mathrm{Build}, \mathrm{Reduce}, \mathrm{Transition}, \mathrm{Reconstruct}, \mathrm{Verify}, \mathrm{Output}\}$.

6. *The declared schema output contains one constant-size transition record for each of the n stages, so* $N_{\mathrm{Output}} = \Omega(n)$. *Together with the uniform* $O(n)$ *upper bounds for all six components, this yields*

$$N_{\mathrm{CPR}}(P^{\mathrm{sch}}_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{sch}}; \mathfrak{M}_{\mathrm{sch}}) = \Theta(n).$$

Hence the ripple-adder schema family possesses a complete uniform CPR certificate for the declared schema task: bounded operational interface width, polynomial semantic structure, BC1–BC4, and polynomial bounds for all six components of the complete runtime model are established simultaneously for one and the same family-level construction.

Task-relative status of the certificate. BC1–BC4 are applied here in their task-relative form to the declared relation/transducer task τ_{sch}. Theorem 5.1 and the decision-language class $\mathbf{P}_{\mathrm{CPR}}$ are not invoked, because τ_{sch} is not a decision task. Accordingly, Theorem 7.3 is a complete CPR certificate for a uniform transducer family; it is not a statement that a decision language L belongs to $\mathbf{P}_{\mathrm{CPR}}$ or to $\mathbf{P}$.

Proof. Reconstruction safety is Proposition 7.1. At every internal stage, Proposition 7.2 shows that the singleton $\{\rho_{i-1}\}$, containing the most recently processed stage record, is task-sufficient, while Proposition 7.3 shows that the empty interface is not task-sufficient because both incoming carry values occur among reachable prefixes and induce distinct carry signatures. This proves the exact internal interface size for $n \ge 2$. When $n = 1$, there is no internal stage $1 \le i \le n-1$, and both boundary stages have empty minimum interface; hence the unnormalized width is zero in that case. The stated values of $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}$ and $\boldsymbol{\omega}_{\mathrm{rec}}$ follow.

The carry signature takes values in the fixed alphabet $\mathcal{Q}_{\mathrm{carry}} = \{0, 1\}$. Equations 7.4 and 7.5 therefore give $V_{\mathrm{sem}} \le 2(n+1)$ and $E_{\mathrm{sem}} \le 8n$. Proposition 7.4 establishes uniform linear construction of this semantic structure.

The four boundary conditions are verified explicitly in Section 7.6: BC1 follows from the uniform component, ordering, reachability, signature, transition, and output descriptions; BC2 follows from linear construction of the stage descriptors and constant local transition data; BC3 follows from the local identity $a_i + b_i + c_i = s_i + 2c_{i+1}$, which characterizes exactly the labeled ripple-adder transitions; and BC4 follows from the linear semantic vertex and edge bounds above.

Finally, each stage requires only a fixed amount of work for every declared local construction, reduction, transition, reconstruction, verification, and output action. Therefore fixed constants C_j exist with $N_j \le C_j(n+1)$ for all six runtime components. Their sum is $O(n)$. Conversely, the declared schema output contains one constant-size transition record for each of the n stages, so $N_{\mathrm{Output}} = \Omega(n)$, and hence $N_{\mathrm{CPR}} = \Omega(n)$. Combining the upper and lower bounds gives $N_{\mathrm{CPR}} = \Theta(n)$. □

Table 7.2: Complete CPR certificate for the uniformly encoded ripple-adder schema family. All statements refer to the same task τ_{sch}, model $\mathfrak{M}_{\mathrm{sch}}$, and ordering π_{add}.

Certificate item	Established bound or property
Reconstruction safety	Carry signature satisfies S1–S3 (Proposition 7.1).
Minimum active interface	Internal minimum size 1 for $n \ge 2$; terminal and $n = 1$ boundary cases give the stated exact width.
Operational CPR width	$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}} = 0$ for $n = 1$, and $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}} = 1$ for $n \ge 2$.
Normalized CPR width	$\boldsymbol{\omega}_{\mathrm{rec}} = 0$ for every $n \ge 1$.
Semantic vertices	$V_{\mathrm{sem}} \le 2(n+1)$.
Semantic edges	$E_{\mathrm{sem}} \le 8n$.
BC1	Explicit uniform structural representation.
BC2	Uniform preprocessing and quotient construction in $O(n)$.
BC3	Exact soundness and completeness for the labeled carry-transition schema.
BC4	Polynomial, in fact linear, semantic-state and transition structure.
Six runtime components	Uniform upper bounds $N_j \le C_j(n+1) = O(n)$, all six j.
Output lower bound	$N_{\mathrm{Output}} = \Omega(n)$.
Complete operation count	$N_{\mathrm{CPR}} = \Theta(n)$.

Why this is a family-level result. The preceding theorem is not inferred from a single full-adder truth table or from a fixed value of n. One model description, one task, one ordering rule, one signature family, and one collection of component-wise polynomial bounds apply to every member of the unbounded family

$\mathcal{F}^{\mathrm{sch}}_{\mathrm{add}}$. The result therefore supplies the family-level uniformity that finite examples alone cannot provide. It remains a certificate for the declared schema/transducer task and is not a membership statement in the decision-language class $\mathbf{P}_{\mathrm{CPR}}$.

Proposition 7.5 (Task Separation and Non-Transfer)**.** *The schema task τ_{sch} of this chapter and the fixed-target reconstruction-counting task $\tau_{\mathrm{count},s}$ of Chapter 8 are distinct declared tasks:*

$$\tau_{\mathrm{sch}} \neq \tau_{\mathrm{count},s}.$$

Consequently, the CPR width, semantic quotient, BC1–BC4 certificate, and runtime bounds proved for τ_{sch} do not transfer to $\tau_{\mathrm{count},s}$ without a separate task-relative proof. Conversely, the counting identity and empirical measurements of Chapter 8 do not constitute task-relative evidence for the schema certificate proved here.

Proof. The output object of τ_{sch} is the complete stage-indexed labeled carry-transition relation of the all-input transducer. The output object of $\tau_{\mathrm{count},s}$ is an integer multiplicity: the number of input-pair sequences compatible with one prescribed target sum-bit vector s.
The one-bit carry signature used for τ_{sch} is not, by itself, the complete state required for the counting recurrence of Chapter 8. That recurrence propagates the carry-indexed multiplicity vector

$$M_i = (\mu_i(0), \mu_i(1)),$$

relative to the prescribed target sequence s. Thus the two tasks have different output objects, reconstruction objectives, and task-relative state information.
Since CPR interfaces, reconstruction-safe signatures, semantic equivalence, and the six runtime components are task-relative by Chapters 2–5, no width, quotient, runtime, or evidence claim transfers from one task to the other without a separate proof. □

Concrete finite instantiations of the certificate. The uniform theorem immediately specializes to small and moderate values of n. For $n = 1$, no internal stage $1 \leq i \leq n-1$ exists, so the maximum minimum-interface size is zero and

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}} = 0, \qquad \boldsymbol{\omega}_{\mathrm{rec}} = 0.$$

For $n = 2$, the only internal stage is $i = 1$, where the exact minimum interface is $\{\rho_0\}$; hence $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}} = 1$ and $\boldsymbol{\omega}_{\mathrm{rec}} = 0$. For $n = 5$, the four internal stages have exact minimum interfaces

$$\{\rho_0\}, \quad \{\rho_1\}, \quad \{\rho_2\}, \quad \{\rho_3\},$$

respectively, so again $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}} = 1$ and $\boldsymbol{\omega}_{\mathrm{rec}} = 0$.
For $n = 10$, the complete input domain has cardinality

$$|\mathbb{B}^{10} \times \mathbb{B}^{10} \times \mathbb{B}| = 2^{21},$$

whereas the certified stage-indexed semantic structure satisfies

$$V_{\mathrm{sem}} \leq 22, \qquad E_{\mathrm{sem}} \leq 80,$$

with exact operational interface width $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}} = 1$. This is a comparison of structural quantities, not a measured or asymptotic speedup claim.
For $n = 100$, specializing the linear bounds of Theorem 7.3, items 5–6, gives the concrete numerical bounds

$$N_j(P^{\mathrm{sch}}_{\mathrm{add},100}, \pi_{\mathrm{add}}, \tau_{\mathrm{sch}}; \mathfrak{M}_{\mathrm{sch}}) \leq 101\, C_j, \qquad j \in \{\mathrm{Build}, \mathrm{Reduce}, \mathrm{Transition}, \mathrm{Reconstruct}, \mathrm{Verify}, \mathrm{Output}\},$$

for the fixed positive constants C_j of Theorem 7.3, item 5, while the required stage-wise schema output independently gives the numerical lower bound

$$N_{\mathrm{Output}}(P^{\mathrm{sch}}_{\mathrm{add},100}, \pi_{\mathrm{add}}, \tau_{\mathrm{sch}}; \mathfrak{M}_{\mathrm{sch}}) \geq 100.$$

Landau notation is asymptotic in n and is not meaningful applied to a single fixed instance such as $n = 100$; the two displayed inequalities are instead the concrete numerical specialization, at $n = 100$, of the linear family-level scaling $N_{\mathrm{CPR}} = \Theta(n)$ established for the complete family $\mathcal{F}^{\mathrm{sch}}_{\mathrm{add}}$ in Theorem 7.3, item 6.
As a concrete task-separation example, let $n = 4$, fix the initial carry $c_0 = 0$, and fix the target sum-bit sequence $s = (1, 0, 1, 1)$. The schema task τ_{sch} materializes the complete four-stage labeled carry-transition relation, with at most eight local labeled transitions per stage. The distinct task $\tau_{\mathrm{count},s}$ asks instead for one multiplicity, namely the number of input-pair sequences reproducing this prescribed target sequence under the fixed initial carry $c_0 = 0$ and unrestricted final carry c_4, exactly as required by the hypotheses of Proposition 8.1. Proposition 8.1 gives

$$N_{\mathrm{count}}(s) = 2^4 = 16.$$

The first task outputs a transition relation; the second outputs one integer and propagates $M_i = (\mu_i(0), \mu_i(1))$ relative to s. The two records therefore illustrate why the task-relative CPR certificate of this chapter and the counting evidence of Chapter 8 must remain separate.

Structural contrast with the global input space. For fixed n, the complete input domain of the ripple-adder function contains

$$|\mathbb{B}^n \times \mathbb{B}^n \times \mathbb{B}| = 2^{2n+1}$$

input triples (A, B, c_0). The schema represents the complete local transition behavior for all admissible input labels without enumerating this global set of 2^{2n+1} complete input triples. Its internal future-active interface has cardinality one for every nonterminal internal stage when $n \geq 2$, and its semantic carry alphabet has cardinality two. Thus the global input-domain cardinality grows exponentially with n, while the exact operational CPR interface width remains constant.

This comparison is structural, not an algorithmic speedup statement. Standard ripple-carry addition already admits linear sequential evaluation, and the present theorem does not claim an exponential improvement over that algorithm. The contribution is instead a complete resource-separation certificate showing, within one explicit uniform family, that global input-domain size, active-interface width, stage-indexed semantic structure, and complete operation count are mathematically distinct quantities.

7.7 Structural Interpretation, Scope, and Conclusion

The isolated full adder realizes the exact finite factorization

$$\mathbb{B}^3 \xrightarrow{C_{\mathrm{FA}}} \{0,1,2,3\} \xrightarrow{Q_{\mathrm{FA}}} \mathbb{B}^2,$$

with $S_{\mathrm{out}}(P_{\mathrm{FA}}) = 4$.

For the all-input ripple-adder schema,

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P^{\mathrm{sch}}_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{sch}}; \mathfrak{M}_{\mathrm{sch}}) = \begin{cases} 0, & n = 1, \\ 1, & n \geq 2, \end{cases}$$

and

$$\boldsymbol{\omega}_{\mathrm{rec}}(P^{\mathrm{sch}}_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{sch}}; \mathfrak{M}_{\mathrm{sch}}) = 0.$$

The same declared schema has

$$N_{\mathrm{CPR}} = \Theta(n)$$

for materializing and verifying its stage-indexed transition relation.

A fixed encoded addition input does not receive a separate CPR-width claim in this chapter. Instead, it selects one deterministic path through the already declared schema, and that path evaluates the exact sum in

$$N_{\mathrm{eval}}(n, x) = \Theta(n).$$

7.7.1 Ripple Addition as a Resource-Separation Example

Ripple addition demonstrates that a bounded active interface does not imply constant total cost. For $n \geq 2$, the schema retains one future-active carry component at every internal stage, while the stage depth grows as

$$m_{\mathrm{stage}} = n.$$

The complete schema relation contains only a constant number of labeled transitions per stage, yet materializing all n stages requires linear output and therefore

$$N_{\mathrm{CPR}}(P^{\mathrm{sch}}_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{sch}}; \mathfrak{M}_{\mathrm{sch}}) = \Theta(n). \tag{7.9}$$

The same carry state is also the stage-local information used by any fixed-input forward evaluation path. Thus the carry is not hidden outside the structural interface merely to obtain a smaller width.

This is a diagnostic use of CPR-WIDTH, not an algorithmic novelty claim. The standard ripple adder is already known to have linear sequential evaluation cost. The present analysis identifies the one-component carry interface of the declared all-input transducer and separately records the linear cost of evaluating one selected input path.

The results do not imply analogous width, quotient, or runtime properties for arbitrary Boolean or arithmetic circuits.

Chapter 8 records the exhaustive finite verification of the isolated full-adder factorization and its output-equivalence fibers, and then reports a distinct fixed-target reconstruction-counting experiment under its own counting task.

8 Finite Verification Records

Chapter 7 established the full-adder factorization and a uniform all-input ripple-adder carry-transition schema. The empirical ripple-adder experiment below uses a different fixed-target counting task and is not treated as a measurement of the schema task.

This chapter records three distinct forms of evidence:

- exhaustive finite verification of the isolated full adder;
- implementation-specific reconstruction measurements for the ripple-adder family;
- and exploratory structural-interface measurements on random 3-SAT instances.

Their evidential levels remain separate throughout.

For self-contained use in this chapter, the evidence labels used below are fixed before their first occurrence. The classification is:

E0 – Definition Evidence.
Definition or formal setup only; no verification claim.
E1 – Finite Verification Evidence.
Verification on an explicitly stated finite record.
E2 – Structural Pattern Evidence.
Structural evidence on sampled or constructed instances.
E3 – Operation-Count Evidence.
A finite or symbolic elementary-operation comparison under a declared accounting convention.
E4 – Asymptotic Theorem Evidence.
A theorem proved uniformly under explicit mathematical assumptions.
E5 – Measured Computational Evidence.
A runtime or memory claim is supported by a specified implementation, hardware description, software environment, benchmark protocol, and reproducible measurements.

These labels classify the support attached to a claim.
They do not form an implication chain.
In particular, finite or measured evidence does not by itself imply an E4 asymptotic theorem.
Chapter 9 gives the manuscript-wide evidence policy and interpretation.
The isolated full-adder verification does not construct a stage-wise CPR model for the full-adder block.
In particular, it does not define

- a reconstruction-component ordering;
- a sequence of reachable prefix-state spaces;
- a sequence of minimum task-sufficient interfaces;
- or stage-dependent CPR semantic quotient spaces.

Its verified finite relation is

$$\mathbb{B}^3 \xrightarrow{C_{\mathrm{FA}}} Z_{\mathrm{FA}} \xrightarrow{Q_{\mathrm{FA}}} \mathbb{B}^2, \tag{8.1}$$

where C_{FA} forms Hamming-weight classes and Q_{FA} reconstructs the exact full-adder output pair.

8.1 Purpose and Finite Input/Output Spaces

The finite record checks the symbolic full-adder factorization of Chapter 7 against all eight inputs and distinguishes

- input states;
- Hamming-weight states;
- projection fibers;
- output-equivalence classes;
- and output states.

It is an exact finite verification, not an asymptotic tractability or operation-count result.
Let

$$\mathbb{B} = \{0, 1\}. \tag{8.2}$$

The complete local input space is

$$X_{\mathrm{FA}} = \mathbb{B}^3, \tag{8.3}$$

so

$$|X_{\mathrm{FA}}| = 2^3 = 8. \tag{8.4}$$

Every input state has the form

$$x = (a, b, c_{\mathrm{in}}), \qquad a, b, c_{\mathrm{in}} \in \mathbb{B}. \tag{8.5}$$

The full-adder output map is

$$F_{\mathrm{FA}} : X_{\mathrm{FA}} \longrightarrow \mathbb{B}^2, \tag{8.6}$$

with

$$F_{\mathrm{FA}}(x) = (s, c_{\mathrm{out}}), \tag{8.7}$$

where s is the sum bit and c_{out} the outgoing carry bit.

8.2 Complete Finite Verification Table

Define the Hamming weight of an input state by

$$h = a + b + c_{\text{in}}. \tag{8.8}$$

The complete finite input space and the corresponding outputs are shown in Table 8.1.

Table 8.1: Complete finite verification table for the full-adder Hamming-weight factorization.

a	b	c_{in}	h	s	c_{out}	$Q_{\text{FA}}(h)$
0	0	0	0	0	0	$(0,0)$
0	0	1	1	1	0	$(1,0)$
0	1	0	1	1	0	$(1,0)$
0	1	1	2	0	1	$(0,1)$
1	0	0	1	1	0	$(1,0)$
1	0	1	2	0	1	$(0,1)$
1	1	0	2	0	1	$(0,1)$
1	1	1	3	1	1	$(1,1)$

For every row, the arithmetic identity

$$a + b + c_{\text{in}} = s + 2c_{\text{out}} \tag{8.9}$$

holds.
The last three columns also show directly that

$$F_{\text{FA}}(x) = Q_{\text{FA}}(h) \tag{8.10}$$

for every one of the eight possible input states.

8.3 Hamming-Weight Projection and Fibers

Define the Hamming-weight projection

$$C_{\text{FA}} : \mathbb{B}^3 \longrightarrow \{0,1,2,3\} \tag{8.11}$$

by

$$C_{\text{FA}}(a,b,c_{\text{in}}) = a + b + c_{\text{in}}. \tag{8.12}$$

The reduced state space is

$$Z_{\text{FA}} = \{0,1,2,3\}, \tag{8.13}$$

with

$$|Z_{\text{FA}}| = 4. \tag{8.14}$$

Thus, the finite projection cardinality pattern is

$$8 \longrightarrow 4. \tag{8.15}$$

Define the output-reconstruction map

$$Q_{\text{FA}} : Z_{\text{FA}} \longrightarrow \mathbb{B}^2 \tag{8.16}$$

by

$$Q_{\text{FA}}(h) = \left(h \bmod 2, \left\lfloor \frac{h}{2} \right\rfloor\right). \tag{8.17}$$

The row-wise verification of Table 8.1 yields

$$F_{\text{FA}} = Q_{\text{FA}} \circ C_{\text{FA}}. \tag{8.18}$$

The map C_{FA} is not injective.
Therefore, the original input state cannot generally be reconstructed uniquely from its Hamming weight.
The complete output pair, however, is reconstructed uniquely by Q_{FA}.
For every

$$h \in Z_{\text{FA}},$$

define the projection fiber

$$\Omega_{\text{FA}}(h) = C_{\text{FA}}^{-1}(\{h\}). \tag{8.19}$$

The four fibers are

$$\Omega_{\mathrm{FA}}(0) = \{(0,0,0)\}, \tag{8.20}$$

$$\Omega_{\mathrm{FA}}(1) = \{(1,0,0),(0,1,0),(0,0,1)\}, \tag{8.21}$$

$$\Omega_{\mathrm{FA}}(2) = \{(1,1,0),(1,0,1),(0,1,1)\}, \tag{8.22}$$

and

$$\Omega_{\mathrm{FA}}(3) = \{(1,1,1)\}. \tag{8.23}$$

Their cardinalities are therefore

$$(|\Omega_{\mathrm{FA}}(0)|, |\Omega_{\mathrm{FA}}(1)|, |\Omega_{\mathrm{FA}}(2)|, |\Omega_{\mathrm{FA}}(3)|) = (1,3,3,1). \tag{8.24}$$

The fibers form a partition of the complete input space:

$$\mathbb{B}^3 = \bigsqcup_{h \in Z_{\mathrm{FA}}} \Omega_{\mathrm{FA}}(h), \tag{8.25}$$

and

$$\sum_{h \in Z_{\mathrm{FA}}} |\Omega_{\mathrm{FA}}(h)| = 1 + 3 + 3 + 1 = 8. \tag{8.26}$$

The maximum projection-fiber size is

$$d_{\mathrm{fiber}}^{\max}(P_{\mathrm{FA}}) = \max_{h \in Z_{\mathrm{FA}}} |\Omega_{\mathrm{FA}}(h)| = 3. \tag{8.27}$$

This is a finite fiber-cardinality statement.
It is not a CPR-Width value, a stage-wise semantic-state count, or a runtime measure.

8.4 Exact Output and Output-Equivalence Verification

Theorem 8.1 (Complete Finite Full-Adder Factorization Verification)**.** *For every input state*

$$x \in \mathbb{B}^3,$$

the reduced Hamming-weight state

$$h = C_{\mathrm{FA}}(x)$$

uniquely determines the complete full-adder output

$$F_{\mathrm{FA}}(x) = (s, c_{\mathrm{out}}).$$

Consequently,

$$F_{\mathrm{FA}} = Q_{\mathrm{FA}} \circ C_{\mathrm{FA}} \tag{8.28}$$

on the complete finite input space.

Proof. Let

$$h = a + b + c_{\mathrm{in}}.$$

Since

$$h \in \{0,1,2,3\},$$

its parity determines the sum bit,

$$s = h \bmod 2,$$

and its integer quotient by two determines the outgoing carry,

$$c_{\mathrm{out}} = \left\lfloor \frac{h}{2} \right\rfloor.$$

Thus,

$$F_{\mathrm{FA}}(x) = \left(h \bmod 2, \left\lfloor \frac{h}{2} \right\rfloor\right) = Q_{\mathrm{FA}}(h) = Q_{\mathrm{FA}}\left(C_{\mathrm{FA}}(x)\right).$$

Table 8.1 lists all eight elements of $\mathbb{B}^3$ and confirms the equality for every row.
Therefore, the factorization holds on the complete finite input space.

□

For the isolated full-adder output task, define

$$x \equiv_{\mathrm{out}} y \quad \Longleftrightarrow \quad F_{\mathrm{FA}}(x) = F_{\mathrm{FA}}(y). \tag{8.29}$$

This relation compares complete input states according to their output behavior.
It is not the general stage-wise CPR semantic equivalence of Chapter 3.

Lemma 8.1 (Equality of Output and Hamming-Weight Classes)**.** *For all*

$$x, y \in \mathbb{B}^3,$$

one has

$$F_{\mathrm{FA}}(x) = F_{\mathrm{FA}}(y) \quad \Longleftrightarrow \quad C_{\mathrm{FA}}(x) = C_{\mathrm{FA}}(y). \tag{8.30}$$

Proof. If

$$C_{\mathrm{FA}}(x) = C_{\mathrm{FA}}(y),$$

then the factorization

$$F_{\mathrm{FA}} = Q_{\mathrm{FA}} \circ C_{\mathrm{FA}}$$

gives

$$F_{\mathrm{FA}}(x) = F_{\mathrm{FA}}(y).$$

Conversely,

$$Q_{\mathrm{FA}}(0) = (0,0), \qquad Q_{\mathrm{FA}}(1) = (1,0),$$

$$Q_{\mathrm{FA}}(2) = (0,1), \qquad Q_{\mathrm{FA}}(3) = (1,1).$$

These four values are pairwise distinct.
Therefore,

$$Q_{\mathrm{FA}} : Z_{\mathrm{FA}} \longrightarrow \mathbb{B}^2$$

is bijective and, in particular, injective.
Hence,

$$F_{\mathrm{FA}}(x) = F_{\mathrm{FA}}(y)$$

implies

$$C_{\mathrm{FA}}(x) = C_{\mathrm{FA}}(y).$$

□

Define the isolated full-adder output quotient by

$$\mathcal{S}_{\mathrm{FA}}^{\mathrm{out}} = \mathbb{B}^3 / \equiv_{\mathrm{out}}. \tag{8.31}$$

Its classes are precisely the Hamming-weight fibers:

$$\mathcal{S}_{\mathrm{FA}}^{\mathrm{out}} = \{\Omega_{\mathrm{FA}}(0), \Omega_{\mathrm{FA}}(1), \Omega_{\mathrm{FA}}(2), \Omega_{\mathrm{FA}}(3)\}. \tag{8.32}$$

Define

$$S_{\mathrm{out}}(P_{\mathrm{FA}}) = \left|\mathcal{S}_{\mathrm{FA}}^{\mathrm{out}}\right|. \tag{8.33}$$

Therefore,

$$S_{\mathrm{out}}(P_{\mathrm{FA}}) = 4. \tag{8.34}$$

The binary output-quotient dimension is

$$D_{\mathrm{out}}(P_{\mathrm{FA}}) = \lceil \log_2 \max\{1, S_{\mathrm{out}}(P_{\mathrm{FA}})\} \rceil = 2. \tag{8.35}$$

Thus,

$$S_{\mathrm{out}}(P_{\mathrm{FA}}) = 4$$

counts the output-equivalence classes, while

$$D_{\mathrm{out}}(P_{\mathrm{FA}}) = 2$$

counts the bits required to identify one of those four classes.
Neither quantity is asserted here to equal a stage-wise quantity

$$S_{\mathrm{CPR}}(P_{\mathrm{FA}}, \pi, \tau; \mathfrak{M})$$

or

$$D_{\mathrm{CPR}}(P_{\mathrm{FA}}, \pi, \tau; \mathfrak{M}),$$

because no staged CPR reconstruction model for the isolated full-adder block is constructed in this chapter.

8.5 Verification Record and Evidence Status

The exhaustive finite verification establishes that

- the complete input space contains exactly eight states;
- the Hamming-weight projection produces four reduced states;
- the projection fibers have cardinalities $(1, 3, 3, 1)$;
- the four fibers are pairwise disjoint and cover $\mathbb{B}^3$;
- the maximum projection-fiber size is 3;
- the full-adder output map factors through the Hamming weight;
- Q_{FA} is bijective on Z_{FA};
- two input states have the same output exactly when they have the same Hamming weight;
- and the isolated output quotient contains exactly four classes.

This is a finite verification record.
It establishes the stated properties for the complete isolated full-adder input space, but it does not establish an asymptotic theorem for an unrestricted circuit family.
The present full-adder record has finite-verification status:

Evidence Status: E1 – Finite Verification Evidence

The record qualifies as E1 because all eight input states are enumerated, all four Hamming-weight values are listed, every projection fiber is computed exactly, the factorization is checked on the complete finite input space, and the output-equivalence quotient is determined exactly.
The evidence level supports only the corresponding finite conclusions.
In particular,

$$\mathrm{E1} \not\Longrightarrow \mathrm{E4}. \tag{8.36}$$

Likewise,

$$8 \longrightarrow 4 \tag{8.37}$$

is a finite state-cardinality relation and does not by itself imply an operation-count advantage or a measured wall-clock advantage.

8.6 Empirical Illustration: Reconstruction Counting and Interface Growth

The preceding sections verify the full-adder factorization exhaustively but do not measure a runtime.
This section reports two small computational experiments: an exact reconstruction-counting task for the ripple-adder family, cross-checked against a brute-force baseline, and an interface-size measurement on random 3-SAT instances.
Both experiments are illustrative rather than a benchmark suite; neither is compared against a production SAT or arithmetic solver.

Scope and reproducibility, stated once. Every measurement in this section shares the same limitations, so they are stated here rather than repeated after each table.
All reported wall-clock figures are single runs, not averaged over repeated trials, taken in a shared virtualized sandbox rather than dedicated benchmarking hardware. No runtime exponent, asymptotic timing law, or hardware-independent speedup should be inferred from them.

Two items needed for full reproducibility were not retained in the archived record: the exact target sum-bit sequences used for the Section 8.6.2 timing runs, and the random-number-generator seed together with a complete generator specification for the Section 8.6.3 formulas.
Neither gap affects the formal mathematical results proved below: the counting recurrence is justified independently by Lemma 8.2 and Proposition 8.1. The missing records do, however, prevent independent regeneration of the archived timing and random-instance measurements from the manuscript text alone.
A future reproduction package should retain the target sequence, seed, generator specification, full source code, hardware and software environment, repetition protocol, and raw per-instance record for every reported figure.
Under the E0–E5 evidence classification introduced at the start of this chapter, the finite implementation cross-checks provide finite verification evidence, the timing figures are preliminary measured implementation evidence, and the 3-SAT interface measurements are E2 structural-pattern evidence.
None of these empirical records constitutes an E4 asymptotic theorem or a validated E5 benchmark. Proposition 8.1, by contrast, is a formal mathematical result proved independently of those measurements.

8.6.1 Implementation, Environment, and Protocol

All code was written in Python using CPython 3.10.12.
The brute-force adder enumeration additionally uses vectorized NumPy 2.2.6 array operations to make exact cross-checking feasible up to

$$n = 12.$$

All code was executed single-threaded in a sandboxed Ubuntu 22.04 Linux container on aarch64 hardware.
Wall-clock time was measured with `time.perf_counter()` around the reconstruction routine only, excluding random-instance generation.
Peak memory was additionally traced during the runs with Python's `tracemalloc` module. These traced values reflect Python-level allocations rather than full process resident-set size and are not part of the reported numerical record; only the wall-clock timings below are reported as measured implementation evidence.
The three scripts used to produce the reported numerical record are

- `cpr_reconstruction.py`;
- `bruteforce_numpy.py`;
- `sat_interface_growth.py`.

They are intended to accompany this manuscript in a `code/` directory of the supplementary submission.
The exact contents of the submitted supplement should be verified against this list immediately before final submission; this manuscript text alone does not establish that the corresponding files have been uploaded.

8.6.2 Ripple-Adder Reconstruction Counting

The reconstruction-counting experiment in this subsection uses a different declared task from the schema task τ_{sch} of Chapter 7. For a fixed encoded addition input, forward evaluation selects one deterministic path through that schema; the counting experiment below instead asks for the multiplicity of input-pair sequences compatible with a prescribed target sum-bit sequence.
For a fixed target sum-bit sequence

$$s = (s_0, s_1, \ldots, s_{n-1}) \in \mathbb{B}^n \tag{8.38}$$

and fixed initial carry

$$c_0 = 0, \tag{8.39}$$

define the counting task

$$\tau_{\mathrm{count},s} = \text{count all input-pair sequences } \big((a_i, b_i)\big)_{i=0}^{n-1} \text{ that reproduce } s. \tag{8.40}$$

Thus, the object to be reconstructed is not the unique forward output of a fixed pair (A, B), but the multiplicity of all admissible input prefixes compatible with the prescribed target bits.
Accordingly, no CPR-Width or task-sufficient signature statement from the schema task τ_{sch}, nor from fixed-input forward evaluation through that schema, is transferred to $\tau_{\mathrm{count},s}$ without a separate task-relative argument.
For stage i, let

$$\mu_i(c) \tag{8.41}$$

denote the number of admissible partial input-pair sequences

$$(a_0, b_0), \ldots, (a_{i-1}, b_{i-1})$$

that reproduce

$$s_0, \ldots, s_{i-1}$$

and end with carry

$$c_i = c.$$

Define the carry-multiplicity state by

$$M_i = \big(\mu_i(0), \mu_i(1)\big). \tag{8.42}$$

The initial state is

$$M_0 = (1, 0), \tag{8.43}$$

because $c_0 = 0$ is fixed.
For a prescribed target bit s_i, define the constant local multiplicity coefficient

$$m_{s_i}(c, c') = \left|\left\{(a, b) \in \mathbb{B}^2 : a + b + c = s_i + 2c'\right\}\right|. \tag{8.44}$$

The next multiplicity state is therefore computed by

$$\mu_{i+1}(c') = \sum_{c \in \mathbb{B}} \mu_i(c)\, m_{s_i}(c, c'). \tag{8.45}$$

Only the two carry indices

$$c \in \{0, 1\}$$

are required, although their integer multiplicities may grow with i.
Lemma 8.2 (Task-Relative Carry-Multiplicity Sufficiency)**.** *For the declared counting task $\tau_{\mathrm{count},s}$, the state*

$$M_i = \big(\mu_i(0), \mu_i(1)\big)$$

is sufficient to determine the aggregate number of admissible continuations represented by each carry class at stage i, compatible with

$$s_0, \ldots, s_{i-1}.$$

Moreover, the update from M_i to M_{i+1} depends only on M_i and the next fixed target bit s_i.

Proof. Every admissible continuation from stage i depends on the processed prefix only through its current carry c_i.
The local ripple-adder constraint at the next position is

$$a_i + b_i + c_i = s_i + 2c_{i+1}. \tag{8.46}$$

Hence all processed input prefixes ending in the same carry value have the same set of admissible next input-pair types and the same continuation structure for the unresolved target suffix.
The number of such processed prefixes is exactly $\mu_i(c)$.
Therefore, the two multiplicities

$$\mu_i(0) \qquad \text{and} \qquad \mu_i(1)$$

contain all information needed to propagate the exact counting recurrence.
Equation (8.45) computes each successor multiplicity from these two values and the fixed target bit s_i.
No earlier input prefix must be retained individually.
Thus, M_i is sufficient for the declared counting task and admits a constant-dimension stage-local update. □

The lemma establishes a two-coordinate dynamic counting state for $\tau_{\mathrm{count},s}$.
This is a task-specific result.
It is not an inference from the normalized forward-evaluation width

$$\boldsymbol{\omega}_{\mathrm{rec}}(P^{\mathrm{sch}}_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{sch}}; \mathfrak{M}_{\mathrm{sch}}) = 0 \tag{8.47}$$

of Chapter 7. That width belongs to the all-input schema task τ_{sch}, not to the counting task $\tau_{\mathrm{count},s}$.
In particular, the phrase "two semantic states" in the empirical record refers to the two carry-indexed multiplicity channels

$$c \in \{0,1\},$$

not to a claim that the complete multiplicity vector M_i has only two possible numerical values.
Its coordinates are unbounded nonnegative integers whose bit lengths must be included in a strict bit-cost analysis.

CPR counting procedure. At each stage, the procedure maintains only the two carry-indexed multiplicities.
For each current carry value, it examines the four local input pairs

$$(a_i, b_i) \in \mathbb{B}^2$$

and accumulates their contributions into the two successor carry channels whenever

$$a_i + b_i + c_i = s_i + 2c_{i+1}.$$

The local transition pattern has constant size.
The experiment therefore avoids enumerating all

$$4^n = 2^{2n} \tag{8.48}$$

input-pair sequences $(a_0, b_0), \ldots, (a_{n-1}, b_{n-1}) \in \mathbb{B}^2$, while the initial carry $c_0 = 0$ stays fixed rather than enumerated.
The exact final reconstruction count is

$$N_{\text{count}}(s) = \mu_n(0) + \mu_n(1), \tag{8.49}$$

because the experiment constrains the n sum bits but does not fix a prescribed final carry.

Brute-force baseline. The baseline enumerates all

$$2^{2n}$$

raw input sequences directly and checks whether each ripple-carry addition reproduces the prescribed sum-bit sequence s.
No carry-multiplicity compression is used.

Correctness cross-check. For every tested

$$n \in \{4, 6, 8, 10, 12\}, \tag{8.50}$$

the counting procedure and brute-force enumeration produced identical counts:

$$16,\ 64,\ 256,\ 1024,\ 4096, \tag{8.51}$$

respectively, i.e.

$$2^n \tag{8.52}$$

for the particular target sequences used in the experiment.

Proposition 8.1 (General Reconstruction-Count Identity)**.** *For every $n \geq 1$, every target sum-bit sequence $s \in \mathbb{B}^n$, fixed initial carry $c_0 = 0$, and unrestricted final carry c_n,*

$$N_{\text{count}}(s) = 2^n.$$

Proof. Fix any stage i with incoming carry $c_i \in \{0,1\}$ and target bit $s_i \in \{0,1\}$. Among the four local input pairs $(a_i, b_i) \in \mathbb{B}^2$, the value $a_i + b_i$ takes value 0 once, value 1 twice, and value 2 once. Exactly those pairs satisfying $a_i + b_i + c_i \equiv s_i \pmod 2$ are admissible at this stage; checking all four cases of (c_i, s_i) shows this condition holds for exactly two of the four pairs, regardless of the values of c_i and s_i. Hence exactly two admissible local input pairs exist at every stage, independently of the target bit and the incoming carry. Consequently the total prefix count $\mu_i(0) + \mu_i(1)$ exactly doubles at every stage,

$$\mu_{i+1}(0) + \mu_{i+1}(1) = 2\big(\mu_i(0) + \mu_i(1)\big),$$

starting from $\mu_0(0) + \mu_0(1) = 1$ for the fixed initial carry $c_0 = 0$. By induction, $\mu_n(0) + \mu_n(1) = 2^n$, and since $N_{\text{count}}(s) = \mu_n(0) + \mu_n(1)$, the claim follows. □

The five values reported above therefore illustrate a general identity holding for every target sequence s, rather than a property specific to the five tested instances. That generality is supplied by Proposition 8.1

and does not depend on the missing target-sequence record discussed in the reproducibility note at the start of Section 8.6.
The exact numerical agreement between the carry-multiplicity procedure and the brute-force baseline for

$$n \in \{4, 6, 8, 10, 12\}$$

provides finite implementation cross-check evidence for those tested sizes.
Because the exact target sequences used in the archived runs were not retained, these historical cross-checks are not presented as a fully reproducible finite benchmark record. Their mathematical target values, however, are independently fixed by Proposition 8.1, which proves

$$N_{\mathrm{count}}(s) = 2^n$$

for every $s \in \mathbb{B}^n$.
The wall-clock figures in Table 8.2 are preliminary measured implementation evidence. Because the archived record does not retain the exact target sequences and does not provide a repeated-trial measurement protocol, those timings remain below the manuscript's strict E5 threshold.
The "Active carry channels" column records the number of carry-indexed channels maintained by the recurrence, which is always two. It does not represent the number of possible numerical multiplicity vectors, and it does not constitute a task-relative CPR-Width measurement.
The exact brute-force cross-check was performed at all five sizes $n \in \{4, 6, 8, 10, 12\}$. Table 8.2 reports only a selected subset of those cross-checked sizes together with additional larger timing-only runs; the omission of $n = 6$ and $n = 10$ from the table does not mean that those two cross-checks were not performed.
The reported timings include the actual arbitrary-precision integer arithmetic performed by CPython. They are empirical implementation measurements and must not be interpreted as a proof of constant bit-cost per recurrence update; the encoded multiplicities grow in bit length with n.

Table 8.2: Ripple-adder reconstruction counting: carry-multiplicity reconstruction versus full raw enumeration. Brute-force beyond $n = 12$ was not run; 2^{2n} is shown analytically to indicate the raw enumeration size.

n	**CPR counting time (s)**	**Active carry channels**	**Brute-force 2^{2n}**
4	2.5×10^{-5}	2	256
8	4.0×10^{-5}	2	65,536
12	6.9×10^{-5}	2	16,777,216
20	1.3×10^{-4}	2	1.1×10^{12}
1,000	5.7×10^{-3}	2	$\approx 10^{602}$
100,000	9.7×10^{-1}	2	$\approx 10^{60,206}$

The timing values are retained as the original single-run measurement record described in Section 8.6.1.
They therefore provide preliminary measured implementation evidence only.
They are not a hardware-independent asymptotic benchmark or a production-solver comparison.

8.6.3 Interface Growth on Random 3-SAT

For n Boolean variables,

$$m = \lfloor 4.2n \rfloor \tag{8.53}$$

random 3-clauses were generated, placing the sampled clause density near the classical empirical random-3-SAT satisfiability transition region [6].
The structural candidate interface $\widetilde{B}_i^{\pi,\tau}$ of Chapter 6—assigned variables that still occur in a clause containing an unassigned variable—was tracked stage by stage under three orderings:

- the natural variable order;
- a random order;
- and a greedy order that locally minimizes the resulting boundary at each step.

The greedy procedure is a heuristic.
It is not claimed to produce an optimal reconstruction ordering.
Table 8.3 reports the mean, over eight random instances at

$$n = 14,$$

of the maximum tracked interface size under each ordering.
The retained experimental record does not contain a pinned random-number-generator seed or a sufficiently detailed generator specification to reconstruct the exact eight sampled formulas independently.
Therefore, Table 8.3 reports the archived aggregate means only.
With eight instances and no retained per-instance values, no standard deviation, min–max range, confidence interval, or inferential statistical claim is reported.

Table 8.3: Mean maximum structural interface size over 8 random 3-SAT instances, $n = 14$ variables, $m = 58$ clauses.

Ordering	Mean maximum interface size
Natural	11.75
Random	11.62
Greedy (local)	11.50

Unlike the ripple-adder and the C_5 graph-coloring instance of Section 6.3.2, none of the three tested orderings produces a small structural candidate interface on these sampled random 3-SAT instances.
The mean maximum tracked interface remains above 82% of n under every tested ordering, including the local greedy heuristic.
This is a negative structural observation: the tested candidate boundary does not expose a small active boundary on these sampled instances.
Because $\widetilde{B}_i^{\pi,\tau}$ is not a proved cardinality-minimal task-sufficient interface, the experiment does not establish that

$$b_i^{\pi,\tau}(P) = \left|\widetilde{B}_i^{\pi,\tau}\right|. \tag{8.54}$$

Consequently, it also does not establish an exact value of

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}$$

or

$$\boldsymbol{\omega}_{\mathrm{rec}}$$

for the sampled formulas.
Likewise, the tested orderings are not proved optimal.
The experiment therefore neither establishes that the exact CPR interface is equally large nor excludes better admissible orderings.
Its status is diagnostic E2 structural-pattern evidence, not E4 asymptotic theorem evidence or a complexity-theoretic claim.
The missing pinned seed, full generator specification, and per-instance dispersion statistics further restrict this record to an exploratory illustration.
A future reproduction package should retain those items explicitly.

8.7 Scope, Evidence Status, Non-Claims, and Conclusion

The three records developed in this chapter have distinct evidential status and must therefore be interpreted at different levels.
The isolated full-adder record exhaustively verifies

$$\mathbb{B}^3 \xrightarrow{C_{\mathrm{FA}}} \{0,1,2,3\} \xrightarrow{Q_{\mathrm{FA}}} \mathbb{B}^2 \tag{8.55}$$

with fiber-cardinality vector

$$(1,3,3,1), \tag{8.56}$$

together with

$$Q_{\mathrm{FA}}(h) = \left(h \bmod 2, \left\lfloor \frac{h}{2} \right\rfloor\right) \tag{8.57}$$

and

$$F_{\mathrm{FA}} = Q_{\mathrm{FA}} \circ C_{\mathrm{FA}}. \tag{8.58}$$

Because Q_{FA} is injective, the Hamming-weight fibers are also the output-equivalence classes.
Therefore,

$$S_{\mathrm{out}}(P_{\mathrm{FA}}) = 4 \tag{8.59}$$

and

$$D_{\mathrm{out}}(P_{\mathrm{FA}}) = 2. \tag{8.60}$$

These are exact finite output-quotient quantities. They are not stage-wise CPR semantic-state quantities.
The isolated full-adder record has E1 finite-verification status because the complete eight-element input

space is explicitly enumerated and verified. It establishes the stated properties on the complete isolated full-adder input space, but it does not establish an asymptotic theorem for an unrestricted circuit family.
The ripple-adder reconstruction-counting experiment has a different evidential status. It provides implementation-specific wall-clock observations over the tested scaling range, supplemented by finite implementation cross-checks against direct raw enumeration for

$$n \in \{4, 6, 8, 10, 12\}.$$

Peak memory was traced during these runs as described in Section 8.6.1, but those traced values are not part of the reported numerical record. CPR-only timing measurements were obtained up to

$$n = 100{,}000.$$

The timing values are single-run measurements obtained in the declared shared virtualized sandbox environment. They therefore constitute preliminary measured implementation evidence for the specified implementation and measurement conditions.
Because the exact target sum-bit sequences used in the archived timing runs were not retained and no repeated-trial protocol was executed, these measurements do not satisfy the manuscript's strict reproducibility requirement for E5 evidence. They remain below the E5 threshold and are not used as evidence for an asymptotic runtime theorem, a hardware-independent speedup, or the schema task τ_{sch}.
The exact counting recurrence is instead justified mathematically by Lemma 8.2 and Proposition 8.1. In particular,

$$N_{\mathrm{count}}(s) = 2^n \qquad \text{for every } s \in \mathbb{B}^n, \tag{8.61}$$

under the declared assumptions of fixed initial carry $c_0 = 0$ and unrestricted final carry c_n.
The five archived brute-force agreements provide finite implementation cross-check evidence for the tested sizes. Because the exact target sequences used in those historical runs were not retained, those runs are not presented as a fully reproducible finite benchmark record. Their expected counting values are nevertheless determined independently by Proposition 8.1.
The reported two-channel carry structure concerns the two carry-indexed multiplicity channels of the counting task $\tau_{\mathrm{count},s}$. It does not constitute a CPR-Width measurement for that task.
The carry-based organization is qualitatively compatible with the carry-state structure used in Chapter 7. However, the empirical reconstruction-counting measurements belong to the distinct task $\tau_{\mathrm{count},s}$, whereas the E4 structural and runtime results of Chapter 7 concern the schema task τ_{sch}.
Consequently, the measurements of Section 8.6.2 do not constitute task-relative evidence for the E4 schema theorem of Chapter 7, and the two evidence records are not combined into one task-relative evidence claim.
The random 3-SAT experiment provides complementary E2 structural-pattern evidence. For

$$n = 14,$$

the reported interface values are means over eight generated instances under the natural, random, and local greedy orderings.
Because the retained record does not contain a pinned random-number seed, a complete generator specification, or the individual per-instance values, no standard deviation, range, confidence interval, or other dispersion statistic is asserted.
Under all three tested orderings, the selected structural candidate interface remains large.
Because

$$\widetilde{B}_i^{\pi,\tau}$$

is not a proved cardinality-minimal task-sufficient interface and the tested orderings are not proved optimal, this record neither establishes that the exact CPR interface is equally large nor excludes better admissible orderings.
In particular, the experiment does not determine an exact value of

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}$$

or

$$\boldsymbol{\omega}_{\mathrm{rec}}$$

for the sampled formulas.
The experiments of Section 8.6 are illustrations rather than a benchmark suite. No production CDCL/DPLL solver, optimized arithmetic tool, or complete G_{count} comparison is used. The brute-force adder enumeration serves only as an exact raw-state baseline.
The supported claims of this chapter are therefore limited to

- the exact finite full-adder factorization and output quotient;

- the finite implementation cross-checks for the tested reconstruction-counting sizes;
- the preliminary measured wall-clock behavior of the specified ripple-adder counting implementation under the stated experimental conditions;
- the general counting identity of Proposition 8.1;
- and the observed structural candidate-interface pattern for the eight sampled random 3-SAT instances.

In particular, these records do not establish

- a hardware-independent speedup;
- a universal CPR advantage;
- a polynomial-time result for SAT;
- a small exact CPR-Width for random 3-SAT;
- tractability of any unrestricted problem class;
- an E5 benchmark result for the archived ripple-adder timings;
- a transfer of the Chapter 7 schema certificate from τ_{sch} to $\tau_{\mathrm{count},s}$;
- or a stage-wise identity $S_{\mathrm{CPR}}(P_{\mathrm{FA}}) = 4$ for the isolated full adder.

Their role is to verify exact finite claims, prove the separately stated counting identity, and illustrate both favorable and unfavorable structural behavior under explicitly stated conditions.

Chapter 9 develops the complexity-theoretic scope, open problems, and future research program of CPR-WIDTH.

9 Complexity-Theoretic Scope, Open Problems, and Future Research

The preceding chapters established Controlled Partial Reconstruction Width as a reconstruction-oriented structural framework for finite combinatorial problems. Throughout this chapter, let P be a finite problem instance, let τ be the declared computational task, let $\mathfrak{M}$ be the fixed CPR reconstruction model, and let

$$\pi \in \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M})$$

be an admissible reconstruction ordering. The central structural-runtime chain of the manuscript is

$$\boldsymbol{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) \longrightarrow S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) \longrightarrow N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}). \tag{9.1}$$

The first quantity is the normalized CPR-Width, the second is the maximum number of semantic quotient states occurring at one reconstruction stage, and the third is the complete CPR operation count. When unambiguous, dependence on τ and $\mathfrak{M}$ may be suppressed. The arrows in Equation (9.1) express structural and accounting dependencies. They do not state that small reconstruction width alone implies polynomial-time tractability. As established in Chapter 4, the complete CPR operation count is

$$\begin{aligned} N_{\mathrm{CPR}} = {} & N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} \\ & + N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}. \end{aligned} \tag{9.2}$$

A favorable value of $\boldsymbol{\omega}_{\mathrm{rec}}$ therefore controls only one structural contribution to the complete analysis. A polynomial-time conclusion requires uniform polynomial control of every runtime component. Let

$$J = \{\mathrm{Build}, \mathrm{Reduce}, \mathrm{Transition}, \mathrm{Reconstruct}, \mathrm{Verify}, \mathrm{Output}\}. \tag{9.3}$$

Table 9.1 instantiates the common width–state–runtime architecture for the principal staged problem-class realizations developed in Chapters 6–8. It complements Table 6.1 rather than replacing it. Table 6.1 records the structural candidate/interface realization and semantic interpretation, whereas Table 9.1 records the corresponding problem-specific instantiation of the common CPR chain. The isolated full-adder block is intentionally not assigned a stage-wise $\boldsymbol{\omega}_{\mathrm{rec}}$ here because Chapter 6 treats it only through the finite output quotient

$$S_{\mathrm{out}}(P_{\mathrm{FA}}) = 4.$$

The staged arithmetic entry therefore uses the ripple-adder family of Chapter 7.

The ripple-adder row refers only to the all-input schema object of Chapter 7. The fixed-target counting experiment of Section 8.6.2 uses the different task $\tau_{\mathrm{count},s}$ and is therefore not a runtime measurement of the table's schema task.

Remark 9.1 (Reading convention for this chapter)**.** Chapter 9 repeatedly compares a structural or empirical CPR quantity against a classical parameter, a stronger theorem, or a general claim, and each time states explicitly what is *not* thereby established. This is a deliberate convention rather than incidental repetition: this chapter is where scope claims are made, so every one-sided bound, conditional generalization, and empirical observation introduced below is paired at the point of introduction with the specific stronger statement it does *not* license – rather than relying on the reader to recall the corresponding caveat from Chapters 2–8, or on one disclaimer stated once and assumed to cover every subsequent instance. The recurring phrasing ("conditional," "one-sided," "does not by itself establish") is therefore intentional and

Table 9.1: Problem-class realizations of the CPR width–state–runtime chain.

Problem class	CPR-Width	Semantic-state quantity	CPR operation count
SAT	$\boldsymbol{\omega}_{\mathrm{rec}}(\Phi, \pi, \tau; \mathfrak{M})$	$S_{\mathrm{CPR}}(\Phi, \pi, \tau; \mathfrak{M})$	$N_{\mathrm{CPR}}(\Phi, \pi, \tau; \mathfrak{M})$
CSP	$\boldsymbol{\omega}_{\mathrm{rec}}(P_{\mathrm{CSP}}, \pi, \tau; \mathfrak{M})$	$S_{\mathrm{CPR}}(P_{\mathrm{CSP}}, \pi, \tau; \mathfrak{M})$	$N_{\mathrm{CPR}}(P_{\mathrm{CSP}}, \pi, \tau; \mathfrak{M})$
Graph coloring	$\boldsymbol{\omega}_{\mathrm{rec}}(G, \pi, \tau; \mathfrak{M})$	$S_{\mathrm{CPR}}(G, \pi, \tau; \mathfrak{M})$	$N_{\mathrm{CPR}}(G, \pi, \tau; \mathfrak{M})$
Tensor	$\boldsymbol{\omega}_{\mathrm{rec}}(T_P, \pi, \tau; \mathfrak{M})$	$S_{\mathrm{CPR}}(T_P, \pi, \tau; \mathfrak{M})$	$N_{\mathrm{CPR}}(T_P, \pi, \tau; \mathfrak{M})$
Ripple-adder schema	$\boldsymbol{\omega}_{\mathrm{rec}}(P^{\mathrm{sch}}_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{sch}}; \mathfrak{M}_{\mathrm{sch}})$	$S_{\mathrm{CPR}}(P^{\mathrm{sch}}_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{sch}}; \mathfrak{M}_{\mathrm{sch}})$	$N_{\mathrm{CPR}}(P^{\mathrm{sch}}_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{sch}}; \mathfrak{M}_{\mathrm{sch}})$

should be read as a scope annotation attached to each individual claim, not as a stylistic lapse.

9.1 Related Work: Classical Structural Width Parameters

Structural width parameters that bound the state space of dynamic-programming procedures have been studied extensively in structural graph theory and parameterized complexity. Treewidth was developed and studied extensively within the graph-minors project [7] and is associated with a large class of dynamic-programming algorithms whose running time is exponential only in the width and polynomial in the instance size [3]. Pathwidth is the linear-ordering restriction of treewidth and coincides with the vertex-separation number of a graph [5]. Branch-width, a further tree-structured width parameter closely related to treewidth, was developed within the same graph-minors line of work [8]. Clique-width generalizes these notions to dense graphs by measuring the complexity of a term-based construction rather than a decomposition tree [2]. Parameterized complexity separates the contribution of such structural parameters from the contribution of the input encoding length to the total running time [4, 3].

These classical parameters share a common algorithmic mechanism: a decomposition (a tree, path, or branch decomposition, or a term construction) fixes, at every node, a bounded set of vertices or a bounded algebraic state that must be retained while the remainder of the graph is processed, and dynamic programming propagates a table indexed by that bounded set [3]. Both the decomposition and the associated dynamic-programming tables are constructed independently of any declared computational task; the same decomposition supports many different task formulations.

CPR-WIDTH addresses a related but distinct question. Rather than fixing a decomposition of the problem representation in advance, it fixes an admissible reconstruction ordering and asks how much task-relevant information carried by already-processed components must remain active to reconstruct the declared task correctly, under a reconstruction-safe signature family specific to that task (Definition 2.1 and Chapter 2). Consequently, CPR-Width is defined relative to a declared task τ and reconstruction model $\mathfrak{M}$, not only relative to a graph or hypergraph representation; different declared tasks on the same instance may induce different CPR-Width values (Proposition 7.5), whereas treewidth, pathwidth, branch-width, and clique-width are representation-relative structural parameters that do not depend on a declared computational task.

The formal relationship between CPR-Width and classical width parameters that the present manuscript is able to establish is developed in Section 9.3.2: for linear admissible orderings, the unnormalized CPR-Width is bounded above by the ordering-specific vertex-separation value whenever the corresponding structural candidate boundary is task-sufficient (Bound (9.37)), and this relation is illustrated on finite path and cycle instances (P_4 and C_4). No equality is claimed in general, no comparison with treewidth, branch-width, or clique-width beyond linear vertex separation is established, and no comparison on constraint-satisfaction or Boolean-satisfiability instances is developed in the present manuscript. Extending the instance-based comparison to CSP and SAT representations, and relating CPR-Width to tree- or branch-structured decompositions rather than only to linear vertex separation, remain open directions; see Section 9.3 and Section 9.4 for the corresponding open-problem and validation agenda.

This section positions CPR-WIDTH within the literature on structural width parameters. It does not assert an equivalence, a reduction, or a uniform inequality between CPR-Width and any classical parameter beyond the one-sided linear-ordering bound established in Section 9.3.2.

9.2 Generalizations and Subclasses

9.2.1 Possible Generalization to CPR-Tractable Subclasses

The realizations in Table 9.1 suggest a common family-level research target. The following statement is not asserted here as a theorem for unrestricted SAT, CSP, graph coloring, tensor models, or arbitrary circuit families. Rather, it is a conditional generalization template. Whenever a nonempty subclass satisfies the uniform CPR requirements already established in Chapters 4–5, the common width–state–runtime chain can be lifted from individual instances to the entire subclass.

Corollary 9.1 (Certification Restatement for a Controlled CPR Subclass)**.** *Let $\mathcal{C}$ be a uniformly encoded*

nonempty subclass of a finite combinatorial problem family. Assume that there exist

- *one fixed task* τ*;*
- *one fixed uniform CPR reconstruction model* $\mathfrak{M}$*;*
- *one polynomial-time preprocessing procedure* $\mathcal{A}_{\mathrm{Preprocess}}$ *containing an ordering-supply component* $\mathcal{A}_\pi$*;*
- *one width-bound function* $w_{\mathcal{C}} : \mathbb{N} \to \mathbb{N}_0$*;*
- *and fixed polynomial functions* $p_j : \mathbb{N} \to \mathbb{N}$*,* $j \in J$*.*

Assume further that, for every encoded instance $P_x \in \mathcal{C}$*,*

$$\pi_x = \mathcal{A}_\pi(x) \in \Pi_{\mathrm{adm}}(P_x, \tau; \mathfrak{M}), \tag{9.4}$$

$$\boldsymbol{\omega}_{\mathrm{rec}}(P_x, \pi_x, \tau; \mathfrak{M}) \le w_{\mathcal{C}}(|x|), \tag{9.5}$$

BC1–BC4 hold uniformly, and

$$N_j(P_x, \pi_x, \tau; \mathfrak{M}) \le p_j(|x|) \qquad (j \in J). \tag{9.6}$$

Then the decision language induced by $\mathcal{C}$ *is solvable in deterministic polynomial time by the declared CPR procedure. If, in addition, there exists a constant*

$$k_{\mathcal{C}} \in \mathbb{N}_0$$

independent of the instance size such that

$$\boldsymbol{\omega}_{\mathrm{rec}}(P_x, \pi_x, \tau; \mathfrak{M}) \le k_{\mathcal{C}} \qquad \textit{for every } P_x \in \mathcal{C}, \tag{9.7}$$

then $\mathcal{C}$ *is a constant-CPR-Width subclass in the sense of Section 6.5.2.*

Remark 9.2 (Status of this corollary)**.** This statement is deliberately labeled a corollary rather than a theorem or proposition: it introduces no mathematical content beyond the uniform tractability conditions of Chapter 5 and the operation-count decomposition of Chapter 4. Its purpose is bookkeeping, not novelty. It consolidates those conditions into a single subclass-level certification criterion, stated once here so that Sections 9.1–9.3 can invoke it by reference instead of re-deriving it for each candidate subclass. No tractability implication stronger than Theorem 5.1 is asserted by this restatement.

Proof. The statement is a family-level specialization of the uniform tractability conditions of Chapter 5 together with the complete operation-count decomposition of Chapter 4. The width bound controls the active-interface contribution. BC1 provides one explicit uniform structural representation. BC2 provides polynomial uniform constructibility. BC3 provides exact task-relative soundness and completeness. BC4 provides polynomial control of the total stage-indexed semantic state and semantic transition structure. Finally, the six polynomial component bounds control the complete execution cost. Therefore,

$$N_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M})$$

is polynomially bounded in $|x|$. The constant-width clause is the special case

$$w_{\mathcal{C}}(|x|) \le k_{\mathcal{C}}.$$

Constant CPR-Width strengthens only the structural part of the hypothesis. It does not replace BC1–BC4 or the complete runtime accounting.

□

Table 9.2 records how this principle could be investigated for the principal realizations already developed in the manuscript. The entries are research targets, not claims that the indicated unrestricted problem classes satisfy the stated conditions.

Potential candidate classes for future subclass analysis include SAT and bounded-occurrence SAT variants, CSP, graph coloring, graph homomorphism, independent set, vertex cover, dominating set, clique, Hamiltonian path and cycle, traveling salesman, set cover, exact cover, subset sum, knapsack, partition, bin packing, scheduling, matching, network flow, routing, Steiner tree, facility location, clustering, constraint-based allocation, tensor-structured problems, Boolean circuits, arithmetic circuits, and ripple-type arithmetic reconstruction. These classes are listed solely as candidates for identifying structurally restricted subclasses satisfying the hypotheses of Corollary 9.1. No tractability claim for the unrestricted classes is implied. The intended future research direction is therefore not to assign a small CPR-Width to an unrestricted problem class by definition. It is to identify mathematically natural subclasses $\mathcal{C}$ for which

$$\forall P_x \in \mathcal{C} : \qquad \boldsymbol{\omega}_{\mathrm{rec}}(P_x, \pi_x, \tau; \mathfrak{M}) \le w_{\mathcal{C}}(|x|) \tag{9.8}$$

can be proved together with BC1–BC4 and all six polynomial operation-count bounds. The strongest structural special case is

Table 9.2: Possible generalization for CPR-controlled subclasses.

Problem class	Possible subclass condition	Target CPR realization	Required conclusion test
SAT	Uniformly encoded formula subclasses admitting constructible task-sufficient interfaces of controlled or constant CPR-Width.	$\boldsymbol{\omega}_{\mathrm{rec}}(\Phi,\pi,\tau;\mathfrak{M}) \to$ $S_{\mathrm{CPR}}(\Phi,\pi,\tau;\mathfrak{M}) \to$ $N_{\mathrm{CPR}}(\Phi,\pi,\tau;\mathfrak{M})$.	Prove BC1–BC4, uniform ordering supply, and polynomial bounds for all six runtime components. No claim is made for unrestricted SAT.
CSP	Constraint subclasses whose unresolved scopes induce uniformly constructible bounded task-sufficient interfaces.	$\boldsymbol{\omega}_{\mathrm{rec}}(P_{\mathrm{CSP}},\pi,\tau;\mathfrak{M}) \to$ $S_{\mathrm{CPR}}(P_{\mathrm{CSP}},\pi,\tau;\mathfrak{M}) \to$ $N_{\mathrm{CPR}}(P_{\mathrm{CSP}},\pi,\tau;\mathfrak{M})$.	Show that interface construction, semantic transitions, reconstruction, verification, and output are uniformly polynomial.
Graph coloring	Graph subclasses admitting uniformly supplied orderings with controlled or constant task-sufficient reconstruction interfaces.	$\boldsymbol{\omega}_{\mathrm{rec}}(G,\pi,\tau;\mathfrak{M}) \to$ $S_{\mathrm{CPR}}(G,\pi,\tau;\mathfrak{M}) \to$ $N_{\mathrm{CPR}}(G,\pi,\tau;\mathfrak{M})$.	Establish the CPR bound independently of merely assuming a classical width bound, then verify BC1–BC4 and the complete cost model.
Tensor	Tensor-structured subclasses with polynomially constructible reconstruction-safe signatures and bounded active structural indices.	$\boldsymbol{\omega}_{\mathrm{rec}}(T_P,\pi,\tau;\mathfrak{M}) \to$ $S_{\mathrm{CPR}}(T_P,\pi,\tau;\mathfrak{M}) \to$ $N_{\mathrm{CPR}}(T_P,\pi,\tau;\mathfrak{M})$.	Prove that tensor compression and signature construction do not hide superpolynomial work and that all induced transitions are polynomial.
Arithmetic / ripple-type	Uniform arithmetic families whose future behavior is controlled by a fixed-size or polynomially bounded carry/memory signature.	$\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \to$ $S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \to$ $N_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M})$.	Generalize the ripple-adder realization only where a uniform task-sufficient signature and complete polynomial accounting can be proved.
Further finite combinatorial subclasses	Any uniformly encoded subclass for which a sound, complete, and constructible CPR model yields controlled or constant width.	$\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \to$ $S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \to$ $N_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M})$.	Apply Corollary 9.1; tractability follows only after all uniform CPR conditions are proved.

$$\forall P_x \in \mathcal{C}: \qquad \boldsymbol{\omega}_{\mathrm{rec}}(P_x,\pi_x,\tau;\mathfrak{M}) \le k_{\mathcal{C}}, \tag{9.9}$$

where $k_{\mathcal{C}}$ is independent of the instance size. Such results would convert the instance-level realizations of Table 9.1 into uniform subclass theorems. Let

$$\mathcal{F} = \{P_x : x \in \{0,1\}^*\}$$

be a uniformly encoded family of finite decision-problem instances with associated decision language

$$L_{\mathcal{F}} = \{x \in \{0,1\}^* : P_x \text{ is a yes-instance}\}. \tag{9.10}$$

A complete uniform CPR decision model requires

- one fixed decision task τ;
- one fixed uniform reconstruction model $\mathfrak{M}$;
- one deterministic preprocessing algorithm $\mathcal{A}_{\mathrm{Preprocess}}$;
- its ordering-supply component $\mathcal{A}_\pi$;
- and six fixed polynomial functions $\{p_j\}_{j\in J}$.

Soundness and completeness are already required uniformly through BC3 and therefore do not constitute a separately quantified object. For every encoding x, let

$$D_x = \mathcal{A}_{\mathrm{Preprocess}}(x) \tag{9.11}$$

denote the complete uniformly constructed CPR representation of P_x, and let

$$\pi_x = \mathcal{A}_\pi(x) \in \Pi_{\mathrm{adm}}(P_x,\tau;\mathfrak{M}) \tag{9.12}$$

denote the ordering supplied by the ordering component. Every preprocessing operation is assigned exactly once to either

$$N_{\mathrm{Build}}$$

or

$$N_{\mathrm{Reduce}},$$

according to the disjoint accounting convention of Chapters 4–5. In particular, the operation count of $\mathcal{A}_\pi$

is included in N_{Build}. The complete uniform tractability requirement is

$$\Bigg[\exists \tau\, \exists \mathfrak{M}\, \exists \mathcal{A}_{\mathrm{Preprocess}}\, \exists \{p_j\}_{j\in J} \\ \forall x \in \{0,1\}^* : \quad D_x = \mathcal{A}_{\mathrm{Preprocess}}(x) \\ \wedge\ \pi_x = \mathcal{A}_\pi(x) \in \Pi_{\mathrm{adm}}(P_x, \tau; \mathfrak{M}) \\ \wedge \bigwedge_{k=1}^{4} \mathrm{BC}_k(P_x, \pi_x, \tau; \mathfrak{M}) \\ \wedge \bigwedge_{j\in J} [N_j(P_x, \pi_x, \tau; \mathfrak{M}) \le p_j(|x|)]\Bigg] \\ \Longrightarrow L_{\mathcal{F}} \in \mathbf{P}. \tag{9.13}$$

Here $\mathcal{A}_\pi$ denotes the ordering-supply component of $\mathcal{A}_{\mathrm{Preprocess}}$ and is not separately quantified. The quantifier order is essential. The task, reconstruction model, preprocessing algorithm including its ordering-supply component, and all polynomial bounds must be fixed for the complete encoded family before an individual encoding is selected. Under the complete definition of $\mathbf{P}_{\mathrm{CPR}}$ in Chapter 5,

$$\mathbf{P}_{\mathrm{CPR}} \subseteq \mathbf{P}. \tag{9.14}$$

This inclusion is not a complexity-class collapse statement. A separate formal question remains open in the present framework:

$$\mathbf{P}_{\mathrm{CPR}} \stackrel{?}{=} \mathbf{P}.$$

The manuscript proves only $\mathbf{P}_{\mathrm{CPR}} \subseteq \mathbf{P}$. It does not prove that every deterministic polynomial-time language admits a CPR certificate satisfying the declared non-gameability, uniform constructibility, task-sufficiency, semantic-transition, and complete accounting requirements. Determining whether the certification language is extensionally equal to $\mathbf{P}$ or is strictly more restrictive is therefore left as an open formal question. For an NP-complete language L^*, the implication

$$L^* \in \mathbf{P}_{\mathrm{CPR}} \Longrightarrow \mathbf{P} = \mathbf{NP} \tag{9.15}$$

remains strictly conditional. No NP-complete language is proved in this work to belong to $\mathbf{P}_{\mathrm{CPR}}$. Chapter 6 presents structural realization templates, controlled-width subclass theory, and finite illustrative calculations. Chapter 7 establishes symbolic arithmetic reconstruction results and a uniform carry-transducer analysis. Chapter 8 provides an exhaustive finite verification record for the isolated full-adder Hamming-weight factorization and supplements the formal theory with measured ripple-adder runtime data, exact brute-force cross-checks on small adder instances, and a structural interface-growth experiment on random 3-SAT instances. Peak memory was traced during these runs but, as stated in Section 8.6.1, is not part of the reported record. These empirical records provide finite, structural, and measured evidence only under the stated implementation and benchmark conditions. None of these results by itself establishes asymptotic tractability for unrestricted problem classes. Count-level gain and measured wall-clock gain must also remain separate. A favorable elementary-operation count is a statement within the declared accounting model. A measured runtime claim requires a separate implementation, hardware description, software environment, benchmark protocol, and reproducible measurement procedure.

9.3 Structural and Algorithmic Open Problems

The first group of open problems concerns the internal mathematical structure of CPR reconstruction models. The central questions are

- whether favorable reconstruction orderings can be constructed efficiently;
- whether exact semantic-state encodings can be computed without exhaustive future enumeration;
- how tight the width-based state bound is;
- how CPR-Width relates to classical width parameters;
- and which nontrivial uniformly encoded families satisfy the complete CPR tractability requirements.

9.3.1 Reconstruction Orderings, Width Ratio, and Semantic-State Construction

Let

$$\Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M})$$

denote the nonempty set of admissible reconstruction orderings. The minimum normalized CPR-Width is

$$\boldsymbol{\omega}_{\mathrm{rec}}(P, \tau; \mathfrak{M}) = \min_{\pi \in \Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})} \boldsymbol{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}). \tag{9.16}$$

The corresponding unnormalized minimum is

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\tau;\mathfrak{M}) = \min_{\pi\in\Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})} \widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}). \tag{9.17}$$

The definition does not imply that a minimizing ordering can be found efficiently. The exact ordering problem is to determine

$$\pi^* \in \operatorname*{arg\,min}_{\pi\in\Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})} \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}). \tag{9.18}$$

A complete algorithmic analysis should distinguish the following questions:

- whether the exact minimum width can be computed in polynomial time for a specified uniformly encoded family;
- whether one can decide
$$\boldsymbol{\omega}_{\mathrm{rec}}(P,\tau;\mathfrak{M}) \le k$$
for a declared integer k;
- whether polynomial-time approximation algorithms exist when exact minimization is difficult;
- whether fixed-parameter algorithms are possible with respect to $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}$, $q(P)$, or another declared structural parameter;
- whether useful orderings can be constructed without first constructing the complete reachable prefix-state space or semantic quotient;
- and whether structural lower bounds on CPR-Width can be obtained directly from the encoded instance.

Width ratio. Let

$$\widetilde{\pi} \in \Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})$$

be a declared admissible ordering. The tilde marks an arbitrary ordering symbol here and must not be confused with the hat accent used for the unnormalized width $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}$. Define

$$\alpha_{\mathrm{width}}(P,\widetilde{\pi},\tau;\mathfrak{M}) = \frac{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\widetilde{\pi},\tau;\mathfrak{M})+1}{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\tau;\mathfrak{M})+1}. \tag{9.19}$$

Adding one guarantees a positive denominator while preserving the exact optimality criterion. This is a structural width ratio. It is not an operation-count ratio or measured runtime overhead.

Proposition 9.1 (Width-Ratio Bound)**.** *For every admissible ordering $\widetilde{\pi}$,*

$$\alpha_{\mathrm{width}}(P,\widetilde{\pi},\tau;\mathfrak{M}) \ge 1, \tag{9.20}$$

with equality if and only if

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\widetilde{\pi},\tau;\mathfrak{M}) = \widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\tau;\mathfrak{M}). \tag{9.21}$$

Proof. By definition of the minimum unnormalized width,

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\widetilde{\pi},\tau;\mathfrak{M}) \ge \widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\tau;\mathfrak{M}).$$

Adding one to both sides and dividing by the strictly positive denominator gives

$$\alpha_{\mathrm{width}}(P,\widetilde{\pi},\tau;\mathfrak{M}) \ge 1.$$

Equality holds exactly when the two unnormalized width values are equal.

□

Efficient semantic-state construction. At stage i, the semantic quotient is

$$S_i(P,\pi,\tau;\mathfrak{M}) = U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})/{\equiv^{\tau,\mathfrak{M}}_{P,\pi,i}}, \tag{9.22}$$

with primary semantic equivalence

$$u \equiv^{\tau,\mathfrak{M}}_{P,\pi,i} v \quad\Longleftrightarrow\quad \widehat{\mathrm{Sig}}^{\pi}_{\tau,i}(u) = \widehat{\mathrm{Sig}}^{\pi}_{\tau,i}(v). \tag{9.23}$$

Thus, semantic equivalence is defined by equality of induced reconstruction-safe signatures. It is not generally defined by literal equality of complete future-continuation sets. Exact continuation-set equality may be used as a stronger special signature realization only when it is well defined independently of the selected prefix representative, satisfies S1–S3, and is required by the declared task. The principal algorithmic question is whether semantic classes admit a finite efficiently computable canonical encoding. Let

$$\chi_i : U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M}) \longrightarrow \Gamma_i(P, \pi, \tau; \mathfrak{M}) \tag{9.24}$$

be a finite encoding map. The symbol χ_i is used to avoid conflict with the stage-wise branching number λ_i of Chapter 4 and with the admissible next-value sets Λ_i^π. The encoding is exact if

$$\chi_i(u) = \chi_i(v) \quad \Longleftrightarrow \quad u \equiv_{P,\pi,i}^{\tau,\mathfrak{M}} v. \tag{9.25}$$

Under this condition, the fibers of χ_i coincide exactly with the semantic equivalence classes, producing a natural bijection

$$S_i(P, \pi, \tau; \mathfrak{M}) \cong \chi_i\left(U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M})\right). \tag{9.26}$$

The construction of polynomial-time exact encodings for nontrivial uniformly encoded families remains an open problem. A one-sided condition must be interpreted carefully. If

$$\chi_i(u) = \chi_i(v) \Longrightarrow u \equiv_{P,\pi,i}^{\tau,\mathfrak{M}} v, \tag{9.27}$$

the encoding may refine the semantic quotient by assigning different labels to semantically equivalent states. This may reduce compression but does not merge semantically different states. Consequently,

$$\left|\chi_i\left(U_i^{\mathrm{reach}}\right)\right| \geq |S_i|. \tag{9.28}$$

If implication (9.27) fails, the encoding may merge states whose reconstruction-safe behavior differs. Such merging may destroy task preservation, admissible-next-value preservation, successor congruence, soundness, or completeness. Approximate state merging must therefore be separated from exact CPR semantic quotienting and requires an explicit approximation guarantee.

Resolved family example: ripple-adder quotient construction. The general construction problem remains open for arbitrary uniformly encoded families. The ripple-adder family of Chapter 7 provides one explicit case in which semantic-state construction does not require enumeration of the complete suffix space. The reconstruction-safe state is the carry bit. At every internal bit position,

$$c_i \in \{0, 1\}.$$

The successor signature is obtained locally from the current carry and the next reconstruction-component value by the update mechanism proved in Chapter 7. Consequently, the quotient state alphabet has constant cardinality and the stage-wise update is local. Across an n-bit ripple-adder schema, the complete stage-indexed semantic quotient and transition structure are constructible in $O(n)$ elementary operations. This is a resolved construction result for one declared family. It is not a general polynomial-construction theorem for arbitrary CPR models.

9.3.2 Width-Bound Slack and Relationship to Classical Width Parameters

The sharp operational Width–State Bound is

$$S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) \leq q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}. \tag{9.29}$$

Since

$$S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) \geq 1,$$

define the sharp width-bound slack factor by

$$\kappa_{\mathrm{bound}}(P, \pi, \tau; \mathfrak{M}) = \frac{q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}}{S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M})}. \tag{9.30}$$

Proposition 9.2 (Width-Bound Slack)**.** *For every finite problem instance, declared task, reconstruction model, and admissible ordering,*

$$\kappa_{\mathrm{bound}}(P, \pi, \tau; \mathfrak{M}) \geq 1. \tag{9.31}$$

Proof. By Equation (9.29),

$$S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) \leq q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}.$$

Dividing the upper bound by the positive semantic-state count gives

$$\kappa_{\mathrm{bound}}(P, \pi, \tau; \mathfrak{M}) \geq 1.$$

□

A value of κ_{bound} close to one indicates that the general raw-interface-state bound is close to the observed maximum semantic-state count. A large value indicates that reachability restrictions or semantic quotienting produce a substantially smaller state space than the raw width bound. The slack factor is not a measured compression speedup, an operation-count gain, or a wall-clock gain.

Concrete comparison on the path graph P_4. To illustrate the structural relationship between CPR-Width and classical width parameters on a finite instance, consider the four-vertex path graph P_4 with vertices v_1, v_2, v_3, v_4 and edges $\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}$.
Pathwidth. The pathwidth of P_4 is $\mathrm{pw}(P_4) = 1$. At each internal stage $1 \le i \le 3$, the vertex-separation boundary has cardinality 1, consisting of the most recently processed vertex, while at the terminal stage $i = 4$ the boundary is empty. Hence the maximum vertex-separation value is 1.
CPR-Width for the 2-coloring decision task. For the decision variant of proper 2-coloring with the natural linear ordering $\pi = (v_1, v_2, v_3, v_4)$ and the reconstruction-safe signature family $\mathrm{Sig}^{\pi}_{\tau,i}(r) = r|_{\{v_i\}}$ (the color of the most recently processed vertex), the running example of Chapter 2 (Section 2.12) yields

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P_4, \pi, \tau_{\mathrm{dec}}; \mathfrak{M}) = 1, \qquad \boldsymbol{\omega}_{\mathrm{rec}}(P_4, \pi, \tau_{\mathrm{dec}}; \mathfrak{M}) = 0.$$

The unnormalized CPR-Width $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}} = 1$ coincides with the pathwidth bound $\mathrm{pw}(P_4) = 1$, while the normalized convention reports $\boldsymbol{\omega}_{\mathrm{rec}} = 0$.
Interpretation. This finite sanity check shows that, for this particular representation, task, and ordering, the CPR interface cardinality is upper-controlled by the classical pathwidth. At every internal stage $1 \le i \le 3$, the structural candidate boundary $\widetilde{B}^{G,\pi}_i$ (processed vertices adjacent to unprocessed vertices) is task-sufficient for the declared reconstruction-safe signature, and cardinality minimality yields $b^{\pi,\tau}_i(P_4) = |\widetilde{B}^{G,\pi}_i| = 1$. At the terminal stage $i = 4$, both quantities are zero.
Caveats. This observation does not establish a general equality or inequality between CPR-Width and pathwidth.

- CPR-Width is additionally task-relative: a different declared task (e.g., exact witness reconstruction) may require a strictly larger interface.
- The comparison depends on the chosen ordering: Proposition 2.2 shows that different admissible orderings can yield different CPR-Width values on the same instance.
- The normalized minus-one convention means that $\boldsymbol{\omega}_{\mathrm{rec}} = 0$ may correspond to either an empty maximum interface or a one-component maximum interface, so direct numerical comparison with pathwidth must use $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}$.

A formal theorem relating minimum CPR-Width over all admissible orderings to pathwidth, treewidth, or another classical width parameter requires a fixed representation, task, signature family, and admissibility convention, and is developed further below and as an open problem in this section.

Concrete comparison on the cycle graph C_4. As a second illustration, consider the four-vertex cycle graph C_4 with vertices v_1, v_2, v_3, v_4 and edges $\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}, \{v_4, v_1\}$.
Pathwidth and treewidth. The pathwidth of C_4 is $\mathrm{pw}(C_4) = 2$, and its treewidth is $\mathrm{tw}(C_4) = 2$. Because C_4 contains a cycle, no linear ordering can reduce the maximum vertex separator below cardinality 2: removing any single vertex from C_4 leaves a connected path on the remaining three vertices, so the wrap-around edge always keeps two processed vertices simultaneously adjacent to the unresolved suffix at some stage.
CPR-Width for the 3-coloring decision task. Consider the decision variant of proper 3-coloring with the natural ordering $\pi = (v_1, v_2, v_3, v_4)$ and the reconstruction-safe signature family

$$\mathrm{Sig}^{\pi}_{\tau,i}(r) = r|_{\widetilde{B}^{G,\pi}_i}$$

For this declared 3-coloring decision model, the signature is reconstruction-safe. Equality of the boundary-color assignment preserves the residual feasibility status because every edge crossing from the processed prefix to the unresolved suffix is incident with a vertex in $\widetilde{B}^{G,\pi}_i$. It also preserves the admissible color set of the next vertex, and after assigning the same admissible next color, the successor boundary-color assignments agree. Hence conditions S1–S3 hold for this finite realization.
(the colors of the processed vertices that remain adjacent to an unresolved vertex), generalizing the single-vertex convention $r|_{\{v_i\}}$ used in the running example of Chapter 2, where the structural boundary always had cardinality one. The structural candidate boundary at stage i is

$$\widetilde{B}^{G,\pi}_i = \{v \in V^{\pi}_i : \exists w \in V \setminus V^{\pi}_i \text{ with } \{v, w\} \in E\}.$$

Tracking the interface stage by stage:

- Stage 1 (v_1 processed): $\widetilde{B}^{G,\pi}_1 = \{v_1\}$, $|\widetilde{B}^{G,\pi}_1| = 1$.
- Stage 2 (v_1, v_2 processed): $\widetilde{B}^{G,\pi}_2 = \{v_1, v_2\}$, $|\widetilde{B}^{G,\pi}_2| = 2$ (v_1 remains adjacent to unprocessed v_4 through the wrap-around edge $\{v_4, v_1\}$, and v_2 is adjacent to unprocessed v_3).
- Stage 3 (v_1, v_2, v_3 processed): $\widetilde{B}^{G,\pi}_3 = \{v_1, v_3\}$, $|\widetilde{B}^{G,\pi}_3| = 2$ (both v_1 and v_3 are adjacent to unprocessed v_4; v_2 is no longer adjacent to any unresolved vertex).

- Stage 4 (all vertices processed): $\widetilde{B}_4^{G,\pi} = \varnothing$, $|\widetilde{B}_4^{G,\pi}| = 0$.

This yields the structural candidate-boundary maximum

$$\max_i \left|\widetilde{B}_i^{G,\pi}\right| = 2.$$

The declared signature family $\mathrm{Sig}_{\tau,i}^{\pi}(r) = r|_{\widetilde{B}_i^{G,\pi}}$ is task-sufficient by construction, so Bound (6.1) gives the conditional upper bound

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(C_4, \pi, \tau_{\mathrm{dec}}; \mathfrak{M}) \leq 2, \qquad \boldsymbol{\omega}_{\mathrm{rec}}(C_4, \pi, \tau_{\mathrm{dec}}; \mathfrak{M}) \leq 1.$$

Interpretation. The maximum structural candidate-boundary size, 2, coincides with both $\mathrm{pw}(C_4) = 2$ and $\mathrm{tw}(C_4) = 2$. The cycle structure forces a larger interface than the path P_4 because the wrap-around edge $\{v_4, v_1\}$ keeps v_1 in the active interface from stage 2 onward: from that stage on, two processed vertices simultaneously remain adjacent to the unresolved suffix, until v_4 is finally processed. Establishing $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(C_4, \pi, \tau_{\mathrm{dec}}; \mathfrak{M}) = 2$ as an equality, rather than only the upper bound above, requires the cardinality-minimality proof noted in the Caveats below; no such proof is given here.

Caveats. As with P_4, this is a single-instance comparison.

- The CPR-Width value depends on the chosen ordering π. A different ordering (e.g., $\pi' = (v_1, v_3, v_2, v_4)$) yields a different interface sequence, although it also attains maximum cardinality 2 for this instance.
- The comparison uses the decision task for 3-coloring. A witness reconstruction task might require additional information to be retained.
- The structural candidate boundary $\widetilde{B}_i^{G,\pi}$ is task-sufficient by construction for the declared signature family above, but cardinality minimality has not been established here. Consequently only the upper bound $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(C_4, \pi, \tau_{\mathrm{dec}}; \mathfrak{M}) \leq 2$ above is certified; the exact minimum CPR-Width for this instance remains open.

This example reinforces the pattern observed for P_4: for these graph-coloring instances, the CPR interface cardinality is upper-controlled by the classical pathwidth. These two finite comparisons illustrate the structural relationship on simple graph instances; the general vertex-separation bound underlying both is established next, and a systematic family-level comparison across broader graph classes remains an important direction for future work.

Relationship to classical width parameters. The comparison with classical width notions serves two distinct purposes:

1. identifying structural regimes in which CPR-Width is controlled by an established width parameter;
2. isolating the additional task-relative information measured by CPR-WIDTH.

Pathwidth and treewidth are representation-relative structural parameters. CPR-Width is additionally relative to the declared computational task, reconstruction model, reconstruction-safe signature family, and admissible ordering convention. A formal comparison must therefore fix all of these objects. The normalized and unnormalized CPR widths satisfy

$$\boldsymbol{\omega}_{\mathrm{rec}} = \max\left\{0, \widehat{\boldsymbol{\omega}}_{\mathrm{rec}} - 1\right\}. \tag{9.32}$$

Because this normalization maps both $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}} = 0$ and $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}} = 1$ to $\boldsymbol{\omega}_{\mathrm{rec}} = 0$, it is not injective and therefore does not retain the full operational interface cardinality. Whether the normalized minus-one convention is preferable to reporting the operational interface width directly remains a definitional design question for future versions of the framework. All operational state-count and runtime bounds in the present manuscript continue to use $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}$ explicitly where that distinction matters. The general open problem may be formulated using the normalized width

$$\boldsymbol{\omega}_{\mathrm{rec}},$$

consistent with its role as the reported CPR-Width parameter. The concrete comparison with vertex separation is more naturally stated using

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}},$$

because vertex separation is itself a maximum boundary cardinality. Every admissible reconstruction ordering in the present manuscript is a linear ordering of components. For the graph representation of Section 6.3, define the structural candidate boundary

$$\widetilde{B}_i^{G,\pi} = \left\{v \in V_i^{\pi} : \exists w \in V \setminus V_i^{\pi} \text{ with } \{v, w\} \in E\right\}. \tag{9.33}$$

For the ordering π, define its vertex-separation value by

$$\mathrm{vs}(G, \pi) = \max_i \left|\widetilde{B}_i^{G,\pi}\right|. \tag{9.34}$$

The structural candidate interface must not be identified automatically with the certified cardinality-minimal CPR interface $B_i^{\pi,\tau}$. If $\widetilde{B}_i^{G,\pi}$ is proved task-sufficient for the declared reconstruction-safe signature,

then minimality gives

$$|B_i^{\pi,\tau}| \le \left|\widetilde{B}_i^{G,\pi}\right|. \tag{9.35}$$

Therefore,

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \le \max_i \left|\widetilde{B}_i^{G,\pi}\right| = \mathrm{vs}(G,\pi). \tag{9.36}$$

For a graph representation G_P, define the minimum vertex-separation value over all linear vertex orderings by

$$\mathrm{vs}(G_P) = \min_{\pi} \mathrm{vs}(G_P,\pi).$$

Consequently, if there exists an ordering

$$\pi^* \in \Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})$$

that attains $\mathrm{vs}(G_P)$ and whose candidate boundary is task-sufficient at every stage, then

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\tau;\mathfrak{M}) \le \mathrm{vs}(G_P). \tag{9.37}$$

This is a conditional one-sided relation. Equality is not claimed in general. The minimum task-sufficient CPR interface may be strictly smaller than the structural separator. This effect is distinct from subsequent semantic quotienting. Semantic quotienting acts after the active interface has been selected:

$$B_i^{\pi,\tau} \longrightarrow U_i^{\mathrm{reach}} \longrightarrow S_i. \tag{9.38}$$

It does not retroactively change the cardinality of $B_i^{\pi,\tau}$. For the running path example P_4 of Chapter 2,

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P_4,\pi,\tau;\mathfrak{M}) = 1$$

and

$$\boldsymbol{\omega}_{\mathrm{rec}}(P_4,\pi,\tau;\mathfrak{M}) = 0.$$

The structural candidate boundary has maximum cardinality one and is task-sufficient, while

$$\mathrm{pw}(P_4) = 1.$$

Thus,

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P_4,\tau;\mathfrak{M}) \le \mathrm{pw}(P_4) = 1. \tag{9.39}$$

For the declared natural ordering π, one therefore has

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P_4,\pi,\tau;\mathfrak{M}) = 1 = \mathrm{pw}(P_4).$$

No equality is inferred here for the ordering-minimized quantity $\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P_4,\tau;\mathfrak{M})$ without a separate proof that no admissible ordering has smaller unnormalized CPR-Width. This is only a finite sanity check. It is not a general comparison theorem. Treewidth is naturally associated with tree-shaped elimination or decomposition structures rather than purely linear orderings. Because the present CPR formulation uses linear reconstruction orderings, a general CPR-Width comparison with treewidth may require a different argument or an extension of the admissible reconstruction structure beyond linear sequences. Such an extension remains future work.

9.3.3 Identification of Additional CPR-Tractable Families

The present work establishes

- an exact finite output-factorization result for the isolated full adder;
- and a uniform carry-interface reconstruction result for the ripple-adder family.

The broader classification problem is to identify nontrivial uniformly encoded infinite families satisfying BC1–BC4 together with the complete six-component polynomial-cost requirements. For a candidate uniformly encoded family

$$\mathcal{F} = \{P_x : x \in \{0,1\}^*\},$$

a complete CPR-tractability proof must provide one uniform preprocessing algorithm whose ordering-supply component produces

$$\pi_x \in \Pi_{\mathrm{adm}}(P_x,\tau;\mathfrak{M}), \tag{9.40}$$

together with fixed polynomial functions

$$p_j : \mathbb{N} \longrightarrow \mathbb{N}, \qquad j \in J, \tag{9.41}$$

such that

$$N_j(P_x, \pi_x, \tau; \mathfrak{M}) \le p_j(|x|) \tag{9.42}$$

for every encoding x and every $j \in J$. Promising research areas include

- restricted SAT and 3-SAT families;
- bounded-domain CSP families;
- graph-coloring subclasses;
- Hamiltonian-path and Hamiltonian-cycle subclasses;
- bounded-interface routing and scheduling systems;
- arithmetic and Boolean circuit families;
- tensor-network families;
- database-join families;
- coding and decoding systems;
- and other families admitting exact local reconstruction-safe continuation summaries.

No family should be classified as CPR-tractable merely because a compact representation or small structural candidate interface has been identified. The complete uniform requirement must be proved.

Most compelling presently supported candidate. Among the realizations presently studied, bounded-state sequential arithmetic systems provide the strongest currently supported example of controlled or constant CPR-Width. The ripple-adder family provides the concrete prototype:

- reconstruction proceeds through a fixed local carry state;
- the active-interface cardinality does not grow with bit length;
- the successor signature is locally computable;
- the semantic state alphabet is bounded;
- and the required state information propagates sequentially rather than through an expanding global boundary.

This does not imply the same result for arbitrary Boolean circuits, SAT, or general CSP. None of those unrestricted classes receives a family-level BC1–BC4 tractability certification in this manuscript. The supported direction is therefore bounded-state sequential arithmetic and, more generally, finite-state-transducer-like reconstruction families.

9.4 Validation, Comparative Analysis, and Future Research Program

The second group of open problems concerns implementation, measurement, comparison, reproducibility, and evidence classification. The objective is not merely to produce a working prototype. A valid prototype should expose every structural and computational layer of the declared CPR procedure.

9.4.1 Prototype Architecture and Reference Pipeline

For a finite problem instance P, task τ, reconstruction model $\mathfrak{M}$, and ordering π, define the core structural and operation-count record

$$\begin{aligned}\mathcal{R}_{\mathrm{core}}(P, \pi, \tau; \mathfrak{M}) = (&\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}, \boldsymbol{\omega}_{\mathrm{rec}}, S_{\mathrm{CPR}}, V_{\mathrm{sem}}, E_{\mathrm{sem}}, m_{\mathrm{stage}}, \\ &N_{\mathrm{Build}}, N_{\mathrm{Reduce}}, N_{\mathrm{Transition}}, N_{\mathrm{Reconstruct}}, N_{\mathrm{Verify}}, N_{\mathrm{Output}}).\end{aligned} \tag{9.43}$$

All quantities in Equation (9.43) are evaluated at $(P, \pi, \tau; \mathfrak{M})$. The total stage-indexed semantic-state count is

$$V_{\mathrm{sem}}(P, \pi, \tau; \mathfrak{M}) = \sum_{i=0}^{m_{\mathrm{stage}}(P,\pi)} \left|S_i(P, \pi, \tau; \mathfrak{M})\right|. \tag{9.44}$$

The total semantic-transition count is

$$E_{\mathrm{sem}}(P, \pi, \tau; \mathfrak{M}) = \sum_{i=0}^{m_{\mathrm{stage}}(P,\pi)-1} \left|E_i^{\mathrm{sem}}(P, \pi, \tau; \mathfrak{M})\right|, \tag{9.45}$$

where E_i^{sem} denotes the reachable labeled semantic quotient transitions from stage i to stage $i+1$. If $m_{\mathrm{stage}}(P, \pi) = 0$, the sum is interpreted as zero. A complete measured implementation record should additionally include the following fields; this is a prescriptive template for future benchmark reporting, not a description of the record actually reported in Chapter 8, where peak memory was traced but is explicitly not part of the published measurement record (Section 8.6.1):

$$\mathcal{R}_{\mathrm{measured}} = (T_{\mathrm{CPR}}, M_{\mathrm{peak}}, L_{\mathrm{Output}}, \mathcal{I}_{\mathrm{CPR}}, H, S_{\mathrm{env}}, P_{\mathrm{measure}}), \tag{9.46}$$

where the entries represent

- measured wall-clock time;

- peak memory use;
- encoded output length;
- implementation identifier;
- hardware identifier;
- software-environment identifier;
- and measurement-protocol identifier.

All fields must refer to the same representation, task, model, ordering, correctness requirement, and output requirement. A prototype that reports only the reduced-state count or only the transition count does not test the complete CPR runtime framework. A reproducible implementation should respect the non-circular construction order of Chapters 2 and 3. The reference execution pipeline is:

1. parse and validate the encoded instance;
2. construct the complete initial CPR representation;
3. construct or read an admissible reconstruction ordering;
4. generate the processed-prefix and unresolved-suffix sequence;
5. generate the reachable prefix-state spaces under the declared reachability and transition rules;
6. compute, retrieve, or certify the reconstruction-safe prefix signatures;
7. determine task-sufficient interfaces at every stage;
8. select a cardinality-minimal task-sufficient interface whenever exact CPR-Width is required;
9. construct the raw interface-state spaces;
10. project reachable prefix states onto the selected interfaces;
11. construct the induced reconstruction-safe interface-signature maps;
12. form the semantic quotients induced by the reconstruction-safe signatures;
13. construct and process semantic quotient transitions;
14. perform task-relevant reconstruction;
15. verify soundness and completeness for the declared task;
16. materialize the required output;
17. and record all structural quantities, operation counts, memory use, and measured runtime data.

In particular, Step 12 does not generally mean quotienting by literal future-continuation-set equality. The formal quotient is induced by the certified reconstruction-safe signature. For every stage i, define the stage record

$$\begin{aligned}\mathcal{A}_i(P,\pi,\tau;\mathfrak{M}) = (&b_i^{\pi,\tau}(P), |U_i(P,\pi,\tau;\mathfrak{M})|,\\ &|U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})|, |S_i(P,\pi,\tau;\mathfrak{M})|).\end{aligned} \tag{9.47}$$

Define the complete stage sequence by

$$\mathcal{A}(P,\pi,\tau;\mathfrak{M}) = \left(\mathcal{A}_0,\ldots,\mathcal{A}_{m_{\mathrm{stage}}(P,\pi)}\right). \tag{9.48}$$

This record separates

- interface width;
- raw interface-state growth;
- reachability filtering;
- semantic quotienting;
- transition cost;
- reconstruction cost;
- verification cost;
- and output cost.

9.4.2 Comparative Baselines, Width Parameters, and Validation Audit

Every comparative claim requires a declared classical baseline. Let

$$N_{\mathrm{Classic}}(P,\tau;\mathcal{A}_{\mathrm{Classic}})$$

denote the complete elementary-operation count of a declared classical algorithm. Let

$$T_{\mathrm{Classic}}(P,\tau;\mathcal{I}_{\mathrm{Classic}})$$

denote the measured wall-clock time of the corresponding classical implementation. Let

$$T_{\mathrm{CPR}}(P,\pi,\tau;\mathcal{I}_{\mathrm{CPR}}) > 0$$

denote the measured wall-clock time of the declared CPR implementation. Assuming

$$N_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) > 0,$$

the count-level gain is

$$G_{\text{count}} = \frac{N_{\text{Classic}}(P, \tau; \mathcal{A}_{\text{Classic}})}{N_{\text{CPR}}(P, \pi, \tau; \mathfrak{M})}, \tag{9.49}$$

whereas the measured wall-clock gain is

$$G_{\text{time}} = \frac{T_{\text{Classic}}(P, \tau; \mathcal{I}_{\text{Classic}})}{T_{\text{CPR}}(P, \pi, \tau; \mathcal{I}_{\text{CPR}})}. \tag{9.50}$$

The two gains remain distinct:

$$G_{\text{count}} > 1 \not\Longrightarrow G_{\text{time}} > 1. \tag{9.51}$$

A valid comparison must fix

- problem representation;
- computational task;
- required output;
- correctness criterion;
- preprocessing;
- stopping condition;
- classical algorithm;
- CPR reconstruction model;
- both implementations;
- hardware;
- software environment;
- and measurement protocol.

For graph-based or hypergraph-based benchmarks, the experimental record may include classical width parameters of the declared representation $G_P = G(P)$. One possible comparison record is

$$\mathcal{W}(P, \pi, \tau; \mathfrak{M}, G_P) = (\boldsymbol{\omega}_{\text{rec}}(P, \pi, \tau; \mathfrak{M}), \text{tw}(G_P), \text{pw}(G_P), \text{bw}(G_P), \text{cw}(G_P)) . \tag{9.52}$$

Treewidth tw, pathwidth pw, branch-width bw, and clique-width cw are included here, consistently with the four classical width parameters that are given a primary reference in the bibliography (treewidth: Bodlaender, 1998 [1]; Robertson and Seymour, 1986 [7]; pathwidth, via the equivalent vertex-separation characterization: Kinnersley, 1992 [5]; branch-width: Robertson and Seymour, 1991 [8]; clique-width: Courcelle and Olariu, 2000 [2]) and discussed elsewhere in this manuscript (Section 9.1). No formal relation between CPR-Width and branch-width or clique-width is established anywhere in the present manuscript; their inclusion in this comparison record is for future empirical and structural study only, not a claim that such a relation has been proved. Such a record is intended to identify candidate upper bounds, lower bounds, separation examples, ordering effects, and representation dependence. Empirical correlation does not establish a formal theorem.

Boundary-condition audit. The CPR audit must distinguish finite instance records from uniform family-level certification. For one finite benchmark instance P, define the instance-level record

$$\mathcal{A}_{\text{BC}}^{\text{inst}}(P) = \left(\sigma_1^{\text{inst}}(P), \sigma_2^{\text{inst}}(P), \sigma_3^{\text{inst}}(P), \sigma_4^{\text{inst}}(P)\right). \tag{9.53}$$

Each component takes one of the values

$$\{\text{verified}, \text{failed}, \text{undetermined}\}.$$

The entries respectively record whether

1. the required structural objects are explicitly available;
2. the finite construction process and its measured operation count are completely recorded;
3. soundness and completeness are verified for the declared finite task;
4. and the complete stage-indexed semantic-state and transition record is available.

An instance-level record does not prove polynomial family behavior. For a uniformly encoded family $\mathcal{F}$, define the family-level certification record

$$\mathcal{A}_{\text{BC}}^{\text{fam}}(\mathcal{F}) = \left(\sigma_1^{\text{fam}}(\mathcal{F}), \sigma_2^{\text{fam}}(\mathcal{F}), \sigma_3^{\text{fam}}(\mathcal{F}), \sigma_4^{\text{fam}}(\mathcal{F})\right), \tag{9.54}$$

with values in

$$\{\text{proved}, \text{disproved}, \text{open}\}.$$

Only the family-level record can support CPR-tractability certification. BC2 and BC4 are asymptotic uniform conditions and cannot be proved by one finite benchmark instance. BC4 controls the structural

quantities

$$V_{\mathrm{sem}}$$

and

$$E_{\mathrm{sem}},$$

whereas the actual elementary-operation cost of processing semantic transitions remains the separate runtime quantity

$$N_{\mathrm{Transition}}.$$

If any required family-level condition remains open or is disproved, the family is not certified as CPR-tractable.

Benchmark families and ordering ablation. A future benchmark corpus should contain favorable, neutral, and unfavorable instances. The purpose is to determine

- where CPR-WIDTH produces small active interfaces;
- where semantic quotienting is effective;
- where quotient construction dominates runtime;
- where reconstruction or output dominates runtime;
- and where CPR-WIDTH provides no computational advantage.

Candidate benchmark groups include Boolean satisfiability instances with controlled clause-interaction structure, finite CSP instances with varying domain size and constraint arity, graph-coloring instances with varying density and separator structure, arithmetic systems, tensor systems, routing and scheduling problems, coding and decoding systems, and deliberately selected negative-control instances. For each benchmark family, the instance-generation procedure, encoding, task, reconstruction model, admissibility rules, output requirements, baseline algorithms, and correctness criteria must be documented. Because CPR-Width is ordering-dependent, experiments should compare multiple admissible orderings

$$\Pi_{\mathrm{test}}(P,\tau;\mathfrak{M}) = \{\pi_1,\dots,\pi_r\} \subseteq \Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M}) \tag{9.55}$$

while keeping every other model component fixed. For each ordering π_j, record

$$\begin{aligned}&(\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi_j,\tau;\mathfrak{M}), \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi_j,\tau;\mathfrak{M}),\\ &\ S_{\mathrm{CPR}}(P,\pi_j,\tau;\mathfrak{M}), N_{\mathrm{CPR}}(P,\pi_j,\tau;\mathfrak{M}),\\ &\ T_{\mathrm{CPR}}(P,\pi_j,\tau;\mathcal{I}_{\mathrm{CPR}})).\end{aligned} \tag{9.56}$$

Only the ordering may vary. The representation, task, reconstruction model, signature family, implementation, hardware, software environment, output requirement, and measurement protocol must remain fixed. A smaller width need not imply a smaller measured runtime:

$$\begin{aligned}&\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi_a,\tau;\mathfrak{M}) < \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi_b,\tau;\mathfrak{M})\\ &\quad \not\Longrightarrow T_{\mathrm{CPR}}(P,\pi_a,\tau;\mathcal{I}_{\mathrm{CPR}}) < T_{\mathrm{CPR}}(P,\pi_b,\tau;\mathcal{I}_{\mathrm{CPR}}).\end{aligned} \tag{9.57}$$

9.4.3 Evidence Classification and Future Research Sequence

CPR-WIDTH results should be classified by the specific evidence categories supporting them. The categories do not form one linear hierarchy. A result may belong to more than one category. The classification is:

E0 – Definition Evidence.
A symbol, object, relation, operator, or structural quantity is introduced formally.

E1 – Finite Verification Evidence.
A claim is checked exhaustively or directly on one explicitly stated finite example.

E2 – Structural Pattern Evidence.
A recurring structural pattern is identified across sampled, constructed, or representative instances.

E3 – Operation-Count Evidence.
A finite or symbolic elementary-operation comparison is made relative to a declared baseline and accounting convention.

E4 – Asymptotic Theorem Evidence.
A theorem is proved uniformly for an encoded family under explicit assumptions and boundary conditions.

E5 – Measured Computational Evidence.
A runtime or memory claim is supported by an implementation, hardware description, software environment, benchmark protocol, and reproducible measurements.

The categories satisfy the non-implications

$$\text{E1} \not\Longrightarrow \text{E4}, \tag{9.58}$$

$$\text{E3} \not\Longrightarrow \text{E5}, \tag{9.59}$$

$$\text{E4} \not\Longrightarrow \text{E5}, \tag{9.60}$$

and

$$\text{E5} \not\Longrightarrow \text{E4}. \tag{9.61}$$

Thus, finite verification does not establish an asymptotic theorem. An operation-count advantage does not establish measured acceleration. An asymptotic theorem does not establish practical wall-clock advantage. Measured performance on a benchmark corpus does not establish a uniform asymptotic theorem. Every result should therefore state its complete evidence profile. The concrete realizations and verification models developed in Chapters 6–8 fall into three principal groups. First, mathematically established results include the isolated full adder and the all-input ripple-adder schema of Chapter 7. The isolated full-adder factorization is verified exhaustively on its complete finite input space and therefore carries E1 finite-verification evidence. The ripple-adder schema family is established by a uniform proof for all n and therefore carries E4 asymptotic theorem evidence for the declared schema task τ_{sch}. Separately, Section 8.6.2 reports preliminary measured implementation evidence for the different fixed-target counting task $\tau_{\text{count},s}$.

Because the exact target sum-bit sequences used in the archived timing runs were not retained, those measurements do not satisfy the strict reproducibility requirement for E5 and are therefore classified below the E5 threshold.

They are not evidence for the schema task τ_{sch}, and they are not combined with the E4 schema theorem into one task-relative evidence profile. Second, the finite graph-coloring realizations provide instance-level evidence. The C_4 realization gives an exact finite raw/admissible/reconstruction calculation. The C_5 realization of Section 6.3.2 provides an explicit stage-by-stage record of structural candidate-interface growth and ordering dependence. These are finite records. Neither establishes a family-level E4 tractability theorem. Third, SAT, CSP, and tensor representations remain theoretically formulated realizations. They provide structural candidate interfaces and reconstruction templates, but no general family-level CPR-tractability theorem is established for these unrestricted classes. The random 3-SAT interface-growth experiment of Chapter 8 provides E2 structural-pattern evidence for the tested instance distribution and orderings only. It does not establish the exact minimum CPR-Width of those instances. In particular, it is a measurement of the selected structural candidate boundary, not a proof of either $\boldsymbol{\omega}_{\text{rec}}$ or $\widehat{\boldsymbol{\omega}}_{\text{rec}}$ for the sampled instances. It therefore does not constitute E4 evidence for random 3-SAT as a family. It is a negative structural observation on the tested candidate interfaces. A disciplined CPR-WIDTH research program may proceed through the following sequence:

1. complete exact mathematical characterization of CPR-Width for selected finite models;
2. implementation of exact reachable-prefix, signature, interface, and semantic-quotient construction;
3. recording BC1–BC4 evidence at the finite instance level;
4. proving BC2 and BC4 uniformly whenever an asymptotic tractability claim is intended;
5. publishing complete component-wise operation-count records;
6. comparing multiple admissible reconstruction orderings while keeping all other conditions fixed;
7. comparing CPR-WIDTH with declared classical baselines;
8. extending finite analysis to uniformly encoded families;
9. deriving formal relations among interface width, semantic-state growth, transition structure, and complete operation count;
10. investigating exact and approximate relations to classical width parameters;
11. determining controlled or constant CPR-Width bounds for additional subclasses;
12. and conducting reproducible wall-clock experiments only after correctness, accounting rules, and output requirements have been fixed.

This sequence prevents finite observations from being interpreted as asymptotic theorems and prevents structural state reduction from being interpreted as a complete runtime proof.

9.5 Final Conclusion

This chapter has combined the complexity-theoretic scope of CPR-WIDTH with its open mathematical, algorithmic, and experimental research program. The central structural-runtime relation remains

$$\boldsymbol{\omega}_{\text{rec}}(P, \pi, \tau; \mathfrak{M}) \longrightarrow S_{\text{CPR}}(P, \pi, \tau; \mathfrak{M}) \longrightarrow N_{\text{CPR}}(P, \pi, \tau; \mathfrak{M}). \tag{9.62}$$

The first arrow expresses the influence of the active reconstruction interface on the raw and semantic state spaces. The second expresses the contribution of semantic-state processing to the complete CPR operation count. Neither arrow is a universal tractability theorem. The principal unresolved questions concern

- efficient construction of favorable orderings;

- exact polynomial-time semantic encodings;
- efficient construction of minimum task-sufficient interfaces;
- controlled or constant subclass-specific CPR-Width bounds;
- identification of nontrivial uniformly CPR-tractable families;
- comparison with classical width parameters;
- complete operation-count validation;
- and reproducible wall-clock benchmarking.

The complete future tractability requirement is not merely

$$\mathrm{BC1} \wedge \mathrm{BC2} \wedge \mathrm{BC3} \wedge \mathrm{BC4}.$$

A polynomial-time conclusion is permitted only after one fixed decision task, one fixed uniform reconstruction model, one deterministic preprocessing algorithm containing its ordering-supply component, and six fixed polynomial component bounds have been established uniformly, together with BC1–BC4. No NP-complete language is proved in this manuscript to belong to

$$\mathbf{P}_{\mathrm{CPR}}.$$

Accordingly, the contribution of CPR-WIDTH is not a proof of

$$\mathbf{P} = \mathbf{NP}.$$

Its contribution is a reconstruction-oriented mathematical language for separating

- active-interface width;
- reachable-state growth;
- semantic-state compression;
- semantic-transition structure;
- transition-processing cost;
- reconstruction cost;
- verification cost;
- output cost;
- operation-count gain;
- and measured runtime gain.

Every conclusion must be limited to the evidence category or evidence profile supporting it. Finite verification supports finite conclusions. Structural pattern evidence supports only the corresponding observed structural pattern. Operation-count evidence supports only the declared accounting comparison. Uniform theorems support only the family and assumptions for which they are proved. Measured computational evidence supports only the specified implementation and experimental environment. Accordingly,

$$\boxed{\begin{array}{c} \text{small CPR-Width} \not\Longrightarrow \text{polynomial runtime}, \\ \text{small semantic quotient} \not\Longrightarrow \text{cheap quotient construction}, \\ \text{finite verification} \not\Longrightarrow \text{uniform tractability}, \\ G_{\mathrm{count}} > 1 \not\Longrightarrow G_{\mathrm{time}} > 1. \end{array}} \tag{9.63}$$

The strongest fully established uniform arithmetic realization in the present manuscript remains the ripple-adder family. For the unrestricted SAT, CSP, graph-coloring, tensor, or NP-complete problem classes, no uniform CPR tractability result is established. Future work must therefore proceed by identifying explicitly defined subclasses for which favorable width, reconstruction-safe semantic compression, uniform constructibility, BC1–BC4, and all six complete polynomial runtime bounds can be proved simultaneously.

A Core Definitions and Dependency Map

This appendix provides a compact reference map for the principal objects of the CPR-WIDTH framework. Detailed definitions, assumptions, and proofs remain in the main text.

A.1 Declared Reconstruction Setting

Let P be a finite problem instance, let τ be the declared computational task, and let $\mathfrak{M}$ be the fixed CPR reconstruction model. An admissible reconstruction ordering satisfies

$$\pi \in \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M}).$$

For reconstruction stage i, the reachable prefix-state space is

$$\mathcal{R}_i^{\pi}(P),$$

and for

$$r \in \mathcal{R}_i^{\pi}(P),$$

the feasible continuation set is denoted by

$$E_i^{\pi}(r).$$

The reconstruction-safe task-relative signature is

$$\mathrm{Sig}_{\tau,i}^{\pi}(r),$$

subject to task preservation, next-value preservation, and successor congruence.

A.2 Definition Dependency Chain

The principal dependency order is

$$\mathcal{R}_i^{\pi}(P) \longrightarrow E_i^{\pi}(r) \longrightarrow \mathrm{Ans}_{\tau,i}^{\pi}(r) \longrightarrow \mathrm{Sig}_{\tau,i}^{\pi}(r) \longrightarrow B_i^{\pi,\tau}.$$

After the active interface has been defined,

$$B_i^{\pi,\tau} \longrightarrow U_i \longrightarrow U_i^{\mathrm{reach}} \longrightarrow \widehat{\mathrm{Sig}}_{\tau,i}^{\pi} \longrightarrow S_i.$$

Literal equality of complete continuation sets is not the general definition of semantic equivalence on interface states.

A.3 Operational and Normalized Width

Let

$$b_i^{\pi,\tau}(P)$$

denote the minimum cardinality of a task-sufficient active interface at stage i. The unnormalized operational width is

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = \max_{0\le i\le n} b_i^{\pi,\tau}(P).$$

The normalized CPR-Width is

$$\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = \max\left\{0, \widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) - 1\right\}.$$

Hence,

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \le \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) + 1.$$

The minimum normalized width over admissible orderings is

$$\boldsymbol{\omega}_{\mathrm{rec}}(P,\tau;\mathfrak{M}) = \min_{\pi\in\Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})} \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}).$$

The corresponding minimum unnormalized width is defined analogously.

A.4 Three Distinct State Levels

At every stage, three state levels must remain distinct:

$$U_i, \qquad U_i^{\mathrm{reach}}, \qquad S_i.$$

Here,

$$U_i$$

is the raw interface-state space,

$$U_i^{\mathrm{reach}} \subseteq U_i$$

contains only reachable interface states, and

$$S_i = U_i^{\mathrm{reach}} / \equiv_{P,\pi,i}^{\tau,\mathfrak{M}}$$

is the semantic quotient. The semantic equivalence relation is defined by

$$u \equiv_{P,\pi,i}^{\tau,\mathfrak{M}} v \quad \Longleftrightarrow \quad \widehat{\mathrm{Sig}}_{\tau,i}^{\pi}(u) = \widehat{\mathrm{Sig}}_{\tau,i}^{\pi}(v).$$

A.5 Width–State–Runtime Dependency

Let

$$q(P) = \max_{v \in V(P)} |D_v|,$$

with the empty-component convention specified in Chapter 3. Then

$$S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) = \max_i |S_i|$$

satisfies

$$S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) \leq q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}$$

and therefore

$$S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) \leq q(P)^{\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}.$$

Thus the structural dependency is

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) \longrightarrow S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) \longrightarrow N_{\mathrm{CPR}}.$$

A.6 Complete Runtime Decomposition

The complete CPR operation count is

$$N_{\mathrm{CPR}} = N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} + N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}.$$

No single component may be omitted when polynomial-time tractability is claimed.

A.7 Task Relativity

CPR-Width is task-relative. For the same instance P, different declared tasks may induce different reconstruction-safe signatures, different task-sufficient interfaces, and therefore different values of

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M})$$

and

$$\boldsymbol{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}).$$

A.8 Consistency and Audit Map

The following distinctions must be preserved throughout the manuscript:

$$\widetilde{B}_i^{\pi,\tau} \neq B_i^{\pi,\tau} \quad \text{in general,}$$

where the former is a structural candidate interface and the latter is cardinality-minimal and task-sufficient. Likewise,

$$U_i \supseteq U_i^{\mathrm{reach}} \longrightarrow S_i$$

separates raw, reachable, and semantic states. Finally,

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M})$$

is the unnormalized operational quantity, whereas

$$\boldsymbol{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M})$$

is the normalized CPR-Width.

A.9 Interpretive Summary

The CPR-WIDTH framework separates three questions:

How much information must remain active?

How many semantically distinct states remain?

How much computation is required?

These correspond respectively to width, semantic-state complexity, and runtime. A small value of

$$\boldsymbol{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M})$$

is therefore a structural statement. Polynomial-time tractability additionally requires the uniform boundary conditions and runtime bounds established in Chapter 5.

B Core Definition Register

This appendix provides a compact register of the principal objects of the CPR-WIDTH framework. Complete definitions, assumptions, and proofs remain in the main text.

Table B.1: Core definitions of CPR-WIDTH.

Object	Definition / Meaning
Finite component set	$V(P) = \{v_1, \ldots, v_n\}$. Finite reconstruction components of P.
Declared CPR reconstruction model	$\mathfrak{M}(P, \tau) = (\mathrm{Adm}_{P,\tau}, \mathcal{R}, \mathrm{Sig}_\tau)$. It specifies admissibility, reachable reconstruction behavior, and the reconstruction-safe signature family.
Admissible ordering	$\pi \in \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M})$. A reconstruction ordering admissible for the declared task and model.
Processed prefix	$V_i^\pi = \{v_{\pi(1)}, \ldots, v_{\pi(i)}\}$. Components processed through stage i.
Unresolved suffix	$\overline{V}_i^\pi = V(P) \setminus V_i^\pi$. Components not yet processed.
Reachable prefix-state space	$\mathcal{R}_i^\pi(P)$. Prefix assignments reachable under the declared reconstruction model.
Feasible continuation set	$E_i^\pi(r)$. Complete feasible continuations of a reachable prefix state $r \in \mathcal{R}_i^\pi(P)$.
Task-relative future value	$\mathrm{Ans}_{\tau,i}^\pi(r)$. Task value induced by the reachable prefix state r.
Admissible next-value set	$\Lambda_i^\pi(r)$. Values producing reachable successor prefix states.
Prefix transition	$\delta_i^\pi(r, a)$. Reachable successor obtained from admissible next value a.
Reconstruction-safe signature	$\mathrm{Sig}_{\tau,i}^\pi(r)$. Task-relative signature satisfying S1 task preservation, S2 next-value preservation, and S3 successor congruence.
Candidate interface	$\widetilde{B}_i^{\pi,\tau}$. Structural candidate interface; it need not be cardinality-minimal.
Minimum task-sufficient interface	$B_i^{\pi,\tau}$. Cardinality-minimal active interface sufficient for the declared task.
Minimum interface size	$b_i^{\pi,\tau}(P) = \lvert B_i^{\pi,\tau}\rvert$. Minimum task-sufficient interface cardinality at stage i.
Unnormalized operational width	$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = \max_i b_i^{\pi,\tau}(P)$.
Normalized CPR-Width	$\boldsymbol{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = \max\{0, \widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) - 1\}$.
Minimum normalized CPR-Width	$\boldsymbol{\omega}_{\mathrm{rec}}(P, \tau; \mathfrak{M}) = \min_{\pi \in \Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})} \boldsymbol{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M})$.
Raw interface-state space	U_i. All syntactically possible assignments to the active interface.
Reachable interface-state space	$U_i^{\mathrm{reach}} \subseteq U_i$. Interface states induced by reachable prefix states.
Induced interface signature	$\widehat{\mathrm{Sig}}_{\tau,i}^\pi(u)$. Reconstruction-safe task signature induced by reachable interface state u.
Semantic equivalence	$u \equiv_{P,\pi,i}^{\tau,\mathfrak{M}} v \iff \widehat{\mathrm{Sig}}_{\tau,i}^\pi(u) = \widehat{\mathrm{Sig}}_{\tau,i}^\pi(v)$. Literal continuation-set equality is not the general definition.
Semantic quotient	$S_i = U_i^{\mathrm{reach}} / \equiv_{P,\pi,i}^{\tau,\mathfrak{M}}$.
Maximum semantic-state count	$S_{\mathrm{CPR}} = \max_i \lvert S_i\rvert$.
Total semantic-state volume	$V_{\mathrm{sem}} = \sum_i \lvert S_i\rvert$.
Semantic transition count	E_{sem}. Total number of reachable semantic transition edges.
Maximum local domain size	$q(P) = \max_{v \in V(P)} \lvert D_v\rvert$. The empty-component convention is specified in Chapter 3.
Width–state bound	$S_{\mathrm{CPR}} \leq q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} \leq q(P)^{\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}$.

Continued on the next page

Table B.1 – continued from previous page

Object	Definition / Meaning
Complete CPR operation count	$N_{\mathrm{CPR}} = N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} + N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}$.
Post-transition cost	$R_{\mathrm{post}} = N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}$.
Count-level gain	$G_{\mathrm{count}} = N_{\mathrm{Classic}}/N_{\mathrm{CPR}}$.
Measured runtime gain	$G_{\mathrm{time}} = T_{\mathrm{Classic}}/T_{\mathrm{CPR}}$.
CPR tractability	BC1–BC4 together with fixed polynomial bounds for all six runtime components.

The register is descriptive rather than constitutive. Finite verification, small CPR-Width, or a small semantic quotient alone does not establish family-level polynomial-time tractability.

C Proof Details

This appendix collects proof details for selected statements from the main text.

C.1 Interface Bound

By definition,

$$\widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = \max_{0 \le i \le n} b_i^{\pi,\tau}(P).$$

Hence, for every stage i,

$$b_i^{\pi,\tau}(P) \le \widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}).$$

For every selected cardinality-minimal task-sufficient interface,

$$|B_i^{\pi,\tau}| = b_i^{\pi,\tau}(P),$$

and therefore

$$|B_i^{\pi,\tau}| \le \widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}).$$

Since

$$\omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = \max\left\{0, \widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) - 1\right\},$$

one also has

$$\widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) \le \omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) + 1.$$

Consequently,

$$|B_i^{\pi,\tau}| \le \omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) + 1.$$

C.2 Raw State Bound

By definition,

$$U_i(P, \pi, \tau; \mathfrak{M}) = \prod_{v \in B_i^{\pi,\tau}} D_v.$$

Therefore,

$$|U_i(P, \pi, \tau; \mathfrak{M})| = \prod_{v \in B_i^{\pi,\tau}} |D_v|.$$

With

$$q(P) = \max_{v \in V(P)} |D_v|,$$

where $q(P)$ is defined as in Chapter 3, including the empty-component convention $q(P) = 1$ when $V(P) = \varnothing$, each factor satisfies

$$|D_v| \le q(P).$$

Hence,

$$|U_i(P,\pi,\tau;\mathfrak{M})| \le q(P)^{|B_i^{\pi,\tau}|}.$$

Using the interface bound,

$$|U_i(P,\pi,\tau;\mathfrak{M})| \le q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}.$$

Since $q(P) \ge 1$ and

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \le \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) + 1,$$

it follows that

$$|U_i(P,\pi,\tau;\mathfrak{M})| \le q(P)^{\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}.$$

C.3 Reachable State Bound

The reachable interface-state space satisfies

$$U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}) \subseteq U_i(P,\pi,\tau;\mathfrak{M}).$$

Therefore,

$$|U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})| \le |U_i(P,\pi,\tau;\mathfrak{M})|.$$

Using the raw-state bound,

$$|U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})| \le q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}.$$

Consequently, since $q(P) \ge 1$,

$$|U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})| \le q(P)^{\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}.$$

Thus, reachability filtering can reduce, but never increase, the width-based state bound.

C.4 Semantic-State Bound

Since the semantic quotient is obtained from the reachable interface-state space,

$$|S_i(P,\pi,\tau;\mathfrak{M})| \le |U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})|.$$

Using the reachable-state bound,

$$|S_i(P,\pi,\tau;\mathfrak{M})| \le q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}.$$

Since

$$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \le \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) + 1,$$

and $q(P) \ge 1$, it follows that

$$|S_i(P,\pi,\tau;\mathfrak{M})| \le q(P)^{\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}.$$

Therefore,

$$S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) = \max_i |S_i(P,\pi,\tau;\mathfrak{M})|$$

satisfies

$$S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \le q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} \le q(P)^{\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}.$$

The complete CPR operation count is

$$N_{\mathrm{CPR}} = N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} + N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}.$$

Assume fixed polynomial bounds

$$N_j(P,\pi,\tau;\mathfrak{M}) \le p_j(\ell(P))$$

for every

$$j \in \{\mathrm{Build}, \mathrm{Reduce}, \mathrm{Transition}, \mathrm{Reconstruct}, \mathrm{Verify}, \mathrm{Output}\}.$$

Then

$$N_{\mathrm{CPR}} \le \sum_j p_j(\ell(P)) = \mathrm{poly}(\ell(P)).$$

This conclusion requires polynomial control of all six runtime components. It is not implied by small CPR-Width alone. It is also not implied by BC2 or BC4 alone: BC2 controls only the declared preprocessing work, while BC4 controls semantic-state and semantic-transition cardinalities rather than the complete execution cost.

C.5 Quotient Preservation

Suppose

$$u \equiv^{\tau,\mathfrak{M}}_{P,\pi,i} v.$$

By definition,

$$\widehat{\mathrm{Sig}}^{\pi}_{\tau,i}(u) = \widehat{\mathrm{Sig}}^{\pi}_{\tau,i}(v).$$

For reachable prefix representatives $r, r' \in \mathcal{R}^{\pi}_i(P)$ of u and v, respectively, the induced-signature factorization therefore gives

$$\mathrm{Sig}^{\pi}_{\tau,i}(r) = \mathrm{Sig}^{\pi}_{\tau,i}(r').$$

The reconstruction-safety conditions S1–S3 imply

$$\mathrm{Ans}^{\pi}_{\tau,i}(r) = \mathrm{Ans}^{\pi}_{\tau,i}(r'),$$

$$\Lambda^{\pi}_i(r) = \Lambda^{\pi}_i(r'),$$

and, for every common admissible next value a,

$$\mathrm{Sig}^{\pi}_{\tau,i+1}\left(\delta^{\pi}_i(r,a)\right) = \mathrm{Sig}^{\pi}_{\tau,i+1}\left(\delta^{\pi}_i(r',a)\right).$$

Hence semantic quotienting preserves the declared task value, admissible next values, and successor semantic classes. Equality of literal continuation sets is not required in general. For enumeration or another task whose declared signature explicitly preserves complete continuation information, literal continuation equality may arise as a stronger special case, but it is not the general CPR semantic-equivalence definition. Thus quotienting by

$$\equiv^{\tau,\mathfrak{M}}_{P,\pi,i}$$

preserves exactly the task-relative and transition-relative behavior certified by conditions S1–S3.

D Audit Checklist

Table D.1 summarizes the minimum checks required for a CPR-WIDTH analysis. The complete definitions and proofs are given in Chapters 2–5 and Appendices A–C.

Certification rule. A family-level CPR tractability claim is certified only when BC1–BC4 hold uniformly and all six runtime components admit fixed polynomial bounds in the input encoding length.

Table D.1: CPR-WIDTH audit checklist.

Audit item	**Required check**
Problem instance	Verify that P is a finite encoded problem instance and that its reconstruction-component set $V(P)$ and local domains D_v are declared.
Computational task	Verify that the computational task τ is stated explicitly. All task-sufficiency and semantic-equivalence claims are relative to this declared task.
Reconstruction model	Verify that a fixed CPR reconstruction model $\mathfrak{M}$ is declared and that its admissibility, reachability, and signature rules are specified.
Admissible ordering	Verify that $$\pi \in \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M}).$$ The ordering must be admissible under the declared model and task.
Reachable prefix states	Verify that the reachable prefix-state spaces $$\mathcal{R}_i^{\pi}(P)$$ are defined independently of the later semantic quotient construction.
Continuation information	Verify that $$E_i^{\pi}(r)$$ is defined only for reachable prefix states $$r \in \mathcal{R}_i^{\pi}(P),$$ and that continuation information is not used as an undefined property of an arbitrary interface state.
Next-value structure	Verify that the admissible next-value set $$\Lambda_i^{\pi}(r)$$ and the transition map $$\delta_i^{\pi}(r, a)$$ are defined before successor-congruence claims are made.
Reconstruction-safe signature	Verify that $$\mathrm{Sig}_{\tau,i}^{\pi}(r)$$ satisfies the three reconstruction-safety requirements: S1 task preservation, S2 next-value preservation, S3 successor congruence.
Candidate interface	Verify that a structural candidate interface $$\widetilde{B}_i^{\pi,\tau}$$ is not automatically identified with the minimum task-sufficient interface.
Minimum interface	Verify that $$B_i^{\pi,\tau}$$ is task-sufficient and cardinality-minimal among admissible active interfaces at reconstruction stage i.

Continued on the next page

Table D.1 – continued from previous page

Audit item	Required check
Minimum interface size	Verify $$b_i^{\pi,\tau}(P) = \lvert B_i^{\pi,\tau}\rvert.$$
Unnormalized operational width	Verify $$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = \max_i b_i^{\pi,\tau}(P).$$
Normalized CPR-Width	Verify $$\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = \max\left\{0, \widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) - 1\right\}.$$
Width relation	Verify $$\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \le \boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) + 1.$$
Raw interface states	Verify that U_i denotes the raw interface-state space and that it is not confused with the reachable or semantic state spaces.
Reachable interface states	Verify $$U_i^{\mathrm{reach}} \subseteq U_i.$$
Induced interface signature	Verify that $$\widehat{\mathrm{Sig}}_{\tau,i}^{\pi}(u)$$ is well-defined for every reachable interface state $$u \in U_i^{\mathrm{reach}}.$$
Semantic equivalence	Verify $$u \equiv_{P,\pi,i}^{\tau,\mathfrak{M}} v \quad\Longleftrightarrow\quad \widehat{\mathrm{Sig}}_{\tau,i}^{\pi}(u) = \widehat{\mathrm{Sig}}_{\tau,i}^{\pi}(v).$$ Literal equality of complete continuation sets is not the general definition of CPR semantic equivalence.
Semantic quotient	Verify $$S_i = U_i^{\mathrm{reach}} / \equiv_{P,\pi,i}^{\tau,\mathfrak{M}}.$$
Maximum semantic-state count	Verify $$S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) = \max_i \lvert S_i\rvert.$$
Semantic-state volume	Verify $$V_{\mathrm{sem}} = \sum_i \lvert S_i\rvert.$$ This quantity is distinct from the maximum stage-wise count S_{CPR}.
Semantic transition count	Verify that E_{sem} denotes the total number of reachable semantic transition edges and is not identified with the runtime cost $N_{\mathrm{Transition}}$.
Local domain bound	Verify the declared value of $$q(P) = \max_{v \in V(P)} \lvert D_v\rvert,$$ with the empty-component convention specified in Chapter 3.

Continued on the next page

Table D.1 – continued from previous page

Audit item	**Required check**
Width–state bound	Verify $$S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \le q(P)^{\widehat{\boldsymbol{\omega}}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} \le q(P)^{\boldsymbol{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}.$$
Runtime decomposition	Verify the complete six-component operation count $$N_{\mathrm{CPR}} = N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} + N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}.$$ No component may be omitted from a complete tractability claim.
BC1	Verify explicit and uniform structural representation of the declared CPR reconstruction model.
BC2	Verify polynomial uniform constructibility, including construction of the admissible ordering when that ordering is not supplied as part of the input.
BC3	Verify exact task-relative soundness and completeness of the declared reconstruction procedure.
BC4	Verify fixed polynomial bounds on both $$V_{\mathrm{sem}}$$ and $$E_{\mathrm{sem}}$$ for the encoded problem family.
Build cost	Verify a fixed polynomial bound for $$N_{\mathrm{Build}}.$$
Reduce cost	Verify a fixed polynomial bound for $$N_{\mathrm{Reduce}}.$$
Transition cost	Verify a fixed polynomial bound for $$N_{\mathrm{Transition}}.$$
Reconstruction cost	Verify a fixed polynomial bound for $$N_{\mathrm{Reconstruct}}.$$
Verification cost	Verify a fixed polynomial bound for $$N_{\mathrm{Verify}}.$$
Output cost	Verify a fixed polynomial bound for $$N_{\mathrm{Output}}.$$
Uniformity	Verify that the reconstruction model, preprocessing procedure, boundary conditions, and polynomial bounds are fixed uniformly for the entire encoded problem family rather than chosen separately for each individual instance.
Evidence status	Distinguish finite verification, measured implementation evidence, structural family evidence, and formal family-level complexity certification.
Finite-instance scope	Verify that exhaustive or empirical results for finitely many instances are not presented as proofs of uniform family-level tractability.

Continued on the next page

Table D.1 – continued from previous page

Audit item	Required check
Candidate-width evidence	Verify that a measured structural candidate interface is not reported as exact CPR-Width unless task-sufficiency and minimum cardinality have been established.
Runtime evidence	Keep operation-count comparisons and measured wall-clock comparisons separate. A measured speedup is not itself a complexity-theoretic result.
NP-complete scope	For an NP-complete language L^*, verify that any statement involving $$L^* \in \mathbf{P}_{\mathrm{CPR}}$$ is explicitly conditional. The framework does not by itself establish such membership.
Final certification	A family-level CPR tractability claim requires BC1–BC4 uniformly and fixed polynomial bounds for all six runtime components in the input encoding length.

A successful audit establishes consistency with the declared CPR-WIDTH framework. It does not replace the problem-specific proof obligations required for a concrete reconstruction model or problem family.

E Figure Register

This appendix records the scientific figures used in the manuscript. The scientific figures registered here are generated from the manuscript source files and are intended as visual summaries of the corresponding mathematical constructions. Their captions, numbering, and page references should therefore be checked against the compiled manuscript whenever the layout or figure implementation is revised.

Table E.1: Figure Register for CPR-WIDTH.

Figure	Page	Description
Figure 2.1	11	Chapter 2. Stage-by-stage evolution of the active reconstruction interface $B_i^{\pi,\tau}$ on the running example (P_4, π, τ), tracking active, processed, and unresolved vertices.
Figure 2.2	11	Chapter 2. Semantic-runtime architecture of CPR-WIDTH: $$\boldsymbol{\omega}_{\mathrm{rec}} \longrightarrow S_{\mathrm{CPR}} \longrightarrow N_{\mathrm{CPR}}.$$ The diagram summarizes the structural dependency from reconstruction width through semantic-state growth to the complete operation-count model.
Figure 3.1	15	Chapter 3. Controlled partial reconstruction and semantic-state compression: raw interface states, reachable interface states, and semantic quotient states.
Figure 4.1	20	Chapter 4. Complete runtime framework of CPR-WIDTH: the six disjoint operation-count components N_{Build}, N_{Reduce}, $N_{\mathrm{Transition}}$, $N_{\mathrm{Reconstruct}}$, N_{Verify}, and N_{Output}, summing to the complete operation count N_{CPR}.

Bibliography

[1] Hans L. Bodlaender. "A Partial k-Arboretum of Graphs with Bounded Treewidth". In: *Theoretical Computer Science* 209.1–2 (1998), pp. 1–45.

[2] Bruno Courcelle and Stephan Olariu. "Upper bounds to the clique width of graphs". In: *Discrete Applied Mathematics* 101.1–3 (2000), pp. 77–114. DOI: 10.1016/S0166-218X(99)00184-5.

[3] Marek Cygan et al. *Parameterized Algorithms*. Springer, 2015.

[4] Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Springer, 2013.

[5] Nancy G. Kinnersley. "The Vertex Separation Number of a Graph Equals Its Path-Width". In: *Information Processing Letters* 42.6 (1992), pp. 345–350. DOI: 10.1016/0020-0190(92)90234-M.

[6] David Mitchell, Bart Selman, and Hector Levesque. "Hard and Easy Distributions of SAT Problems". In: *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92)*. 1992, pp. 459–465.

[7] Neil Robertson and Paul D. Seymour. “Graph Minors. II. Algorithmic Aspects of Tree-Width”. In: *Journal of Algorithms* 7.3 (1986), pp. 309–322.

[8] Neil Robertson and Paul D. Seymour. “Graph Minors. X. Obstructions to Tree-Decomposition”. In: *Journal of Combinatorial Theory, Series B* 52.2 (1991), pp. 153–190. DOI: 10.1016/0095-8956(91)90061-N.