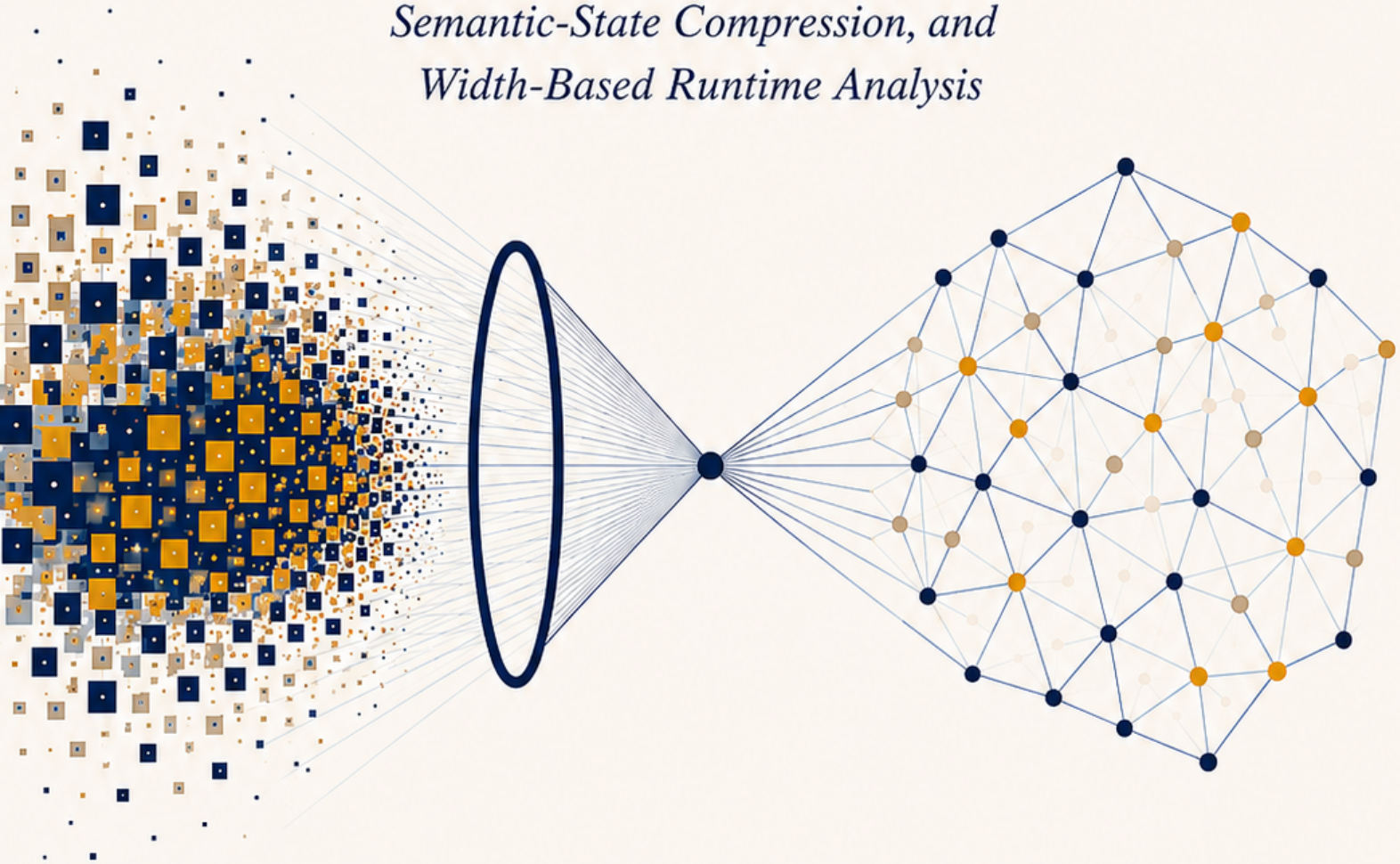
CPR-WIDTH

Controlled Partial Reconstruction Width

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



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

REZA HESAMIY

MUNICH, GERMANY

2026

CPR-WIDTH-V2.0

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-V2.0

Munich, 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. The unnormalized CPR-Width is defined from the maximum cardinality of this interface over the complete reconstruction process and is minimized over admissible orderings; a fixed offset convention then yields the normalized CPR-Width ω_{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 $\omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) \rightarrow S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) \rightarrow N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M})$ (with τ and $\mathfrak{M}$ suppressed when fixed) connecting reconstruction width, semantic-state growth, 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 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, while a uniform family-level result is established for the ripple-adder family.

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

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 is often not the quantity that determines the essential computational difficulty.

During a reconstruction process, previously processed information gradually loses its relevance. At every reconstruction stage, some information may safely be discarded because it can no longer influence any admissible continuation, whereas other information must remain available until reconstruction has been completed. The mathematical distinction between relevant and irrelevant information is therefore fundamental.

The present manuscript studies the amount of information that must remain simultaneously available during a sound and complete reconstruction. Rather than measuring the size of the complete search space, the theory developed here measures the size of the active reconstruction interface. This quantity is called the *Controlled Partial Reconstruction Width (CPR-WIDTH)*.

Unlike many classical structural parameters, CPR-WIDTH is not defined directly on graphs or decomposition trees. Classical graph-width parameters such as treewidth and pathwidth relate structural restrictions of graphs to algorithmic tractability [5, 1]. Parameterized complexity extends this perspective by separating the influence of the input size from the influence of additional structural parameters [3, 2]. CPR-WIDTH is introduced in relation to this line of work, but it is not defined as a graph-decomposition parameter. Rather, it is a reconstruction-oriented width notion whose value depends on the active information maintained during a declared reconstruction process. A formal structural comparison between CPR-Width and classical width parameters is developed as an explicit research problem in Chapter 9. In particular, any valid comparison must fix the underlying representation, computational task, admissible reconstruction model, and ordering convention before inequalities or equivalences between the corresponding width notions can be meaningfully formulated.

The framework is also related in spirit to dynamic programming and state-space reduction, since all three approaches exploit restricted information carried between computational stages. CPR-WIDTH, however, does not identify its structural parameter with the size of a decomposition bag, a dynamic-programming table, or a complete search frontier. Its defining object is the minimum task-sufficient active interface under a declared reconstruction model, while the induced reachable and semantic state spaces are analyzed separately. Formal equivalences or separations from established dynamic-programming and state-space parameters therefore require additional model-specific results and are not assumed in this work.

The value of CPR-WIDTH depends on the chosen reconstruction ordering, the active reconstruction interface, the declared computational task, and the semantic information required for every admissible future continuation. Consequently, two reconstruction procedures operating on the same underlying combinatorial structure may possess different reconstruction widths whenever their active interfaces differ.

The objective of this manuscript is therefore not to replace existing complexity theory, but to introduce a reconstruction-oriented structural parameter that permits a mathematical analysis of active information, semantic-state growth, and reconstruction complexity. The theory developed in the following chapters proceeds through three successive levels: first, the active reconstruction interface is introduced as the central structural object; second, the reachable semantic-state space generated by this interface is studied; finally, the resulting computational effort required by controlled reconstruction is analyzed.

The central mathematical relationship developed throughout this manuscript is

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

The first quantity is the normalized CPR-Width, $\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = \max\{0, \widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) - 1\}$, obtained from the unnormalized width $\widehat{\omega}_{\mathrm{rec}}$ by the fixed offset convention of Chapter 2; it is $\widehat{\omega}_{\mathrm{rec}}$, not ω_{rec} itself, that equals the maximal active reconstruction interface cardinality directly. The second quantity describes the corresponding semantic-state space, and the third represents the complete computational effort required by the declared reconstruction model. The arrows express structural and accounting dependencies rather than implications of polynomial-time tractability. In particular, a small reconstruction width alone does not imply a small complete runtime. The formal construction of these quantities is developed progressively in Chapters 2–4.

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. Instead, the objective is to develop a mathematically rigorous theory of Controlled Partial Reconstruction Width together with its structural, semantic, and computational consequences.

The remaining chapters develop this theory systematically. Chapter 2 introduces the declared reconstruction model, reconstruction-safe signatures, task-sufficient active interfaces, and the formal definition of CPR-WIDTH. Chapter 3 develops the corresponding raw, reachable, and semantic-state framework. Chapter 4 establishes the complete runtime and operation-count model. Chapter 5 formulates the boundary conditions required for uniform CPR tractability. Chapter 6 develops representative structural realizations and the theory of controlled and constant CPR-Width, including concrete reconstruction examples. Chapter 7 develops a uniform arithmetic reconstruction model for the ripple-adder family. Chapter 8 provides finite verification records together with the measured runtime and interface-size illustrations included in the present work. Chapter 9 places these results within their complexity-theoretic scope and formulates the principal mathematical, algorithmic, comparative, and experimental open problems.

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:

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

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. 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, 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 $\tau_{\text{witness}} \neq \tau_{\text{dec}}$ on the same instance (P_4, π) is meant: its task-sufficient interfaces and unnormalized width coincide with the decision variant tracked below, because both share the same future-admissibility structure, but its terminal signature must retain enough information to reconstruct the complete coloring rather than collapse to a constant value, and its reconstruction cost $N_{\text{Reconstruct}}$ is correspondingly nonzero where the decision variant's is not.

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

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

where $\mathrm{Adm}_{P,\tau}$ is the admissibility predicate for reconstruction orderings, $\mathfrak{R}$ specifies the reachable prefix-state spaces $\mathcal{R}_i^\pi(P)$ of Section 2.4 for every admissible π and stage i (the calligraphic $\mathcal{R}_i^\pi(P)$ denotes one resulting reachable prefix-state space; the blackletter $\mathfrak{R}$ denotes the model-level specification producing all of them), Λ and δ denote the admissible-next-value maps Λ_i^π and the transition maps δ_i^π of Section 2.6 — the canonical maps induced by $\mathfrak{R}$ once $\mathfrak{R}$ and π are fixed, and therefore no additional degree of freedom beyond $\mathfrak{R}$ — and Sig_τ specifies a reconstruction-safe, task-relative future-signature family.

In the component-representation setting developed in this chapter, $\mathfrak{R}$ is thus the only primitively specified data among Λ, δ, and $\mathfrak{R}$; Λ and δ are listed as separate components of $\mathfrak{M}(P, \tau)$ only for readability, since a reconstruction model is more naturally described stage by stage through admissible next values and transitions than through the reachable-state spaces alone.

All widths in this chapter are relative to the fixed problem representation, the declared task, and the declared reconstruction model.

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 $\mathrm{Adm}_{P,\tau}(\pi) = 1$. 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

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

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 $V_i^\pi = \{v_{\pi(1)}, \ldots, v_{\pi(i)}\}$, $0 \leq i \leq n$, and the unresolved suffix is $\overline{V}_i^\pi = V(P) \backslash V_i^\pi$, 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 $\delta_i^\pi(r, a) = r \cup \{w_i^\pi \mapsto a\} \in \mathcal{R}_{i+1}^\pi(P)$.

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 $\mathrm{Sig}_{\tau,i}^\pi : \mathcal{R}_i^\pi(P) \to \Sigma_{\tau,i}^\pi$, $0 \leq i \leq n$, 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 $\mathrm{Sig}_{\tau,i+1}^\pi(\delta_i^\pi(r, a)) = \mathrm{Sig}_{\tau,i+1}^\pi(\delta_i^\pi(r', a))$.

Condition S1 preserves the declared task-relative future value. Conditions S2 and S3 guarantee that replacing a prefix state by another state with the same signature does not make the next

reconstruction step ambiguous. The identity signature $\mathrm{Sig}^{\pi}_{\tau,i}(r) = r$ is always reconstruction-safe, so a reconstruction-safe future-signature family always exists, although it may provide no compression. A constant signature is admissible only when it satisfies S1–S3; consequently, the formal model cannot collapse every state to one signature unless task values and transition behavior are genuinely indistinguishable.

2.7.1 Signature-Update Procedure

Conditions S1–S3 constrain which signature families are admissible, but they do not by themselves say how $\mathrm{Sig}^{\pi}_{\tau,i+1}(\delta^{\pi}_i(r,a))$ is computed from $\mathrm{Sig}^{\pi}_{\tau,i}(r)$ once a next value a is chosen. Two computation strategies are possible in principle.

Brute-force recomputation evaluates $\mathrm{Sig}^{\pi}_{\tau,i+1}$ on the full successor state $\delta^{\pi}_i(r,a) = r \cup \{w^{\pi}_i \mapsto a\}$ 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

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

namely $\mathrm{upd}_i : \mathcal{D}^{\pi,\tau}_i \to \Sigma^{\pi}_{\tau,i+1}$ satisfying $\mathrm{upd}_i(\mathrm{Sig}^{\pi}_{\tau,i}(r), a) = \mathrm{Sig}^{\pi}_{\tau,i+1}(\delta^{\pi}_i(r,a))$ for every reachable r and every $a \in \Lambda^{\pi}_i(r)$; condition S3 guarantees this is well defined on $\mathcal{D}^{\pi,\tau}_i$, since the right-hand side depends on r only through $\mathrm{Sig}^{\pi}_{\tau,i}(r)$. Condition S3 makes no claim about pairs $(s,a) \notin \mathcal{D}^{\pi,\tau}_i$; upd_i need not, and in general does not, extend beyond $\mathcal{D}^{\pi,\tau}_i$. By condition S2, the admissible next-value set $\Lambda^{\pi}_i(r)$ likewise depends on r only through $\mathrm{Sig}^{\pi}_{\tau,i}(r)$; the pseudocode below therefore writes $\Lambda^{\pi}_i(s)$ for $s = \mathrm{Sig}^{\pi}_{\tau,i}(r)$ 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^{π}_i 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. Chapter 7 exhibits a concrete instance in which upd_i is genuinely $O(1)$: the ripple-adder carry signature (Proposition 7.3, Carry Signature Is Reconstruction-Safe) updates by a single Boolean majority computation on the incoming carry and the two current input bits, independent of the number of bits already processed, in contrast to brute-force recomputation, which would re-examine the entire processed prefix at every stage.

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

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

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

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

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

The normalized width ω_{rec} is an exponent-oriented analytical parameter; the actual maximum number of simultaneously active components is $\widehat{\omega}_{\mathrm{rec}}$:

$$\boxed{\widehat{\omega}_{\mathrm{rec}} = \text{physical interface cardinality}, \qquad \omega_{\mathrm{rec}} = \text{normalized analytical width.}}$$

Both forms are used consistently throughout this manuscript, for two distinct purposes. The unnormalized width $\widehat{\omega}_{\mathrm{rec}}$ is the operational quantity: it equals the actual interface cardinality and is the form used in every state-count and runtime bound of Chapters 3 and 4. The normalized width $\omega_{\mathrm{rec}} = \max\{0, \widehat{\omega}_{\mathrm{rec}} - 1\}$ is the quantity used when CPR-Width is reported as a width parameter. The minus-one normalization is adopted as a convention analogous to several classical

width parameters whose width is defined as a maximum bag or interface cardinality minus one; no formal identity or comparison with any specific classical width parameter is asserted by this convention. Every occurrence of ω_{rec} in this manuscript, in particular in the boundary conditions of Chapter 5 and the tractability statements of Chapter 9, refers to this normalized quantity; every underlying cardinality bound is proved for $\widehat{\omega}_{\mathrm{rec}}$ first and then restated for ω_{rec} using the identity $\widehat{\omega}_{\mathrm{rec}} = \omega_{\mathrm{rec}} + 1$ for $\widehat{\omega}_{\mathrm{rec}} \geq 1$. Accordingly, $\omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = 0$ may still mean that one component must remain active. When τ and $\mathfrak{M}$ are fixed, the shorter forms $\widehat{\omega}_{\mathrm{rec}}(P, \pi)$ and $\omega_{\mathrm{rec}}(P, \pi)$ 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 $\widehat{\omega}_{\mathrm{rec}}(P_4, \pi, \tau; \mathfrak{M}) = 1$ and $\omega_{\mathrm{rec}}(P_4, \pi, \tau; \mathfrak{M}) = \max\{0, 1-1\} = 0$. 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, $b_i^{\pi,\tau}(P) \leq \widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M})$. Moreover, $\widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) \leq \omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M})+1$. Consequently, for every selected minimum active interface, $|B_i^{\pi,\tau}| \leq \omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) + 1$.*

Proof. The first inequality follows immediately from $\widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = \max_{0 \leq j \leq n} b_j^{\pi,\tau}(P)$. Let $M = \widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M})$. If $M = 0$, then $M = 0 \leq 1 = \omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) + 1$. If $M \geq 1$, then $\omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = M - 1$, and therefore $M = \omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) + 1$. 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

$$\omega_{\mathrm{rec}}(P, \tau; \mathfrak{M}) = \min_{\pi \in \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M})} \omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}).$$

The corresponding unnormalized minimum is $\widehat{\omega}_{\mathrm{rec}}(P, \tau; \mathfrak{M}) = \min_{\pi \in \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M})} \widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M})$. When the task and reconstruction model are fixed, these quantities may be written as $\omega_{\mathrm{rec}}(P)$ and $\widehat{\omega}_{\mathrm{rec}}(P)$. 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 $\omega_{\mathrm{rec}}(P, \pi_1, \tau; \mathfrak{M}) \neq \omega_{\mathrm{rec}}(P, \pi_2, \tau; \mathfrak{M})$.*

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. The reconstruction-safe signature records whether a feasible complete continuation exists, the set of admissible next values, and the signatures of the corresponding successor states.

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 $\widehat{\omega}_{\mathrm{rec}}(P, \pi_1, \mathrm{dec}; \mathfrak{M}) = 2$ and $\omega_{\mathrm{rec}}(P, \pi_1, \mathrm{dec}; \mathfrak{M}) = 1$.

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 $\widehat{\omega}_{\mathrm{rec}}(P, \pi_2, \mathrm{dec}; \mathfrak{M}) = 1$ and $\omega_{\mathrm{rec}}(P, \pi_2, \mathrm{dec}; \mathfrak{M}) = 0$.

Therefore $\omega_{\mathrm{rec}}(P, \pi_1, \mathrm{dec}; \mathfrak{M}) \neq \omega_{\mathrm{rec}}(P, \pi_2, \mathrm{dec}; \mathfrak{M})$. □

2.16 Structural Interpretation

Controlled Partial Reconstruction Width does not measure the total size of the complete candidate space. The unnormalized quantity $\widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M})$ measures the minimum maximal number of processed components whose values must remain simultaneously available under the fixed representation, task, transition rules, and reconstruction-safe signature family; the normalized quantity $\omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M})$ is obtained from it by the minus-one convention explained above. A task-sufficient interface satisfies $r|_B = r'|_B \Rightarrow \mathrm{Sig}_{\tau,i}^{\pi}(r) = \mathrm{Sig}_{\tau,i}^{\pi}(r')$; equivalently, the future-signature map factors through the interface projection, $\mathrm{Sig}_{\tau,i}^{\pi} = \overline{\mathrm{Sig}}_{\tau,i,B}^{\pi} \circ \rho_{i,B}^{\pi}$. This factorization is the formal statement that all jointly forgotten components are semantically and transitionally unnecessary for the declared reconstruction task.

CPR-Width therefore depends on the component representation, the component domains, the reconstruction ordering, the declared computational task, the reachability and transition rules, and the reconstruction-safe future-signature family. The same finite problem instance may therefore possess different CPR-Width values for decision, counting, enumeration, optimization, or exact reconstruction tasks; in particular, $B_i^{\pi,\mathrm{dec}} \neq B_i^{\pi,\mathrm{count}}$ and $\omega_{\mathrm{rec}}(P, \pi, \mathrm{dec}; \mathfrak{M}) \neq \omega_{\mathrm{rec}}(P, \pi, \mathrm{count}; \mathfrak{M})$ may occur. CPR-Width counts active components; it does not directly measure their bit-level information content.

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.

Local domain sizes, projected interface-state spaces, reachable interface states, and semantic quotient states are developed in the following chapter. Figure 2.2 summarizes, at a glance, the complete pipeline connecting the three central quantities of this manuscript.

3 Raw States, Semantic States, and Reconstruction

Chapter 2 introduced reachable prefix states, task-sufficient active interfaces, reconstruction-safe future signatures, and Controlled Partial Reconstruction Width. The present chapter studies the state spaces induced by these objects, proceeding through four 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

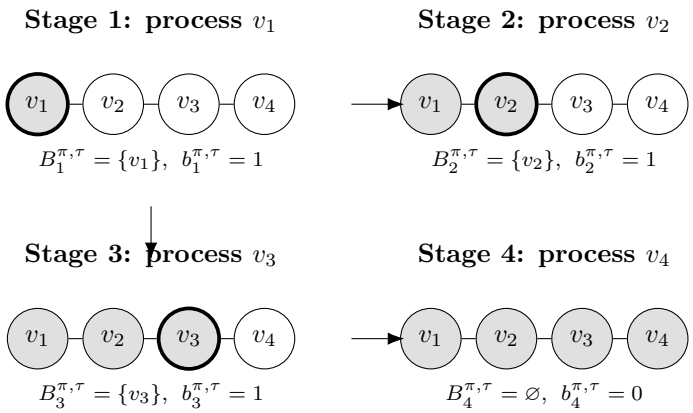


Figure 2.1: Stage-by-stage evolution of the active reconstruction interface $B_i^{\pi,\tau}$ on the running example (P_4, π, τ): outlined vertex, active interface; shaded vertex, processed but no longer active; white vertex, not yet processed. The maximum tracked size is $\widehat{\omega}_{\mathrm{rec}}(P_4,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

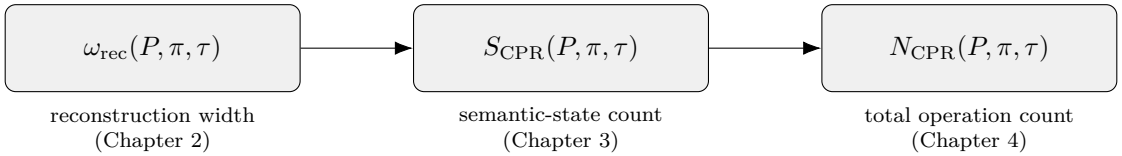


Figure 2.2: Semantic-runtime architecture of CPR-WIDTH: the structural chain connecting reconstruction width, semantic-state growth, and the complete operation-count model, as introduced in Chapter 1 and developed in full in Chapters 2–4. The reconstruction model $\mathfrak{M}$ is fixed throughout and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

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. The space $U_i(P, \pi, \tau; \mathfrak{M})$ contains every syntactically possible assignment to the selected active interface; it does not assert that every such assignment can actually occur during the declared reconstruction process. If $B_i^{\pi,\tau} = \varnothing$, the empty Cartesian product is interpreted as a singleton, $U_i(P, \pi, \tau; \mathfrak{M}) = \{\varnothing\}$, so $|U_i(P, \pi, \tau; \mathfrak{M})| = 1$. Define the maximum local domain size by $q(P) = \max_{v \in V(P)} |D_v|$; since every local domain is nonempty, $q(P) \geq 1$.

3.2 Raw Interface-State Bound

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

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

and consequently

$$|U_i(P, \pi, \tau; \mathfrak{M})| \leq q(P)^{\omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) + 1}.$$

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 $|D_v| \le q(P)$, hence $|U_i(P,\pi,\tau;\mathfrak{M})| \le q(P)^{|B_i^{\pi,\tau}|}$. Because the selected interface is cardinality-minimal, $|B_i^{\pi,\tau}| = b_i^{\pi,\tau}(P) \le \widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})$; since $q(P) \ge 1$, $q(P)^{|B_i^{\pi,\tau}|} \le q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}$, which proves the first inequality. By the Interface Bound of Chapter 2, $\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \le \omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) + 1$, which gives the second. □

The bound using $\widehat{\omega}_{\mathrm{rec}}$ is the sharper operational form; the normalized expression with $\omega_{\mathrm{rec}} + 1$ 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, $|U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})| \le q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}$, and consequently $|U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})| \le q(P)^{\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}$.*

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 $|S_i(P, \pi, \tau; \mathfrak{M})| \leq |U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M})|$.

Definition 3.5 (Maximum Semantic-State Count)**.** The maximum semantic-state count is $S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) = \max_{0 \leq i \leq n} |S_i(P, \pi, \tau; \mathfrak{M})|$.

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).** On (P_4, π, τ) with $k = 2$ colors, the interface at each internal stage is a single vertex with $q(P) = 2$ possible colors, so $|U_i(P, \pi, \tau)| \leq 2$ by the Raw State Bound. Across the reachable prefix states, either color may occur on the active interface vertex, so both raw states are reachable, $|U_i^{\mathrm{reach}}(P, \pi, \tau)| = 2$. No further semantic collapse is possible here: the two colors of the active vertex induce different admissible next values, since the next adjacent vertex must take the opposite color in each case, so $|S_i(P, \pi, \tau)| = |U_i^{\mathrm{reach}}(P, \pi, \tau)| = 2$ (the quotient is canonically identifiable with $U_i^{\mathrm{reach}}(P, \pi, \tau)$, since every equivalence class is a singleton), and $S_{\mathrm{CPR}}(P_4, \pi, \tau) = 2$. This is a case where semantic quotienting provides no compression beyond the reachability filter, in contrast to the ripple-adder developed in Chapters 7 and 8, where a compact carry signature yields a uniformly bounded semantic-state space, and the C_5 ordering-dependence instance (Section 6.3.2), where different admissible orderings produce different structural behavior; not every instance needs to show compression to be a valid illustration of the framework.

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

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

This theorem is a cardinality bound. It does not imply that the semantic quotient can be constructed efficiently, nor that the complete reconstruction runtime is polynomial. The Width–State Bound is a worst-case bound and need not be tight, because reachability restrictions and semantic quotienting may reduce the actual semantic-state count substantially below the raw width-based upper bound.

The two forms of the Width–State Bound serve distinct purposes. The unnormalized width gives the sharper operational exponent, whereas the normalized width preserves the convention adopted for CPR-Width. Since

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

one obtains

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

In general, the stronger expression $S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) \le q(P)^{\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}$ does not follow from the normalization convention; the exponent on the normalized width requires the $+1$.

3.9 Binary Semantic-State Dimension

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

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

This quantity is the minimum fixed-length binary index size required to distinguish among the maximum number of semantic classes occurring at any single reconstruction stage; it is an information-theoretic indexing requirement, not an assertion about how a semantic state is actually represented in memory, nor a measurement of the cost of constructing the semantic quotient, the total number of transitions, the cost of reconstructing outputs, or the complete runtime. If $q(P) \geq 2$, the Width–State Bound implies $D_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \leq \lceil \widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \log_2 q(P) \rceil \leq \lceil (\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) + 1) \log_2 q(P) \rceil$. 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. Different tasks may require different reconstruction-safe signature families, even for the same problem representation and reconstruction ordering: for decision, the required final answer may be a Boolean value; for counting, exact multiplicities must be preserved; for enumeration, every required complete output must remain reconstructible; for optimization, the preserved information may include an optimum value, optimal witnesses, or certified residual objective information. Therefore $\equiv_{P,\pi,i}^{\mathrm{dec},\mathfrak{M}}$, $\equiv_{P,\pi,i}^{\mathrm{count},\mathfrak{M}}$, $\equiv_{P,\pi,i}^{\mathrm{enum},\mathfrak{M}}$, and $\equiv_{P,\pi,i}^{\mathrm{opt},\mathfrak{M}}$ 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 $u \equiv_{P,\pi,i}^{\tau,\mathfrak{M}} v$. 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, $\widehat{\mathrm{Sig}}_{\tau,i}^{\pi}(u) = \widehat{\mathrm{Sig}}_{\tau,i}^{\pi}(v)$. Choose reachable prefix representatives r, r' with $\rho_i^{\pi,\tau}(r) = u$ and $\rho_i^{\pi,\tau}(r') = v$; the induced-signature definition gives $\mathrm{Sig}_{\tau,i}^{\pi}(r) = \mathrm{Sig}_{\tau,i}^{\pi}(r')$. Condition S1 of Chapter 2 gives $\mathrm{Ans}_{\tau,i}^{\pi}(r) = \mathrm{Ans}_{\tau,i}^{\pi}(r')$, proving task preservation. Condition S2 gives $\Lambda_i^{\pi}(r) = \Lambda_i^{\pi}(r')$, so the admissible next-value sets coincide. For every common admissible next value a, condition S3 gives $\mathrm{Sig}_{\tau,i+1}^{\pi}(\delta_i^{\pi}(r,a)) = \mathrm{Sig}_{\tau,i+1}^{\pi}(\delta_i^{\pi}(r',a))$, so the projected successors satisfy $\rho_{i+1}^{\pi,\tau}(\delta_i^{\pi}(r,a)) \equiv_{P,\pi,i+1}^{\tau,\mathfrak{M}} \rho_{i+1}^{\pi,\tau}(\delta_i^{\pi}(r',a))$; the quotient transition is consequently independent of the selected representative. □

Semantic safety does not generally imply $E_i^{\pi}(r) = E_i^{\pi}(r')$. 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 state spaces introduced in this chapter satisfy $U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}) \subseteq U_i(P,\pi,\tau;\mathfrak{M})$, and the semantic quotient is obtained through the surjective map $q_i^{\tau,\mathfrak{M}} : U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}) \twoheadrightarrow S_i(P,\pi,\tau;\mathfrak{M})$. Therefore $|S_i(P,\pi,\tau;\mathfrak{M})| \le |U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})| \le |U_i(P,\pi,\tau;\mathfrak{M})|$, and the complete stage-wise construction is

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

The first map forgets processed components outside the task-sufficient active interface; the second merges reachable interface assignments with identical reconstruction-safe signatures. Because the signature satisfies S1–S3, semantic quotienting preserves the declared task-relative future value, the admissible next-value set, and the semantic class of every admissible successor. Figure 3.1 illustrates this construction, from raw and reachable interface states through the semantic quotient to the maximum semantic-state count.

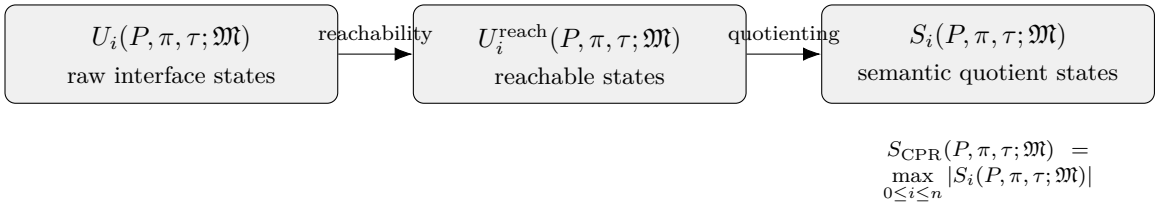


Figure 3.1: Controlled partial reconstruction and semantic-state compression: raw interface states are filtered to reachable states, then quotiented by reconstruction-safe signature equivalence to obtain semantic states, yielding the maximum semantic-state count S_{CPR}.

The resulting width–state relation, using either width form, is

$$\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \to S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \quad \text{or equivalently} \quad \omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \to S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}).$$

The present chapter establishes only a structural cardinality bound; the following chapter extends the framework to the complete operation-count model,

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

A small semantic-state count does not by itself imply that semantic classes are inexpensive to construct: a complete runtime conclusion must additionally include construction, reduction, transition, reconstruction, verification, and output costs.

4 Complete Runtime Framework

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

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

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

The six terms are disjoint accounting categories: every elementary operation of the declared CPR procedure must be assigned to exactly one category, never two. In particular, construction of a semantic class belongs either to N_{Reduce} or to $N_{\mathrm{Transition}}$ according to the declared accounting rule, but not both; writing an output object belongs to N_{Output}, not again to $N_{\mathrm{Reconstruct}}$; and verification steps belong to N_{Verify}, unless correctness follows directly from a proved construction with no separate verification operation executed. The accounting convention must be fixed before any count-level comparison is performed.

Table 4.1 summarizes the six accounting categories, the section in which each is bounded, and its role in the sum N_{CPR} of Equation 4.1, as a single point of reference for the remainder of the chapter.

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
$N_{\mathrm{Transition}}$	Stage-to-stage processing of semantic transitions along π.	Section 4.4
$N_{\mathrm{Reconstruct}}$	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). On $(P_4, \pi, \tau_{\mathrm{witness}})$ with $k = 2$ colors — the witness/exact-reconstruction task variant of Chapter 2's Running Example, which materializes the four color values rather than collapsing to a decision value — every stage processes exactly one vertex against a semantic-state space of size at most 2 (Section 3.6), so N_{Build}, N_{Reduce}, and $N_{\mathrm{Transition}}$ each cost $O(1)$ per stage, over the 4 stages of this instance; $N_{\mathrm{Reconstruct}}$, N_{Verify}, and N_{Output} each cost $O(1)$ per color for reading off or checking the four colors once. On this single fixed four-vertex instance these are concrete constant-size counts rather than an asymptotic statement: $O(\cdot)$-notation is properly defined only for a quantity varying with a growing parameter, and here $n = 4$ is fixed, not varying. The corresponding asymptotic claim — that every one of the six components grows as $O(n)$, giving $N_{\mathrm{CPR}} = O(n)$ — applies instead to the parameterized n-vertex path/cycle coloring family of which P_4 is the $n = 4$ instance, exactly as the same per-vertex accounting applies to the parameterized n-bit ripple-adder family: this is the quantitative content of the empirical measurements in Chapter 8, where semantic-state count stays at 2 and

the measured wall-clock time is consistent with approximately linear scaling in the tested range up to $n = 100{,}000$ bits.

4.2 Build and Reduction Cost

Definition 4.2 (Build Cost)**.** The build cost $N_{\mathrm{Build}}(P, \pi, \tau; \mathfrak{M})$ 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 η_i is the locally newly introduced information at stage i. 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, $S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \le q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}$. Consequently, the per-state transition bound becomes

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

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

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

equivalently with $q(P)^{\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}$ using the normalized width. The bound using $\widehat{\omega}_{\mathrm{rec}}$ 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): $m_{\mathrm{stage}}(P_4,\pi) = 4$, $S_{\mathrm{CPR}}(P_4,\pi,\tau) = 2$ (Section 3.6), and $\lambda_{\max}(P_4,\pi,\tau;\mathfrak{M}) = 1$, since fixing the color of the active interface vertex leaves exactly one admissible color available for the next vertex under $k = 2$ colors. Substituting into the per-edge Width-Based Transition Bound gives
>
> $$\begin{aligned} N_{\mathrm{Transition}}(P_4,\pi,\tau;\mathfrak{M}) \ &\le\ 4 \cdot 2\,(c_{\mathrm{edge}}(P_4,\pi,\tau;\mathfrak{M}) + c_{\mathrm{local}}(P_4,\pi,\tau;\mathfrak{M})) \\ &=\ 8\,(c_{\mathrm{edge}} + c_{\mathrm{local}})\,. \end{aligned}$$
>
> This is not a performance claim; it is a transparent instantiation of the abstract bound, showing explicitly how $\omega_{\mathrm{rec}} \to S_{\mathrm{CPR}} \to N_{\mathrm{Transition}} \to N_{\mathrm{CPR}}$ 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,*

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

and consequently, using the normalized width, $N_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \leq N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + m_{\mathrm{stage}}(P,\pi)\, q(P)^{\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}\, c_{\mathrm{state}}(P,\pi,\tau;\mathfrak{M}) + R_{\mathrm{post}}(P,\pi,\tau;\mathfrak{M})$.

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 $S_{\mathrm{CPR}} \leq q(P)^{\widehat{\omega}_{\mathrm{rec}}}$; substitution proves the first bound. Chapter 3 established $q(P) \geq 1$ and $\widehat{\omega}_{\mathrm{rec}} \leq \omega_{\mathrm{rec}} + 1$; since $x \mapsto q(P)^x$ is nondecreasing for $q(P) \geq 1$, it follows that $q(P)^{\widehat{\omega}_{\mathrm{rec}}} \leq q(P)^{\omega_{\mathrm{rec}}+1}$, proving the normalized bound. □

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

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

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

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

using $S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \leq q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}$ 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* $\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = O(\log \ell(P))$*. Then there exists a constant* $c > 0$ *such that* $q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} \leq \ell(P)^c$ *for all sufficiently large inputs; equivalently,* $q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} = \ell(P)^{O(1)}$.

Proof. If $q(P) = 1$, then $q(P)^{\widehat{\omega}_{\mathrm{rec}}} = 1$ for every input, which is trivially polynomially bounded. If $q(P) \geq 2$, since $q(P) \leq q_0$ and $\widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = O(\log \ell(P))$, there exists a constant $C > 0$ with $\widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) \leq C \log \ell(P)$ for all sufficiently large instances. Therefore $q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} \leq q_0^{C \log \ell(P)} = \ell(P)^{C \log q_0}$; 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 (Sufficient Complete Polynomial-Cost Condition)**.** *Assume there exist 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 one of the six cost components. Then there exists a polynomial p_{CPR} such that $N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) \leq p_{\mathrm{CPR}}(\ell(P))$.*

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 exists when $N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) < N_{\mathrm{Classic}}(P, \tau; \mathcal{A}_{\mathrm{Classic}})$, equivalently $G_{\mathrm{count}}(P, \pi, \tau; \mathfrak{M}, \mathcal{A}_{\mathrm{Classic}}) > 1$. A count-level comparison is meaningful only when both methods solve the same declared task and materialize the same required output; a baseline based on exhaustive enumeration is one possible baseline, but not the only admissible one.

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, $G_{\mathrm{count}} > 1 \not\Longrightarrow G_{\mathrm{time}} > 1$. Measured runtime may depend on implementation overhead, memory allocation, cache behavior, data movement, parallelism, synchronization, compiler behavior, hardware architecture, operating-system effects, and measurement protocol. Operation-count claims and wall-clock claims must therefore remain separate.

4.10 Structural Runtime Chain and Non-Implications

The theory developed in Chapters 2–4 yields the central structural runtime chain

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

or, using the normalized width, the same chain with ω_{rec} in place of $\widehat{\omega}_{\mathrm{rec}}$. The first arrow is a structural cardinality relation: the active-interface width bounds the raw interface-state space, which bounds the reachable interface-state space, which in turn bounds the semantic quotient. The second arrow is an accounting relation: the semantic-state count contributes to the transition cost, while the remaining cost components determine the complete computational effort. Neither arrow is a universal tractability implication.

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

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

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

Controlled Partial Reconstruction Width is a structural control parameter; it does not determine runtime by itself. The complete CPR operation count is

$$N_{\mathrm{CPR}} = N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} + N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}},$$

six disjoint accounting categories to which every elementary operation must be assigned exactly once. The transition cost must account explicitly for the number of reconstruction stages, the number of semantic states, the number of admissible outgoing transitions, the state-local processing cost, and the cost per transition edge: under the per-edge convention, $N_{\mathrm{Transition}} \leq m_{\mathrm{stage}}\, S_{\mathrm{CPR}}(\lambda_{\max} c_{\mathrm{edge}} + c_{\mathrm{local}})$; under the per-state convention, $N_{\mathrm{Transition}} \leq m_{\mathrm{stage}}\, S_{\mathrm{CPR}}\, c_{\mathrm{state}}$. The width-based bound controls the semantic-state contribution through $S_{\mathrm{CPR}} \leq q(P)^{\widehat{\omega}_{\mathrm{rec}}} \leq q(P)^{\omega_{\mathrm{rec}}+1}$. A complete polynomial-time conclusion is permitted only when every cost component is polynomially bounded in the encoding length.

The central structural-runtime chain remains $\boldsymbol{\omega}_{\mathbf{rec}} \to \boldsymbol{S}_{\mathbf{CPR}} \to \boldsymbol{N}_{\mathbf{CPR}}$, and a refined runtime-accounting chain is $\boldsymbol{\omega}_{\mathbf{rec}} \to \boldsymbol{S}_{\mathbf{CPR}} \to \boldsymbol{N}_{\mathbf{Transition}} \to \boldsymbol{N}_{\mathbf{CPR}}$, making explicit that the semantic-state count enters the complete operation count primarily through transition processing, without

implying that the remaining cost components are determined by the transition cost or by the reconstruction width. Figure 4.1 summarizes the complete pipeline.

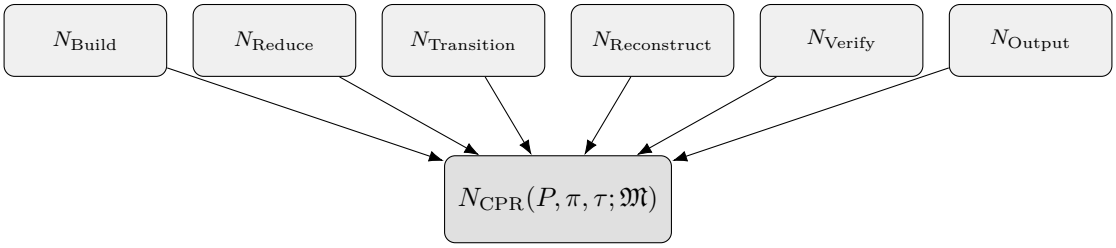


Figure 4.1: Complete runtime framework of CPR-WIDTH: the six disjoint operation-count components of Definition 4.1 sum to the total operation count N_{CPR}.

The following chapter formulates the complete boundary conditions and uniform polynomial-cost requirements needed for CPR tractability.

5 Boundary Conditions for CPR Tractability

The structural and runtime framework developed in the preceding chapters does not by itself imply polynomial-time tractability. Additional uniform conditions are required to ensure that Controlled Partial Reconstruction can be executed within polynomial resources for an entire encoded problem family. This chapter formulates those conditions and establishes the resulting tractability criterion.

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 tractability requires more than a small reconstruction width. The reconstruction representation must be explicitly specified and uniformly constructible; the semantic reconstruction process must be sound and complete for the declared task; the reachable semantic-state system and its transition structure must be polynomially controllable; and every component of the complete runtime model of Chapter 4 must be polynomially bounded. These requirements are organized through four boundary conditions: BC1, explicit uniform structural representation; BC2, polynomial uniform constructibility; BC3, task-relative soundness and completeness; BC4, polynomial semantic-state and transition-structure control. The boundary conditions are necessary components of the declared CPR tractability certificate; they do not, considered in isolation, replace the complete six-component runtime analysis.

Certification versus realization. The boundary conditions introduced below are certification conditions, not assertions that every problem family admits an efficient CPR realization. For a concrete family, a CPR-tractability result therefore requires two logically separate steps. First, one must exhibit a uniform reconstruction model satisfying BC1–BC4. Second, one must

establish polynomial bounds for all six components of the complete operation-count model. A small CPR-Width, a compact semantic quotient, or a finite successful reconstruction example establishes neither step by itself. Consequently, the framework distinguishes between a *structural realization*, which identifies the CPR objects for a problem family, and a *tractability certification*, which proves the complete uniform conditions stated in this chapter. This distinction is essential when the framework is applied to nontrivial problem families in the following chapters.

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 $\mathrm{Sig}_{\tau,i}^{\pi_x}$; 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, $N_{\mathcal{A}_{\mathrm{Preprocess}}}(x) \le p_{\mathrm{BC2}}(|x|)$, 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 requires polynomial constructibility of every structural object the declared model assigns to preprocessing, and is uniform: the same algorithm and the same polynomial bound must apply to every instance in the encoded family; an instance-specific construction procedure or an instance-dependent polynomial is not sufficient. BC2 does not by itself imply that semantic transitions, reconstruction, verification, or output materialization are polynomially bounded; those costs remain separate components of the complete runtime model.

Importantly, BC2 does not permit an exponentially large continuation space to be hidden inside a supposedly elementary preprocessing step. If a reconstruction-safe signature, active interface, or auxiliary semantic representation is constructed by explicit enumeration of all suffix

continuations, then the complete cost of that enumeration must be charged to the appropriate runtime component. Such a construction satisfies BC2 only if its total bit-cost is polynomially bounded in $|x|$. Conversely, CPR does not require continuation-space enumeration when the declared reconstruction model provides a direct or incremental construction of the required structural data. The existence and cost of such a construction are model-dependent and must be proved for the family under consideration.

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; 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, $\Theta_\tau^{\mathrm{CPR}}(P_x, \pi_x; \mathfrak{M}) = \Theta_\tau(P_x)$; 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, $V_{\mathrm{sem}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_{\mathrm{SemanticVertex}}(|x|)$ and $E_{\mathrm{sem}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_{\mathrm{SemanticEdge}}(|x|)$.

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 $N_{\text{Transition}}(P_x, \pi_x, \tau; \mathfrak{M})$, 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 $\mathcal{A}_{\text{Preprocess}}$, containing an ordering-supply component $\mathcal{A}_\pi$ with $\pi_x = \mathcal{A}_\pi(x) \in \Pi_{\text{adm}}(P_x, \tau; \mathfrak{M})$, and six fixed polynomial functions $p_j : \mathbb{N} \to \mathbb{N}$, $j \in J = \{\text{Build}, \text{Reduce}, \text{Transition}, \text{Reconstruct}, \text{Verify}, \text{Output}\}$, such that BC1–BC4 hold uniformly and, for every encoding x, $N_j(P_x, \pi_x, \tau; \mathfrak{M}) \le p_j(|x|)$ for every $j \in J$.

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

Membership in $\mathbf{P}_{\text{CPR}}$ 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 $\mathcal{A}_{\text{Preprocess}}$ with ordering-supply component $\mathcal{A}_\pi$ producing $\pi_x = \mathcal{A}_\pi(x) \in \Pi_{\text{adm}}(P_x, \tau; \mathfrak{M})$, 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 $N_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_{\mathrm{Build}}(|x|) + p_{\mathrm{Reduce}}(|x|) + p_{\mathrm{Transition}}(|x|) + p_{\mathrm{Reconstruct}}(|x|) + p_{\mathrm{Verify}}(|x|) + p_{\mathrm{Output}}(|x|)$, 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 $L_{\mathcal{F}} \in \mathbf{P}$. □

The role of the two groups of hypotheses in Theorem 5.1 is deliberately asymmetric. BC1–BC4 are structural certification conditions: they require the reconstruction model to be explicitly representable, uniformly constructible, sound and complete, and structurally bounded in semantic-state and transition cardinality, but on their own they assert no polynomial-time conclusion. The actual polynomial-time conclusion is carried by the six assumed polynomial bounds $N_j(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_j(|x|)$; once all six are assumed, membership of $L_{\mathcal{F}}$ in $\mathbf{P}$ follows immediately from the additive operation-count decomposition of Chapter 4, essentially by definition of polynomial-time computation. The mathematical content of this theorem is therefore not the final summation step, but the certification structure BC1–BC4 impose on *how* the six bounds may legitimately be obtained and audited for a declared reconstruction model.

The theorem should therefore be read as a certification theorem rather than as a width-only tractability theorem. Its purpose is to prevent three invalid inferences:

$$\text{small CPR-Width} \not\Longrightarrow \text{polynomial construction cost}, \tag{5.1}$$

$$\text{polynomially many semantic states} \not\Longrightarrow \text{polynomial transition cost}, \tag{5.2}$$

and

$$\text{polynomial internal reconstruction} \not\Longrightarrow \text{polynomial total runtime} \tag{5.3}$$

when verification or required output materialization is superpolynomial. A CPR tractability proof must close all three gaps within one uniform encoded family.

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

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

The four boundary conditions define the structural, constructibility, correctness, and semantic-control requirements of a CPR reconstruction model: BC1, explicit uniform structural representation; BC2, polynomial uniform constructibility; BC3, task-relative soundness and completeness; BC4, polynomial semantic-state and transition-structure control. They do not replace the complete runtime model; a polynomial-time conclusion additionally requires fixed polynomial functions $p_j : \mathbb{N} \to \mathbb{N}$, $j \in J$, such that, uniformly for every encoded instance P_x, $N_j(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_j(|x|)$ for every $j \in J$. Accordingly, the complete sufficient tractability condition is

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

where $\mathcal{A}_\pi$ is the ordering-supply component of $\mathcal{A}_{\text{Preprocess}}$. The order of quantifiers is essential: the task, reconstruction model, preprocessing algorithm, admissible ordering-supply rule, and six polynomial functions must be fixed for the complete encoded family before the individual instance x is selected. Instance-dependent reconstruction models, ordering rules, or polynomial bounds do not establish uniform polynomial-time tractability.

The class $\mathbf{P}_{\text{CPR}}$ contains exactly the decision languages certified by such a uniform CPR model. Therefore $\mathbf{P}_{\text{CPR}} \subseteq \mathbf{P}$, and for an NP-complete language the corresponding complexity consequence remains strictly conditional, L^* NP-complete $\wedge\, L^* \in \mathbf{P}_{\text{CPR}} \Longrightarrow \mathbf{P} = \mathbf{NP}$, a premise not established in this manuscript; consequently no conclusion $\mathbf{P} = \mathbf{NP}$ is proved.

Family-level scaling requirement. For a parameterized realization such as an n-component circuit, an n-bit arithmetic system, or a graph family of increasing order, finite verification at one parameter value is insufficient for BC2 or BC4. A family-level certification must derive explicit asymptotic bounds for the construction of the semantic representation and for the number and processing cost of semantic transitions as functions of the input encoding length. This requirement is particularly important when a constant or bounded active interface is observed. Constant interface width alone does not prove that the corresponding semantic quotient can be constructed in polynomial time. The construction mechanism and its asymptotic cost must be established independently.

The following chapters examine representative problem classes in order to identify their reconstruction orderings, task-sufficient active interfaces, reachable semantic states, reconstruction-safe signatures, reconstruction operators, and problem-dependent width expressions. Those realizations illustrate how the boundary conditions may be studied explicitly; they do not by themselves establish CPR tractability for the corresponding unrestricted problem classes.

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 $\pi \in \Pi_{\text{adm}}(P, \tau; \mathfrak{M})$ be an admissible

reconstruction ordering; the dependence on τ and $\mathfrak{M}$ may be suppressed.

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 been proved, one may conclude only $b_i^{\pi,\tau}(P) \le |\widetilde{B}_i^{\pi,\tau}|$, and therefore $\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \le \max_i |\widetilde{B}_i^{\pi,\tau}|$; equality requires an additional minimality proof.

6.1 Boolean Satisfiability

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

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

the set of processed variables whose assigned values may still influence residual clauses; it is a natural structural candidate interface, but not automatically the unique or cardinality-minimal task-sufficient one. For a reachable prefix assignment $r \in \mathcal{R}_i^\pi(\Phi)$, define the exact satisfying-extension set $E_i^{\Phi,\pi}(r) = \{s \in \prod_{x \in \overline{V}_i^\pi} \{0,1\} : r \cup s \models \Phi\}$; equality of these sets may be used as one particular reconstruction-safe prefix signature, $\mathrm{Sig}_{\mathrm{SAT},i}^\pi(r) = E_i^{\Phi,\pi}(r)$. This signature is stronger than the mere current decision value $\mathbf{1}[E_i^{\Phi,\pi}(r) \neq \varnothing]$: because it records the complete residual satisfying-extension set, it preserves the decision answer, admissible next values, and successor residual classes, and so satisfies conditions S1–S3 of Chapter 2. If the candidate interface $\widetilde{B}_i^{\Phi,\pi}$ determines this signature, then it is task-sufficient and $b_i^{\pi,\mathrm{SAT}}(\Phi) \le |\widetilde{B}_i^{\Phi,\pi}|$, so $\widehat{\omega}_{\mathrm{rec}}(\Phi,\pi,\mathrm{SAT};\mathfrak{M}) \le \max_i |\widetilde{B}_i^{\Phi,\pi}|$. An equality claim requires a proof that no smaller processed-variable set determines the same 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 $V = \{x_1, \ldots, x_n\}$, $D = (D_{x_1}, \ldots, D_{x_n})$, and $\mathcal{C} = \{C_1, \ldots, C_m\}$; for a constraint C, let $\mathrm{scope}(C) \subseteq V$ denote its variable scope. For an admissible ordering π, define the structural CSP candidate interface

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

the processed variables participating in constraints whose scopes still contain unresolved variables. For a reachable prefix assignment r, a strong exact residual signature may be defined by $\mathrm{Sig}_{\mathrm{CSP},i}^\pi(r) = \{s \in \prod_{x \in \overline{V}_i^\pi} D_x : r \cup s \text{ satisfies every constraint in } \mathcal{C}\}$; for counting, optimization, or another declared task, the signature may instead preserve the corresponding task value together with the transition information required by S1–S3. If $\widetilde{B}_i^{P_{\mathrm{CSP}},\pi}$ determines the declared safe signature, then $b_i^{\pi,\tau}(P_{\mathrm{CSP}}) \le |\widetilde{B}_i^{P_{\mathrm{CSP}},\pi}|$, hence $\widehat{\omega}_{\mathrm{rec}}(P_{\mathrm{CSP}},\pi,\tau;\mathfrak{M}) \le \max_i |\widetilde{B}_i^{P_{\mathrm{CSP}},\pi}|$. The structural boundary is 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 $K = \{1, \ldots, k\}$ the available color set. For an ordering $\pi = (v_{\pi(1)}, \ldots, v_{\pi(|V|)})$, with $V_i^\pi = \{v_{\pi(1)}, \ldots, v_{\pi(i)}\}$ and $\overline{V}_i^\pi = V \setminus V_i^\pi$, define the structural graph-coloring candidate interface

$$\widetilde{B}_i^{G,\pi} = \left\{v \in V_i^\pi : \exists w \in \overline{V}_i^\pi \text{ with } \{v,w\} \in E\right\},$$

exactly the classical vertex-separation boundary of π. 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, and consequently $\widehat{\omega}_{\mathrm{rec}}(G, \pi, \tau; \mathfrak{M}) \leq \max_i |\widetilde{B}_i^{G,\pi}|$. The right-hand side is the ordering-specific vertex-separation value, the left-hand side the minimum task-sufficient CPR interface width; they may coincide for a particular realization, but this equality requires proof. The role of this realization is therefore not to rename vertex separation, but to embed a classical structural boundary into a task-relative reconstruction model with semantic quotienting and complete runtime accounting.

Preliminary Width-Comparison Hypothesis. The graph-coloring realization suggests a natural comparison with classical ordering-based width parameters.

For the declared coloring reconstruction model, the structural candidate interface at stage i is exactly the vertex-separation boundary of the ordering π. Whenever this boundary is proved task-sufficient for the selected reconstruction-safe signature, one has

$$\widehat{\omega}_{\mathrm{rec}}(G, \pi, \tau; \mathfrak{M}) \leq \mathrm{vs}(G, \pi),$$

where $\mathrm{vs}(G, \pi)$ denotes the ordering-specific vertex-separation number.

This motivates the following preliminary hypothesis:

for suitable graph tasks and representations, CPR-Width is
upper-controlled by a classical separator-based width parameter.

The statement is deliberately one-sided. Semantic quotienting may make the minimum task-sufficient CPR interface strictly smaller than the structural separator, so equality need not hold.

A formal theorem relating minimum CPR-Width to pathwidth, treewidth, or another classical width parameter requires a fixed representation, task, signature family, and convention for normalization and is left for future work.

6.3.1 Quantitative Verification for C_4

Consider the four-vertex cycle $C_4 = (V, E)$ with $V = \{v_1, v_2, v_3, v_4\}$ and three available colors. The 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; the number of admissible colorings is $N_{\mathrm{adm}} = |\Phi(C_4)| = 18$. Fix 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, $N_{\mathrm{raw}} = 81 \neq N_{\mathrm{adm}} = 18 \neq N_{\mathrm{rec}}(f) = 3$: N_{raw} is the cardinality of the complete raw assignment space, N_{adm} that of the admissible proper-coloring space, and $N_{\mathrm{rec}}(f)$ that of the reconstruction fiber induced by the fixed partial coloring f. These are finite cardinalities, not CPR-Width values, and must not be identified with the active-interface width, the minimum task-sufficient interface size, or the semantic-state count. They distinguish the raw assignment space, the admissible coloring space, and one reconstruction fiber, but are not direct runtime comparisons. The admissible-to-fiber factor is $K_{\mathrm{adm/fiber}} = 18/3 = 6$, and the raw-to-fiber factor is $K_{\mathrm{raw/fiber}} = 81/3 = 27$; both factors compare the conditional reconstruction fiber $N_{\mathrm{rec}}(f)$ obtained *after fixing* the partial coloring f against unconditional cardinalities, and must not be read as an algorithmic speedup factor of any reconstruction procedure.

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

The C_4 example above compares raw, admissible, and fiber cardinalities but does not track the structural candidate interface $\widetilde{B}_i^{G,\pi}$ stage by stage, and it fixes a single ordering throughout. The five-vertex cycle $C_5 = (V, E)$ with

$$V = \{v_1, \ldots, v_5\}, \qquad E = \{\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}, \{v_4, v_5\}, \{v_5, v_1\}\},$$

gives a second worked instance that tracks $\widetilde{B}_i^{G,\pi}$ explicitly at every stage and makes the ordering dependence of Proposition 2.2 concrete with exact numbers instead of an abstract statement alone.

Ordering A (consecutive traversal). Let $\pi_A = (v_1, v_2, v_3, v_4, v_5)$. Tracking $\widetilde{B}_i^{G,\pi_A} = \{v \in V_i^{\pi_A} : \exists w \notin V_i^{\pi_A}, \{v, w\} \in E\}$ stage by stage,

$$\widetilde{B}_1^{G,\pi_A} = \{v_1\}, \quad \widetilde{B}_2^{G,\pi_A} = \{v_1, v_2\}, \quad \widetilde{B}_3^{G,\pi_A} = \{v_1, v_3\}, \quad \widetilde{B}_4^{G,\pi_A} = \{v_1, v_4\},$$

with $\widetilde{B}_5^{G,\pi_A} = \varnothing$ once the suffix is empty. The vertex v_1 remains in the interface from stage 1 through stage 4 because its neighbor v_5 is only processed last; every other processed vertex leaves the interface as soon as both of its neighbors have been processed. The maximum boundary size is $\max_i |\widetilde{B}_i^{G,\pi_A}| = 2$, so $\widehat{\omega}_{\mathrm{rec}}(C_5, \pi_A) \leq 2$ and $\omega_{\mathrm{rec}}(C_5, \pi_A) \leq 1$.

Ordering B (interleaved traversal). Let $\pi_B = (v_1, v_3, v_5, v_2, v_4)$, which processes alternating vertices of the cycle before returning to fill in the remainder. Tracking the same quantity,

$$\widetilde{B}_1^{G,\pi_B} = \{v_1\}, \quad \widetilde{B}_2^{G,\pi_B} = \{v_1, v_3\}, \quad \widetilde{B}_3^{G,\pi_B} = \{v_1, v_3, v_5\}, \quad \widetilde{B}_4^{G,\pi_B} = \{v_3, v_5\}.$$

At stage 3, all three processed vertices v_1, v_3, v_5 still have an unprocessed neighbor among $\{v_2, v_4\}$ (each of v_2 and v_4 is adjacent to two of the three processed vertices, while v_2 and v_4 are not adjacent to each other), so all three remain simultaneously active, giving $\max_i |\widetilde{B}_i^{G,\pi_B}| = 3$, $\widehat{\omega}_{\mathrm{rec}}(C_5, \pi_B) \leq 3$, and $\omega_{\mathrm{rec}}(C_5, \pi_B) \leq 2$.

Reading the comparison. The same graph, the same task, and the same reconstruction model give a strictly larger tracked structural candidate boundary under π_B than under π_A at the point of maximum overlap (stage 3 of π_B versus stage 2 or 4 of π_A): consecutive traversal keeps the structural boundary of the cycle at size at most 2 for every prefix, while interleaved traversal temporarily activates every other vertex at once. This qualitatively illustrates, with concrete numbers, the same ordering-dependence phenomenon proved abstractly in Proposition 2.2; it is not itself a further instance of that proposition, since $\max_i |\widetilde{B}_i^{G,\pi_A}| = 2$ and $\max_i |\widetilde{B}_i^{G,\pi_B}| = 3$ only bound $\widehat{\omega}_{\mathrm{rec}}$ from above (Section 6.3), and cardinality-minimality of $\widetilde{B}_i^{G,\pi}$ for the declared graph-coloring signature has not been established at every stage. The structural candidate boundary differs by one between the two orderings; whether the exact normalized CPR-Width $\omega_{\mathrm{rec}}(C_5, \pi)$ differs by a full unit requires a minimality proof and is not claimed here. Since $\omega_{\mathrm{rec}}(C_5, \tau; \mathfrak{M})$ is minimized over admissible orderings by definition (Section 2.14), this instance still illustrates why the minimization over π is not a formality: an unfavorable but still admissible ordering can enlarge the structural candidate boundary.

The present C_5 calculation determines exact structural candidate-boundary sizes for two orderings, but it does not claim an exact CPR-Width because cardinality minimality has not been proved. A larger random 3-SAT interface experiment is reported separately in Chapter 8; it provides empirical structural-width evidence rather than an exact minimum-CPR-Width computation.

6.4 Common Reconstruction Pattern

The representative problem classes considered in this chapter possess different structural candidate interfaces, semantic interpretations, and instance-specific width expressions. For every declared realization, one must distinguish the structural candidate interface $\widetilde{B}_i^{\pi,\tau}$ from the cardinality-minimal task-sufficient interface $B_i^{\pi,\tau}$. Table 6.1 records the structural realizations and the corresponding instance-specific CPR-Width notations.

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 the unrestricted problem classes. For SAT, CSP, graph coloring, and tensor representations, the relevant expressions are

$$\omega_{\mathrm{rec}}(\Phi, \pi), \qquad \omega_{\mathrm{rec}}(P_{\mathrm{CSP}}, \pi), \qquad \omega_{\mathrm{rec}}(G, \pi), \qquad \omega_{\mathrm{rec}}(T_P, \pi).$$

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

Problem class	**Structural/interface realization**	**Semantic interpretation**	**Width or finite quotient quantity**
SAT	Assigned variables still occurring in unresolved clauses.	Future satisfying extensions.	$\omega_{\mathrm{rec}}(\Phi, \pi)$
CSP	Processed variables shared with unresolved constraints.	Future globally compatible assignments.	$\omega_{\mathrm{rec}}(P_{\mathrm{CSP}}, \pi)$
Graph coloring	Processed vertices adjacent to unresolved vertices.	Future proper colorings.	$\omega_{\mathrm{rec}}(G, \pi)$
Tensor	Active indices affecting unresolved structural classes.	Future output-safe tensor states.	$\omega_{\mathrm{rec}}(T_P, \pi)$
Full adder (isolated)	No staged CPR interface is claimed here; the finite Hamming-weight output quotient preserves the exact output.	Finite output-equivalence classes; not a staged semantic quotient.	$S_{\mathrm{out}}(P_{\mathrm{FA}})$; no stage-wise $\omega_{\mathrm{rec}}(A, \pi)$ is claimed in this chapter (see Chapters 7–8 for the staged ripple-adder model).

These denote widths of particular represented instances, declared tasks, reconstruction models, and admissible orderings.

The isolated full adder is treated differently in this chapter: 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 Theory of a Partially Reconstructible Subclass

Let $\mathcal{C}$ be a uniformly encoded combinatorial 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 that one fixed task τ, one fixed uniform reconstruction model $\mathfrak{M}$, and one deterministic uniform preprocessing algorithm $A_{\mathrm{Preprocess}}$ are declared for the subclass, containing an ordering-supply component A_π in the sense of Definition 5.2 (Chapter 5): A_π either reads an admissible ordering from the declared input or constructs one uniformly from x, and its operation count is included in N_{Build}, not treated as a separately declared algorithm.

For every encoded instance

$$P_x \in \mathcal{C}',$$

the algorithm must produce

$$\pi_x = A_\pi(x) \in \Pi_{\text{adm}}(P_x, \tau; \mathfrak{M}).$$

The subclass has controlled CPR-Width if there exists a fixed width-bound function $w_{\mathcal{C}'}$ such that

$$\omega_{\text{rec}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq w_{\mathcal{C}'}(\ell(P_x))$$

for every $P_x \in \mathcal{C}'$.

The width condition alone does not establish polynomial reconstructibility. The four CPR boundary conditions must hold uniformly and every component of the complete CPR operation-count model must satisfy a fixed polynomial bound

$$N_j(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_j(\ell(P_x)),$$

for

$$j \in \{\text{Build}, \text{Reduce}, \text{Transition}, \text{Reconstruct}, \text{Verify}, \text{Output}\}.$$

Under these conditions,

$$N_{\text{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_{\text{CPR}}(\ell(P_x))$$

for one fixed polynomial p_{CPR}.

Let $J = \{\text{Build}, \text{Reduce}, \text{Transition}, \text{Reconstruct}, \text{Verify}, \text{Output}\}$ denote the index set of the six cost components used above. The complete theory may therefore be written compactly as

$$\boldsymbol{\exists \varnothing \neq \mathcal{C}' \subseteq \mathcal{C} \quad \exists \tau \quad \exists \mathfrak{M} \quad \exists A_{\text{Preprocess}} \quad \exists w_{\mathcal{C}'} \quad \exists \{p_j\}_{j \in J} \quad \forall P_x \in \mathcal{C}' :}$$

$$\boldsymbol{\pi_x = A_\pi(x) \in \Pi_{\text{adm}}(P_x, \tau; \mathfrak{M})}$$

together with

$$\boldsymbol{\omega_{\text{rec}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq w_{\mathcal{C}'}(\ell(P_x))}$$

and

$$\boldsymbol{\text{BC}_1 \wedge \text{BC}_2 \wedge \text{BC}_3 \wedge \text{BC}_4}$$

and

$$\boldsymbol{\bigwedge_{j \in J} [N_j(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_j(\ell(P_x))],}$$

which together imply

$$\boldsymbol{N_{\text{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) \in \ell(P_x)^{O(1)}.}$$

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 $\omega_{\text{rec}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq k_{\mathcal{C}'}$ 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. In this case one may choose the constant bound function

$$w_{\mathcal{C}'}(\ell(P_x)) = k_{\mathcal{C}'}.$$

For bounded local domains, $q(P_x) \leq q_0$, the width-dependent state factor is then bounded by $q(P_x)^{\omega_{\text{rec}}(P_x, \pi_x, \tau; \mathfrak{M})+1} \leq q_0^{k_{\mathcal{C}'}+1}$, constant with respect to the encoding length. This does not by itself prove polynomial runtime: the construction of the ordering, interfaces, signatures, quotient states, reconstruction, verification, and output must still satisfy the complete polynomial-cost conditions.

6.5.3 Meaning of the Theory

The theory does not claim that every unrestricted problem class has one fixed constant CPR-Width. It claims that subclasses may exist whose instances admit a suitable admissible reconstruction ordering, have controlled or constant CPR-Width, satisfy all four CPR boundary

conditions, and admit complete polynomial CPR reconstruction for the declared task. Thus, the correct statement is not that every problem class has one universal constant width, but rather

a problem class may contain subclasses with controlled or constant CPR-Width.

For Table 6.1, this means that $\omega_{\mathrm{rec}}(\Phi, \pi)$, $\omega_{\mathrm{rec}}(P_{\mathrm{CSP}}, \pi)$, $\omega_{\mathrm{rec}}(G, \pi)$, and $\omega_{\mathrm{rec}}(T_P, \pi)$ are the corresponding problem-class-specific width expressions; for suitable subclasses, these expressions must be calculated, estimated, or bounded. The isolated full adder is the one exception: as noted in the table, its row records the finite output-quotient state count $S_{\mathrm{out}}(P_{\mathrm{FA}})$ rather than a stage-wise $\omega_{\mathrm{rec}}(A, \pi)$, since no staged CPR model is declared for the isolated adder in this chapter. The table records different realizations of one general CPR-Width theory.

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 distinction permits width to be compared with, but not identified with, other resource quantities such as

$$\begin{gathered}\text{input size}, \quad \text{search-space size}, \quad \text{semantic-state count}, \\ \text{reconstruction depth}, \quad \text{memory representation size}, \quad \text{and runtime}.\end{gathered} \tag{6.1}$$

Consequently, CPR-Width can be useful even when it does not by itself produce a runtime improvement.

For example, a reconstruction process may have a small active interface while requiring many reconstruction stages, expensive signature construction, large encoded semantic states, or costly output materialization.

Conversely, a problem may possess a comparatively large structural interface while still admitting efficient processing because the associated transitions or semantic encodings are inexpensive.

Thus,

$$\text{small CPR-Width} \not\Longrightarrow \text{small runtime} \tag{6.2}$$

and

$$\text{growing CPR-Width alone} \not\Longrightarrow \text{computational intractability}. \tag{6.3}$$

The scientific use of CPR-Width is therefore not limited to speedup claims.

It provides a structural coordinate with which different sources of computational difficulty can be separated experimentally and theoretically.

For a family of instances, one may study whether runtime growth is associated primarily with

- growth of the active reconstruction interface;
- growth of the number of semantic quotient states;
- 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 produces a diagnostic use of width.

Instead of asking only whether a problem is “easy” or “hard”, the framework asks which resource becomes responsible for the observed growth.

Accordingly, CPR-Width may serve as

1. a structural classification parameter for identifying controlled-width subclasses;
2. a design parameter for comparing admissible reconstruction orderings;
3. a diagnostic parameter for separating structural growth from transition, representation, verification, and output costs;
4. an experimental parameter for studying correlations between active-interface growth and measured computational behavior;
5. and a bridge parameter for future comparison with classical width notions such as treewidth, pathwidth, branchwidth, OBDD width, hypertree width, or related problem-specific widths.

The value of the parameter therefore does not depend on demonstrating a universal speedup.

A negative runtime result can itself be informative when a small CPR-Width is observed but another resource dominates the complete operation count.

Such a result identifies a separation between structural width and the actual computational bottleneck rather than invalidating the width description.

This interpretation also explains why the complete CPR framework keeps the chain

$$\boldsymbol{\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} \longrightarrow \boldsymbol{S_{\mathbf{CPR}}(P,\pi,\tau;\mathfrak{M})} \longrightarrow \boldsymbol{N_{\mathbf{CPR}}(P,\pi,\tau;\mathfrak{M})} \tag{6.4}$$

as a dependency structure rather than an equivalence.

CPR-Width controls one structural coordinate of reconstruction. The complete computational behavior is determined only after the remaining resource coordinates have also been analyzed.

6.6 Common Reconstruction Chain

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

$$\text{Ordering} \longrightarrow \text{Structural Candidate Interface} \longrightarrow \text{Task-Sufficient Interface}$$

followed by

$$\text{Reachable Interface States} \longrightarrow \text{Semantic Quotient} \longrightarrow \text{Reconstruction},$$

or, in symbolic form, $\pi \to \widetilde{B}_i^{\pi,\tau} \to B_i^{\pi,\tau} \to U_i^{\mathrm{reach}} \to S_i \to \mathrm{Output}_\tau$. The candidate interface is problem-class dependent; task sufficiency, minimality, semantic quotienting, 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; a reconstruction ordering may process tensor indices, fragments, or local structural relations. This use of "tensor" is structural, denoting a finite multi-indexed array serving as a reconstruction representation; it is not an assertion about, or a bound on, the multilinear-algebraic tensor rank of T_P, which is a separate notion not addressed here. Define a structural tensor candidate interface

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

The exact interpretation of α depends on the declared tensor representation: it may denote an index, a tensor fragment, a local compatibility relation, or another explicitly encoded reconstruction component. A reconstruction-safe tensor signature must preserve the task-relevant future tensor behavior required by S1–S3, for example all admissible future tensor completions, a residual output-equivalence class, a task-relevant structural output vector, or another certified transition-congruent signature. If the structural tensor boundary determines the selected safe signature, then $b_i^{\pi,\tau}(T_P) \le |\widetilde{B}_i^{T_P,\pi}|$ and $\widehat{\omega}_{\mathrm{rec}}(T_P,\pi,\tau;\mathfrak{M}) \le \max_i |\widetilde{B}_i^{T_P,\pi}|$. Tensor representation supplies a structural carrier; it does not by itself prove a small CPR-Width, efficient semantic quotienting, or polynomial runtime.

6.8 Full Adder

The full adder provides a finite output-preserving quotient. Let $\mathbb{B} = \{0, 1\}$ and $X_{\mathrm{FA}} = \mathbb{B}^3$; an input state has the form $x = (a, b, c_{\mathrm{in}})$, and the full-adder output map is $F_{\mathrm{FA}} : \mathbb{B}^3 \to \mathbb{B}^2$. 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 $Q_{\mathrm{FA}}(h) = (h \bmod 2, \lfloor h/2 \rfloor)$. Then $F_{\mathrm{FA}} = Q_{\mathrm{FA}} \circ C_{\mathrm{FA}}$: the eight input states collapse into four output-equivalence classes, $8 \to 4$. This is an exact finite output quotient; it should not automatically be identified with $S_{\mathrm{CPR}} = 4$ unless the full adder is embedded into a fully declared staged CPR model. The value

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

belongs to the isolated output quotient and is not identified here with a stage-wise CPR semantic-state count.

Chapters 7 and 8 preserve this distinction: the isolated full-adder factorization is verified as a finite output quotient, whereas the ripple-adder family receives a separate staged CPR reconstruction model with its own semantic-state 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; its staged arithmetic CPR realization is deferred to the ripple-adder model of Chapter 7. These structural boundaries provide candidate interfaces; they become certified CPR interfaces only after task sufficiency has been proved, and exact minimum CPR interfaces only after cardinality minimality has also been proved. Accordingly, every problem-class realization should establish, in order: an explicit component representation; a declared task; an admissible reconstruction ordering; a structural candidate interface; a reconstruction-safe signature satisfying S1–S3; task sufficiency of the candidate interface; cardinality minimality, if exact width equality is claimed; semantic-state and transition bounds; and all six complete runtime-cost bounds.

The realizations in this chapter do not establish that the unrestricted SAT, CSP, graph-coloring, tensor, or arithmetic problem classes possess small CPR-Width; that every structural candidate interface is minimal; that semantic signatures are always computable in polynomial time; that a favorable ordering is always polynomially constructible; or polynomial runtime from width alone. The supported conclusion is narrower:

> the combinatorial realizations provide problem-specific CPR interface models, while the isolated full adder provides a finite output-preserving quotient that motivates the staged arithmetic realization developed in Chapter 7 .

A tractability conclusion is permitted only for explicitly defined subclasses satisfying controlled width, BC1–BC4, and all six complete polynomial runtime bounds.

6.10 Conclusion

This chapter developed representative realizations of CPR-Width for SAT, CSP, graph coloring, tensor structures, and the isolated full adder. Table 6.1 records the corresponding problem-class-specific width expressions, $\omega_{\mathrm{rec}}(\Phi, \pi)$, $\omega_{\mathrm{rec}}(P_{\mathrm{CSP}}, \pi)$, $\omega_{\mathrm{rec}}(G, \pi)$, and $\omega_{\mathrm{rec}}(T, \pi)$, together with the finite output-quotient state count $S_{\mathrm{out}}(P_{\mathrm{FA}})$ recorded for the isolated full adder in place of a stage-wise width; these do not denote one fixed universal width for an entire unrestricted problem class, but identify the quantity that must be determined for the declared instance, task, model, and ordering.

A problem class may contain a controlled partially reconstructible subclass $\mathcal{C}' \subseteq \mathcal{C}$ whose instances admit a uniform ordering rule. For every encoded instance $P_x \in \mathcal{C}'$, the uniform ordering rule produces

$$\pi_x = A_\pi(x) \in \Pi_{\mathrm{adm}}(P_x, \tau; \mathfrak{M}),$$

with

$$\omega_{\mathrm{rec}}(P_x, \pi_x, \tau; \mathfrak{M}) \le w_{\mathcal{C}'}(\ell(P_x));$$

a constant-width subclass is obtained when $w_{\mathcal{C}'}(\ell(P_x)) = k_{\mathcal{C}'}$ for a fixed constant $k_{\mathcal{C}'}$. Neither controlled width nor constant width alone implies polynomial runtime. The complete sufficient condition remains

$$\mathbf{BC_1} \wedge \mathbf{BC_2} \wedge \mathbf{BC_3} \wedge \mathbf{BC_4} \wedge \boldsymbol{N_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) \in \ell(P_x)^{O(1)}}.$$

Thus, Chapter 6 identifies the structural realization of CPR-Width across several problem classes and formulates the theory under which controlled or constant-width subclasses may become completely CPR-tractable.

7 Adder and Arithmetic Reconstruction

Digital addition provides a finite and exactly verifiable realization of Controlled Partial Reconstruction. The full-adder and ripple-adder constructions are standard objects in digital arithmetic and combinational circuit design; their role here is not to introduce a new addition algorithm, but to verify that the CPR-WIDTH framework reproduces the known local and linear reconstruction structure of binary addition under an explicit component representation, semantic model, and complete operation-count analysis.

Two formally different constructions are distinguished. First, the isolated full adder admits an exact output-preserving quotient from eight input states to four Hamming-weight classes; these classes are treated as an output quotient and are not identified automatically with a stage-wise CPR semantic-state count. Second, the n-bit ripple-adder family admits a uniform staged transducer model whose future behavior is controlled by one Boolean carry state; the declared task is exact evaluation of the addition map

$$\tau_{\mathrm{add}} = \text{exact computation of } (s_0, s_1, \ldots, s_{n-1}, c_n) \text{ from } (A, B, c_0).$$

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

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

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

Proof. Let $h = a + b + c_{\mathrm{in}}$; since $h \in \{0, 1, 2, 3\}$, its parity determines the sum bit, $s = h \bmod 2$,

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

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 $d_{\text{rec}}^{\max}(P_{\text{FA}}) = 3$. 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

For the uniform n-bit transducer, define the reconstruction components $\gamma_1, \gamma_2, \ldots, \gamma_n$, where γ_j represents the carry state after processing bit positions $0, \ldots, j-1$; the component set is $V(P_{\text{add},n}) = \{\gamma_1, \ldots, \gamma_n\}$, every component with Boolean domain $D_{\gamma_j} = \mathbb{B}$. The input bits a_i, b_i and the initial carry c_0 are external input labels of a transducer run, not reconstructed component coordinates. The reconstruction ordering is $\pi_{\text{add}} = (\gamma_1, \gamma_2, \ldots, \gamma_n)$, so the number of nonterminal reconstruction stages is $m_{\text{stage}}(P_{\text{add},n}, \pi_{\text{add}}) = n$. After stage j, $V_j^{\pi_{\text{add}}} = \{\gamma_1, \ldots, \gamma_j\}$, $0 \leq j \leq n$, with $V_0^{\pi_{\text{add}}} = \varnothing$ and unresolved suffix $\overline{V}_j^{\pi_{\text{add}}} = \{\gamma_{j+1}, \ldots, \gamma_n\}$.

The uniform CPR model ranges over all input words of length n. A prefix carry assignment $r = (\gamma_1, \ldots, \gamma_j)$ is family-reachable when there exist local input labels $(a_0, b_0), \ldots, (a_{j-1}, b_{j-1})$ and an initial carry c_0 that generate the declared carry sequence; the family-level reachable prefix space is denoted $\mathcal{R}_j^{\pi_{\mathrm{add}},\mathrm{unif}}(P_{\mathrm{add},n})$. Both carry values are family-reachable at every nonterminal position, $\{0,1\} \subseteq \{\gamma_j(r) : r \in \mathcal{R}_j^{\pi_{\mathrm{add}},\mathrm{unif}}(P_{\mathrm{add},n})\}$. For one fixed encoded input (A, B, c_0), the deterministic transition rule selects exactly one carry sequence $c_1, c_2, \ldots, c_n$, so the corresponding fixed run contains exactly one reachable prefix state at every stage. The family-level state model and the fixed execution path must not be identified: the former characterizes the uniform CPR interface and semantic alphabet, the latter counts operations for one encoded input.

7.4 Carry Signature, Interface Sufficiency, and Minimality

For every stage $1 \le j < n$, define the family-level carry signature $\mathrm{Sig}^{\pi_{\mathrm{add}}}_{\tau_{\mathrm{add}},j}(r) = \gamma_j(r)$; at stage zero, the externally supplied initial carry is the current transducer state, and after the terminal carry γ_n has been materialized, the active interface may be released. For every nonterminal stage $1 \le j < n$, define the candidate interface $\widetilde{B}_j^{\pi_{\mathrm{add}},\tau_{\mathrm{add}}} = \{\gamma_j\}$, and at the initial and terminal stages $\widetilde{B}_0^{\pi_{\mathrm{add}},\tau_{\mathrm{add}}} = \widetilde{B}_n^{\pi_{\mathrm{add}},\tau_{\mathrm{add}}} = \varnothing$.

Proposition 7.1 (Carry-Interface Sufficiency)**.** *For every nonterminal stage $1 \le j < n$, the interface $\widetilde{B}_j^{\pi_{\mathrm{add}},\tau_{\mathrm{add}}} = \{\gamma_j\}$ is task-sufficient for the carry signature.*

Proof. Let r, r' be family-reachable prefix states at stage j satisfying $r|_{\{\gamma_j\}} = r'|_{\{\gamma_j\}}$; then $\gamma_j(r) = \gamma_j(r')$, so $\mathrm{Sig}^{\pi_{\mathrm{add}}}_{\tau_{\mathrm{add}},j}(r) = \gamma_j(r) = \gamma_j(r') = \mathrm{Sig}^{\pi_{\mathrm{add}}}_{\tau_{\mathrm{add}},j}(r')$. Hence the one-component interface is task-sufficient. □

Proposition 7.2 (Carry-Interface Minimality)**.** *For every nonterminal stage $1 \le j < n$, the empty set is not task-sufficient for the uniform ripple-adder transducer. Consequently $b_j^{\pi_{\mathrm{add}},\tau_{\mathrm{add}}}(P_{\mathrm{add},n}) = 1$.*

Proof. By family-level reachability, there exist reachable prefix states r_0, r_1 with $\gamma_j(r_0) = 0$ and $\gamma_j(r_1) = 1$; the two states agree on the empty interface. Choose the next local input label $(a_j, b_j) = (0, 0)$: from carry state 0, the local output is $s_j = 0$ and the successor carry is $\gamma_{j+1} = 0$; from carry state 1, the local output is $s_j = 1$ and the successor carry is again $\gamma_{j+1} = 0$. Thus the two prefix states induce different task-relevant next output behavior, so their reconstruction-safe signatures differ; the empty interface cannot distinguish them and is not task-sufficient. Since the singleton $\{\gamma_j\}$ is task-sufficient, its cardinality is minimal, so $b_j^{\pi_{\mathrm{add}},\tau_{\mathrm{add}}}(P_{\mathrm{add},n}) = 1$. □

It follows that $\widehat{\omega}_{\mathrm{rec}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = 1$ and, by the normalization convention, $\omega_{\mathrm{rec}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = 0$: normalized width zero still corresponds to one active Boolean carry component.

Proposition 7.3 (Carry Signature Is Reconstruction-Safe)**.** *The carry-signature family satisfies the reconstruction-safety conditions S1–S3 of Chapter 2.*

Proof. Let r, r' be family-reachable prefix states at the same stage j with equal carry signatures, $\gamma_j(r) = \gamma_j(r')$. For every common next input label $(a_j, b_j) \in \mathbb{B}^2$, the local arithmetic value $h_j = a_j + b_j + \gamma_j$ is determined; because the carry values agree, the same input label produces the same sum bit and successor carry, so equal carry signatures preserve the exact residual input-output behavior (S1). The complete local input alphabet $\mathbb{B}^2$ is available from both equal carry states, so the admissible next-label sets coincide (S2). The same next input label applied to equal carry states produces equal successor carry values, so the successor signatures coincide (S3). Hence the carry-signature family is reconstruction-safe. □

7.5 Evaluation, Semantic-State Control, and Runtime

Theorem 7.2 (Ripple-Adder Evaluation)**.** *For every $n \geq 1$ and every fixed encoded input (A, B, c_0), the output $(s_0, s_1, \ldots, s_{n-1}, c_n)$ is determined uniquely by the successive carry transitions.*

Proof. At position $i = 0$, the fixed values a_0, b_0, c_0 determine $h_0 = a_0 + b_0 + c_0$, and hence uniquely $s_0 = h_0 \bmod 2$ and $c_1 = \lfloor h_0/2 \rfloor$. Assume inductively that c_i has been computed correctly; then $h_i = a_i + b_i + c_i$ determines uniquely $s_i = h_i \bmod 2$ and $c_{i+1} = \lfloor h_i/2 \rfloor$. Thus every sum bit and the final carry are determined uniquely. □

The uniform carry-state alphabet contains two states, $S^{\mathrm{unif}}_{\mathrm{carry}} = |\mathcal{Q}_{\mathrm{carry}}| = 2$, a family-level state-alphabet count. For a fixed encoded input, the deterministic evaluation follows one carry state at each stage, so $|S^{\mathrm{run}}_j(A, B, c_0)| = 1$ for every stage of one fixed run, while the uniform family-level bound is $|S^{\mathrm{unif}}_j| \leq 2$; consequently $S^{\mathrm{unif}}_{\mathrm{CPR}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) \leq 2$ and $D^{\mathrm{unif}}_{\mathrm{CPR}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) \leq 1$.

Definition 7.1 (Total Stage-Indexed Semantic-State Count)**.** Define $V^{\mathrm{unif}}_{\mathrm{sem}}(P, \pi, \tau; \mathfrak{M}) = \sum_{j=0}^{m_{\mathrm{stage}}(P,\pi)} |S^{\mathrm{unif}}_j(P, \pi, \tau; \mathfrak{M})|$.

For the ripple-adder transducer, $V^{\mathrm{unif}}_{\mathrm{sem}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) \leq 2(n + 1)$.

Definition 7.2 (Total Labeled Semantic-Transition Count)**.** Let $\mathcal{E}^{\mathrm{sem}}_j(P, \pi, \tau; \mathfrak{M})$ denote the set of labeled semantic transitions from stage j to stage $j + 1$. Define $E^{\mathrm{unif}}_{\mathrm{sem}}(P, \pi, \tau; \mathfrak{M}) = \sum_{j=0}^{m_{\mathrm{stage}}(P,\pi)-1} |\mathcal{E}^{\mathrm{sem}}_j(P, \pi, \tau; \mathfrak{M})|$.

The uniform ripple-adder transducer has at most eight labeled transition edges per stage, so $E^{\mathrm{unif}}_{\mathrm{sem}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) \leq 8n$; for a fixed encoded input, exactly one transition edge is executed per bit position, $E^{\mathrm{run}}_{\mathrm{sem}}(A, B, c_0) = n$.

7.5.1 Construction Complexity of the Semantic Quotient

The ripple-adder family permits the construction cost of the semantic quotient to be determined explicitly.

The reconstruction-safe signature is the Boolean carry value

$$\mathrm{Sig}^{\pi_{\mathrm{add}}}_{\tau_{\mathrm{add}},j}(r) = \gamma_j(r) \in \{0, 1\}.$$

Hence, no enumeration of complete future continuations is required to identify the semantic class of a reachable prefix state.

The quotient state is obtained directly from the current carry, and the successor quotient state is computed locally from

$$\gamma_{j+1} = \left\lfloor \frac{a_j + b_j + \gamma_j}{2} \right\rfloor.$$

Thus, the semantic transition rule is an incremental constant-cost update of the form

$$\sigma_{j+1} = \Phi_{\mathrm{add}}(\sigma_j, a_j, b_j), \qquad \sigma_j = \gamma_j.$$

Thus, for this family, the abstract incremental signature-update mechanism discussed in Chapter 4 is realized explicitly by Φ_{add}; its evaluation requires only the current carry and the current local input pair, rather than the unresolved suffix.

The uniform quotient alphabet has constant cardinality, $|Q_{\mathrm{carry}}| = 2$, while the complete stage-indexed semantic representation satisfies

$$V^{\mathrm{unif}}_{\mathrm{sem}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) \leq 2(n + 1), \qquad E^{\mathrm{unif}}_{\mathrm{sem}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) \leq 8n.$$

Consequently, both the stage-indexed semantic-state representation and its labeled transition structure are constructible in $O(n)$ elementary operations under the declared model.

Proposition 7.4 (Linear Semantic-Quotient Construction)**.** *For the uniformly declared n-bit ripple-adder family, the semantic quotient and its stage-indexed transition structure can be constructed in $O(n)$ elementary operations and represented using $O(n)$ stage-indexed storage. The quotient alphabet itself remains constant, $|Q_{\mathrm{carry}}| = 2$.*

Proof. At every stage, the semantic class is determined by one Boolean carry value. The local transition table contains only the two carry states and four possible local input labels. Therefore, the local quotient rule has constant size and constant evaluation cost. There are n nonterminal bit positions. Instantiating the fixed local quotient rule over these positions requires $O(n)$ operations and $O(n)$ stage-indexed representation space. No suffix enumeration or exhaustive construction of future continuations is required. □

This construction bound is specific to the declared ripple-adder model and its explicit carry-signature rule. It is not inferred from constant CPR-Width alone and does not constitute a general semantic-quotient construction theorem for arbitrary CPR models.

The following bounds concern the evaluation of one fixed encoded input (A, B, c_0) by the uniformly declared ripple-adder model; the runtime analysis does not search over all task-sufficient interfaces, since the one-component carry interface is supplied explicitly and its sufficiency and minimality have already been proved, so no exponential interface-minimality search is included. The representation contains n carry components and n local input pairs; constructing the stage representation and the fixed ordering requires $N_{\mathrm{Build}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = O(n)$, and the fixed interface and signature rules are instantiated once per stage, so $N_{\mathrm{Reduce}} = O(n)$. One deterministic transition is executed per bit position, each using a constant number of Boolean and fixed-width arithmetic operations, so $N_{\mathrm{Transition}} = O(n)$; one sum bit is reconstructed at every bit position, followed by one final carry, so $N_{\mathrm{Reconstruct}} = O(n)$. The identity $a_i + b_i + c_i = s_i + 2c_{i+1}$ can be verified independently at every bit position, so $N_{\mathrm{Verify}} = O(n)$. The explicit output contains $n + 1$ bits, $L_{\mathrm{Output}} = n + 1$, giving $N_{\mathrm{Output}} = \Theta(n)$ under the elementary output model of Chapter 4. By the complete six-component decomposition, every component is $O(n)$ and the explicit output requires $\Omega(n)$ operations, so $N_{\mathrm{CPR}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = \Theta(n)$. This reproduces the known linear structure of ripple-carry addition inside the complete CPR accounting framework; it is not presented as a new addition algorithm. Its role is instead diagnostic: the complete CPR accounting shows that the linear growth does not arise from increasing reconstruction width or increasing semantic-alphabet size. Both remain constant. The remaining asymptotic growth is generated by the number of sequential reconstruction stages and the unavoidable explicit output length.

7.6 Boundary-Condition Record

The uniformly declared ripple-adder transducer satisfies the four CPR boundary conditions for the exact-output task.

BC1 – Explicit Uniform Structural Representation. The Boolean carry components, their local domains, the reconstruction ordering, the family-level reachability rule, the one-component active interfaces, the carry-signature family, the uniform semantic carry alphabet, the deterministic quotient transition, the local output rule, the verification identity, and the exact output representation are all explicitly defined; therefore BC1 holds.

BC2 – Polynomial Uniform Constructibility. The component representation, ordering, carry interface, transition table, and auxiliary stage data are constructible uniformly in $O(n)$ operations; therefore BC2 holds.

BC3 – Task-Relative Soundness and Completeness. For every bit position, $a_i + b_i + c_i = s_i + 2c_{i+1}$, and the inductive evaluation theorem proves that every fixed input produces exactly one correct output $(s_0, \ldots, s_{n-1}, c_n)$; therefore BC3 holds.

BC4 – Polynomial Semantic-State and Transition Control. The uniform family-level semantic quantities satisfy $V_{\mathrm{sem}}^{\mathrm{unif}} \leq 2(n + 1)$ and $E_{\mathrm{sem}}^{\mathrm{unif}} \leq 8n$; for every fixed encoded input, only

one state and one transition edge are used at each stage; therefore BC4 holds. Moreover, the semantic quotient does not require enumeration of the suffix space. By Proposition 7.4, the complete stage-indexed quotient representation and its transition structure are constructible in $O(n)$ operations. Thus, BC4 holds not only as a cardinality bound but also as an explicit uniform construction bound for this family.

Consequently $N_{\mathrm{CPR}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = \Theta(n)$, so the uniformly encoded ripple-adder family satisfies BC1–BC4 and the complete polynomial-cost requirements for the declared exact-output task.

7.7 Structural Interpretation, Scope, and Conclusion

The isolated full adder demonstrates the exact output-preserving factorization $\mathbb{B}^3 \xrightarrow{C_{\mathrm{FA}}} \{0,1,2,3\} \xrightarrow{Q_{\mathrm{FA}}} \mathbb{B}^2$, with fiber cardinalities $(1,3,3,1)$ and output-quotient count $S_{\mathrm{out}}(P_{\mathrm{FA}}) = 4$. The ripple-adder family demonstrates uniform staged reconstruction through the two-state carry transducer $(c_i, (a_i, b_i)) \mapsto (s_i, c_{i+1})$: at every nonterminal stage, the active task-sufficient component is the current Boolean carry component γ_j, so $\widehat{\omega}_{\mathrm{rec}} = 1$ and $\omega_{\mathrm{rec}} = 0$. The family-level model contains two possible carry states; a fixed encoded input follows one deterministic path through that two-state model.

7.7.1 Ripple Addition as a Resource-Separation Example

The ripple-adder family provides a concrete realization of the resource-separation interpretation of CPR-Width developed in Chapter 6.

Its active reconstruction requirement is constant,

$$\widehat{\omega}_{\mathrm{rec}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = 1, \qquad \omega_{\mathrm{rec}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = 0,$$

and its uniform semantic alphabet is also constant, $S^{\mathrm{unif}}_{\mathrm{CPR}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) \leq 2$.

Nevertheless, the number of reconstruction stages grows with the input length, $m_{\mathrm{stage}} = n$, the total stage-indexed semantic representation grows as $V^{\mathrm{unif}}_{\mathrm{sem}} = O(n)$, the semantic transition structure grows as $E^{\mathrm{unif}}_{\mathrm{sem}} = O(n)$, and the complete operation count satisfies $N_{\mathrm{CPR}} = \Theta(n)$.

Therefore,

$$\boxed{\text{constant CPR-Width } \not\equiv \text{ constant computational cost.}}$$

The width identifies the amount of task-relevant processed information that must remain simultaneously active; it does not count how many stages must be traversed. In particular, the normalized value $\omega_{\mathrm{rec}} = 0$ does not mean that no active information is required. Under the normalization convention of Chapter 2 it corresponds here to the nonempty one-component interface $\widehat{\omega}_{\mathrm{rec}} = 1$.

For the ripple-adder family, this distinction isolates the source of the remaining growth particularly clearly: computational cost grows because the fixed-width local reconstruction mechanism must be applied across n positions, not because the active reconstruction interface becomes larger.

This provides a diagnostic use of CPR-Width. The parameter separates simultaneous structural information from sequential reconstruction depth. Accordingly, the ripple-adder realization is not merely an example of a small-width computation. It demonstrates how CPR-Width can identify which computational resource remains controlled and which resource is responsible for asymptotic growth.

Diagnostic Interpretation. The family therefore illustrates a use of width that is independent of algorithmic novelty. The standard ripple-adder algorithm is already known to have linear runtime; CPR-WIDTH does not rediscover this fact as a new arithmetic algorithm. Instead, it

decomposes the reason for that scaling,

$$\underbrace{\widehat{\omega}_{\mathrm{rec}} = 1}_{\text{constant simultaneous dependency}} \qquad \text{versus} \qquad \underbrace{m_{\mathrm{stage}} = n}_{\text{growing sequential depth}} \quad .$$

This decomposition is the additional structural information supplied by the width analysis.

The arithmetic results establish exact output reconstruction for the isolated full-adder quotient and the uniformly encoded ripple-adder family defined in this chapter. They do not establish that every Boolean circuit admits a Hamming-weight factorization, that every arithmetic circuit possesses constant CPR-Width, that every circuit topology admits a one-component interface, that the finite ratio 8/4 is a measured hardware speedup, that the family-level carry alphabet is the same object as the state set visited during one fixed run, that a local quotient automatically extends to arbitrary circuit networks, or that structural compression alone guarantees a wall-clock advantage. The supported conclusions are limited to the exact isolated full-adder output quotient, the declared uniform ripple-adder transducer, the one-component carry interface, the exact normalized width, the two-state uniform carry alphabet, the linear family-level semantic bounds, and the linear complete operation count for one encoded input.

The uniform semantic representation satisfies $S_{\mathrm{CPR}}^{\mathrm{unif}} \leq 2$, $V_{\mathrm{sem}}^{\mathrm{unif}} \leq 2(n+1)$, $E_{\mathrm{sem}}^{\mathrm{unif}} \leq 8n$; all six runtime components are linearly bounded, and the explicit output requires linear work, so $N_{\mathrm{CPR}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = \Theta(n)$. These results verify the CPR accounting framework on a standard linear arithmetic process; they do not imply corresponding width or runtime properties for arbitrary Boolean or arithmetic circuits. The following chapter records the exhaustive finite verification of the isolated full-adder factorization and its output-equivalence fibers.

8 Finite Verification Records

The preceding chapter established the full-adder factorization symbolically and developed the ripple-adder family as a uniform arithmetic reconstruction model.

The present chapter provides three complementary verification records. First, it verifies the isolated full-adder Hamming-weight factorization by exhaustive enumeration of the complete finite input space. Second, it reports a measured reconstruction-counting experiment for the ripple-adder family, including exact brute-force cross-checks on small instances and scaling measurements on larger instances. Third, it reports an exploratory structural-interface experiment on random 3-SAT instances.

These records have deliberately different evidential roles. The full-adder record provides exact finite verification; the ripple-adder experiment provides implementation-specific runtime and memory evidence for the reconstruction procedure whose asymptotic properties were established independently in Chapter 7; and the random 3-SAT experiment provides structural pattern evidence for a setting in which the tested candidate interfaces remain large.

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

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

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

Finite verification serves two purposes: first, it checks the symbolic full-adder factorization of Chapter 7 against every element of the complete finite input space; second, it distinguishes between complete input states, reduced Hamming-weight states, projection fibers, output-

equivalence classes, and full-adder output states. The exhaustive full-adder verification remains finite throughout and does not establish an asymptotic tractability result, an operation-count advantage, or a stage-wise CPR-Width theorem. The later empirical sections provide separate measured runtime, memory, and structural evidence under explicitly stated protocols and evidence levels.

Let $\mathbb{B} = \{0, 1\}$; the complete local input space is $X_{\mathrm{FA}} = \mathbb{B}^3$, so $|X_{\mathrm{FA}}| = 2^3 = 8$. Every input state has the form $x = (a, b, c_{\mathrm{in}})$, $a, b, c_{\mathrm{in}} \in \mathbb{B}$, and the full-adder output map is $F_{\mathrm{FA}} : X_{\mathrm{FA}} \to \mathbb{B}^2$, with $F_{\mathrm{FA}}(x) = (s, c_{\mathrm{out}})$, 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 $h = a + b + c_{\mathrm{in}}$. 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}	$Q_{\mathrm{FA}}(h)$
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 $a + b + c_{\mathrm{in}} = s + 2c_{\mathrm{out}}$ holds, and the last three columns also show directly that $F_{\mathrm{FA}}(x) = Q_{\mathrm{FA}}(h)$ for every one of the eight possible input states.

8.3 Hamming-Weight Projection and Fibers

Define the Hamming-weight projection $C_{\mathrm{FA}} : \mathbb{B}^3 \to \{0, 1, 2, 3\}$ by $C_{\mathrm{FA}}(a, b, c_{\mathrm{in}}) = a + b + c_{\mathrm{in}}$, so the reduced state space $Z_{\mathrm{FA}} = \{0, 1, 2, 3\}$ has $|Z_{\mathrm{FA}}| = 4$, the finite projection cardinality pattern $8 \to 4$. Define the output-reconstruction map $Q_{\mathrm{FA}} : Z_{\mathrm{FA}} \to \mathbb{B}^2$ by $Q_{\mathrm{FA}}(h) = (h \bmod 2, \lfloor h/2 \rfloor)$; the row-wise verification of Table 8.1 yields $F_{\mathrm{FA}} = Q_{\mathrm{FA}} \circ C_{\mathrm{FA}}$. The map C_{FA} is not injective, so 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 $h \in Z_{\mathrm{FA}}$, define the projection fiber $\Omega_{\mathrm{FA}}(h) = C_{\mathrm{FA}}^{-1}(\{h\})$, which contains every complete input state mapped to the same reduced Hamming-weight state (it does not define a unique inverse of C_{FA}). The four fibers are $\Omega_{\mathrm{FA}}(0) = \{(0, 0, 0)\}$, $\Omega_{\mathrm{FA}}(1) = \{(1, 0, 0), (0, 1, 0), (0, 0, 1)\}$, $\Omega_{\mathrm{FA}}(2) = \{(1, 1, 0), (1, 0, 1), (0, 1, 1)\}$, and $\Omega_{\mathrm{FA}}(3) = \{(1, 1, 1)\}$, with cardinalities $(1, 3, 3, 1)$. The fibers form a partition of the complete input space, $\mathbb{B}^3 = \bigsqcup_{h \in Z_{\mathrm{FA}}} \Omega_{\mathrm{FA}}(h)$, and indeed $\sum_{h \in Z_{\mathrm{FA}}} |\Omega_{\mathrm{FA}}(h)| = 1+3+3+1 = 8$; the maximum projection-fiber size is $\max_{h \in Z_{\mathrm{FA}}} |\Omega_{\mathrm{FA}}(h)| = 3$. This is a finite fiber-cardinality statement, 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 $F_{\mathrm{FA}} = Q_{\mathrm{FA}} \circ C_{\mathrm{FA}}$ 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 mod 2, and its integer quotient by two determines the outgoing carry, $c_{\text{out}} = \lfloor h/2 \rfloor$. Thus $F_{\text{FA}}(x) = (h \bmod 2, \lfloor h/2 \rfloor) = Q_{\text{FA}}(h) = Q_{\text{FA}}(C_{\text{FA}}(x))$. 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 $x \equiv_{\text{out}} y \Leftrightarrow F_{\text{FA}}(x) = F_{\text{FA}}(y)$; this relation compares complete input states according to their output behavior and is not the general stage-wise CPR semantic equivalence of Chapter 3.

Lemma 8.1 (Equality of Output and Hamming-Weight Classes)**.** *For all $x, y \in \mathbb{B}^3$, $F_{\text{FA}}(x) = F_{\text{FA}}(y) \Leftrightarrow C_{\text{FA}}(x) = C_{\text{FA}}(y)$.*

Proof. If $C_{\text{FA}}(x) = C_{\text{FA}}(y)$, the factorization $F_{\text{FA}} = Q_{\text{FA}} \circ C_{\text{FA}}$ gives $F_{\text{FA}}(x) = F_{\text{FA}}(y)$. Conversely, the values of Q_{FA} are $Q_{\text{FA}}(0) = (0,0)$, $Q_{\text{FA}}(1) = (1,0)$, $Q_{\text{FA}}(2) = (0,1)$, $Q_{\text{FA}}(3) = (1,1)$, which are pairwise distinct, so $Q_{\text{FA}} : Z_{\text{FA}} \to \mathbb{B}^2$ is bijective and in particular injective; hence $F_{\text{FA}}(x) = F_{\text{FA}}(y)$ implies $C_{\text{FA}}(x) = C_{\text{FA}}(y)$. □

The output quotient $S_{\text{FA}}^{\text{out}} = \mathbb{B}^3/\equiv_{\text{out}}$ therefore has classes precisely the Hamming-weight fibers, $S_{\text{FA}}^{\text{out}} = \{\Omega_{\text{FA}}(0), \ldots, \Omega_{\text{FA}}(3)\}$, so $|S_{\text{FA}}^{\text{out}}| = 4$. The binary output-quotient dimension is $D_{\text{FA}}^{\text{out}} = \lceil \log_2 \max\{1, |S_{\text{FA}}^{\text{out}}|\} \rceil = \lceil \log_2 4 \rceil = 2$; $|S_{\text{FA}}^{\text{out}}| = 4$ counts the output-equivalence classes, and $D_{\text{FA}}^{\text{out}} = 2$ counts the bits required to identify one of those four classes. Neither quantity is asserted here to equal a stage-wise quantity $S_{\text{CPR}}(P_{\text{FA}}, \pi, \tau; \mathfrak{M})$ or $D_{\text{CPR}}(P_{\text{FA}}, \pi, \tau; \mathfrak{M})$, 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: 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 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 does not establish an asymptotic theorem for an unrestricted circuit family.

Under the evidence hierarchy introduced in Chapter 9, the present full-adder record has finite-verification status, denoted Evidence Level E1.

Evidence Status: E1 — Finite Verification Evidence

Evidence Level E1 means that a claim is checked on a finite, explicitly stated example. The present record qualifies 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, E1 $\not\Longrightarrow$ E4, where E4 denotes an asymptotic theorem under explicit assumptions. Likewise, $8 \to 4$ 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 of this chapter verify the full-adder factorization exhaustively but do not measure a runtime, and no section of this manuscript prior to this one provides Evidence Level E5 (Measured Runtime Evidence, Chapter 9) support for any claim. This section reports two

small, reproducible computational experiments: an exact reconstruction-counting task for the ripple-adder family compared against a brute-force baseline, and an interface-size measurement on random 3-SAT instances. Both experiments are illustrative, not a benchmark suite, and neither is compared against a production SAT or arithmetic solver; the explicit non-claims are restated in Section 8.6.4.

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 measured with Python's `tracemalloc` module and reflects Python-level allocations, not full process resident set size. Each reported wall-clock and memory figure is a single run per input size (no averaging over repeated trials), so figures should be read as order-of-magnitude evidence, not high-precision benchmarking. This is a shared, virtualized sandbox rather than dedicated benchmarking hardware, so absolute timings are not claimed to be reproducible to within a small constant factor on other hardware; the scaling *pattern* across input sizes, and the exact correctness cross-checks, are the load-bearing claims. The three scripts used to produce every number in this section (`cpr_reconstruction.py`, `bruteforce_numpy.py`, `sat_interface_growth.py`) are included with this manuscript in the `code/` directory so the measurements can be rerun directly rather than taken on faith.

8.6.2 Ripple-Adder Reconstruction Counting

The task is: given a target sum-bit sequence $s = (s_0, \ldots, s_{n-1})$ and fixed $c_0 = 0$, count the number of admissible input sequences $(a_i, b_i)_{i=0}^{n-1}$ whose ripple-carry addition reproduces s exactly, with the final carry c_n left unrestricted. This is a genuine reconstruction-counting task in the sense of this manuscript, and it is distinct from the exact-evaluation task τ_{add} of Chapter 7, which computes $(s_0, \ldots, s_{n-1}, c_n)$ from a fixed (A, B, c_0): here s is fixed and the input pairs are counted, not evaluated forward from a fixed input. The task-relative structure of CPR-WIDTH requires this distinction to be kept explicit.

CPR method. Maintain a distribution over the carry semantic state $c \in \{0, 1\}$ (Proposition 7.3), with multiplicity equal to the number of admissible partial input sequences reaching that carry value; at each stage, extend by the (at most two, per current carry value) input pairs (a_i, b_i) consistent with the required s_i, using the $O(1)$ upd_i of Section 2.7.1. The semantic-state count never exceeds 2 at any stage. This agrees with the carry realization established in Chapter 7: the unresolved suffix depends on one Boolean carry component, corresponding to an unnormalized interface cardinality of one, normalized reconstruction width $\omega_{\text{rec}} = 0$, and at most two semantic carry states.

Brute-force baseline. Enumerate all 2^{2n} raw input sequences directly and check, for each, whether ripple-carry addition reproduces s; no semantic compression is used.

Correctness cross-check. For every $n \in \{4, 6, 8, 10, 12\}$, the CPR reconstruction count was verified to equal the brute-force count exactly (16, 64, 256, 1024, 4096 respectively, in each case 2^n). This cross-check is Evidence Level E1 (Finite Verification Evidence) support for the correctness of the CPR reconstruction procedure on these five instances; it is not a proof for all n. Theorem 7.2 (Ripple-Adder Evaluation, Chapter 7) does not establish this count: it proves only the forward direction, that every fixed input (A, B, c_0) determines a unique output, for the exact-evaluation task τ_{add}, whereas the present claim concerns the reconstruction-counting task defined above. The exact count is instead given by the following proposition.

Proposition 8.1 (Ripple-Adder Reconstruction Count)**.** *Let $s = (s_0, \ldots, s_{n-1}) \in \mathbb{B}^n$ be a fixed target sum-bit sequence and let $c_0 = 0$. If the final carry c_n is unrestricted, then the number of*

input-pair sequences $(a_i, b_i)_{i=0}^{n-1}$ *whose ripple-carry addition reproduces* s *is exactly*

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

Proof. At every stage i, for each fixed incoming carry c_i and prescribed target bit s_i, exactly two of the four pairs $(a_i, b_i) \in \mathbb{B}^2$ satisfy $a_i \oplus b_i \oplus c_i = s_i$, namely the two pairs with $a_i \oplus b_i = s_i \oplus c_i$; each such choice determines the successor carry $c_{i+1} = \lfloor (a_i + b_i + c_i)/2 \rfloor$ uniquely. Thus every admissible partial input sequence has exactly two admissible extensions at the next stage, independently of the carry value reached. Starting from the fixed initial carry $c_0 = 0$, induction over the n stages gives exactly 2^n admissible input-pair sequences. □

This is what the five cross-checked instances confirm numerically.

Table 8.2: Ripple-adder reconstruction counting: CPR (carry semantic state) versus brute-force (full raw enumeration), including peak memory measured with `tracemalloc`. Brute-force beyond $n = 12$ was not run; 2^{2n} is shown analytically to indicate why.

n	CPR t. (s)	CPR mem. (B)	max sem. states	BF t. (s)	BF mem. (B)	2^{2n} enum.
4	2.5×10^{-5}	244	2	1.8×10^{-3}	9,264	256
8	4.6×10^{-5}	268	2	2.5×10^{-3}	1,442,752	65,536
12	8.3×10^{-5}	324	2	3.8×10^{-1}	92,275,688	16,777,216
20	1.3×10^{-4}	380	2	not run	not run	1.1×10^{12}
1,000	5.9×10^{-3}	1,116	2	not run	not run	$\approx 10^{602}$
100,000	9.8×10^{-1}	67,116	2	not run	not run	$\approx 10^{60,206}$

The maximum observed semantic-state count is 2 for every tested n, consistent with the constant-state-count theorem of Chapter 7 (Evidence Level E4, established there as an asymptotic structural result and not re-derived from these measurements), while the brute-force raw state space grows as 2^{2n}; at $n = 12$ the brute-force run already takes 3.8×10^{-1} s and 92,275,688 bytes of peak memory, against 8.3×10^{-5} s and 324 bytes for CPR on the same instance size, and by $n = 20$ the brute-force space ($\approx 1.1 \times 10^{12}$) is, by this analytical estimate rather than a measured run, already impractical to enumerate on this hardware, while CPR continues to $n = 100{,}000$ in under one second using 67,116 bytes of peak memory. This experiment provides Evidence Level E5 support for the measured runtime claim under the stated environment and protocol, together with measured peak-memory evidence for the same implementation, for the specific claim that the measured CPR runtime and traced Python-allocation footprint remain small on instance sizes where brute-force reconstruction is already impractical, and Evidence Level E1 support for the correctness of the count on the five cross-checked instances. The structural explanation for why the semantic-state count does not grow with n is the separate E4 asymptotic-theorem claim of Chapter 7; these measurements corroborate it but do not themselves establish it.

8.6.3 Interface Growth on Random 3-SAT

For n Boolean variables, $m = \lfloor 4.2n \rfloor$ random 3-clauses were generated (near the empirical satisfiability phase transition for random 3-SAT), and the structural candidate interface $\widetilde{B}_i^{\pi}$ (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 (a heuristic, not a claim of ordering optimality). Table 8.3 reports the mean, over eight random instances at $n = 14$, of the maximum tracked interface size under each ordering.

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. It shows that the particular boundary realization $\widetilde{B}_i^{\pi}$ used here does not expose a small active boundary on these instances under the tested orderings.

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

Ordering	Mean max interface size (of $n = 14$)
Natural	11.75
Random	11.62
Greedy (local)	11.50

The experiment does not establish that the exact minimum task-sufficient CPR interface is equally large, because $\widetilde{B}_i^{\pi}$ is a structural candidate interface rather than a proved cardinality-minimal interface. Nor does it exclude the existence of a substantially better admissible ordering outside the three ordering strategies tested here.

Its significance is therefore diagnostic rather than complexity-theoretic: the experiment demonstrates how the CPR framework can reveal when a proposed structural realization fails to produce the small-boundary behavior observed for cycles and ripple-adder chains.

This experiment provides Evidence Level E2 (Structural Pattern Evidence) for that observed behavior on the sampled instances. It is not E4 evidence and makes no claim for all n, all clause-to-variable ratios, all random 3-SAT distributions, or all admissible orderings.

8.6.4 Scope and Non-Claims of This Section

This section does not constitute a benchmark suite: instance sizes are small ($n \leq 12$ for exact brute-force cross-checking, $n \leq 14$ for the SAT experiment, up to $n = 100{,}000$ for the CPR-only adder scaling curve), no comparison is made against a production CDCL/DPLL SAT solver or an optimized arithmetic circuit tool, the reported wall-clock timing measurements are single-run measurements rather than averages over repeated timing trials, whereas the random 3-SAT interface-growth values reported in Table 8.3 are means over the eight generated instances, and the hardware is a shared virtualized sandbox rather than dedicated benchmarking hardware. No claim is made that these two experiments characterize CPR-Width's behavior on arbitrary problem instances. No operation-count comparison (G_{count}, Chapter 4) and no comparison against an optimized classical or production implementation is reported here; the brute-force enumeration used above serves only as an exact raw-state baseline. The supported conclusions are exactly the ones stated in Sections 8.6.2 and 8.6.3: a measured wall-clock, peak-memory, and constant-state-count illustration for the ripple-adder reconstruction task on this implementation and hardware, and a measured interface-size pattern for random 3-SAT under three specific orderings on eight random instances.

8.7 Scope, Non-Claims, and Conclusion

This chapter provides three distinct forms of evidence.

First, it gives exhaustive finite verification of the isolated full-adder output quotient.

Second, it provides a measured reconstruction-counting illustration for the ripple-adder family, including exact brute-force cross-checking on small instances and CPR-only scaling measurements on larger instances.

Third, it reports an exploratory structural experiment on random 3-SAT instances, measuring the maximum candidate-interface size under three specified reconstruction orderings.

These three records have different evidential status and must not be conflated. The full-adder record establishes exact finite properties; the ripple-adder experiment supplies implementation-specific measured runtime and memory evidence in addition to the independent family-level theorems of Chapter 7; and the random 3-SAT experiment supplies only an observed structural pattern on the sampled instances.

The complete isolated full-adder input space has been verified exhaustively. The Hamming-weight

projection is $\mathbb{B}^3 \xrightarrow{C_{\mathrm{FA}}} \{0,1,2,3\}$, with fiber-cardinality vector $(1,3,3,1)$, output-reconstruction map $Q_{\mathrm{FA}}(h) = (h \bmod 2, \lfloor h/2 \rfloor)$, and exact factorization $F_{\mathrm{FA}} = Q_{\mathrm{FA}} \circ C_{\mathrm{FA}}$. Every input state in the same projection fiber induces the same output pair; conversely, because Q_{FA} is injective, two input states with the same output belong to the same Hamming-weight fiber, so $\mathbb{B}^3/\equiv_{\mathrm{out}} = \{\Omega_{\mathrm{FA}}(0), \ldots, \Omega_{\mathrm{FA}}(3)\}$.

The exhaustive full-adder record constitutes exact finite verification of the Hamming-weight factorization, the projection fibers, the output-preservation property, and the output-equivalence quotient.

The empirical ripple-adder record additionally demonstrates, for the specified implementation and execution environment, that the carry-state reconstruction procedure remains computationally small in both measured runtime and peak memory over the tested scaling range, while exact brute-force enumeration becomes rapidly infeasible. This is measured implementation evidence, not a hardware-independent speedup theorem.

The random 3-SAT record provides a complementary negative structural observation: under the three tested orderings, the observed structural candidate interface remains large relative to the number of variables. This supports the diagnostic interpretation of CPR-Width developed in Chapters 6 and 7: width-oriented interface analysis can identify not only structurally favorable realizations but also realizations for which the selected reconstruction representation fails to expose a small active boundary.

None of these experiments establishes a universal CPR advantage, a general polynomial-time result for SAT, or a tractability conclusion for an unrestricted problem class.

The following chapter 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, τ the declared computational task, $\mathfrak{M}$ the fixed CPR reconstruction model, and $\pi \in \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M})$ an admissible reconstruction ordering. The central structural-runtime chain of the manuscript is

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

the first quantity is the normalized CPR-Width, the second the maximum number of semantic quotient states occurring at one reconstruction stage, and the third the complete CPR operation count; when unambiguous, dependence on τ and $\mathfrak{M}$ may be suppressed. This chain expresses structural and accounting dependencies; it does not state that small reconstruction width alone implies polynomial-time tractability. As established in Chapter 4, the complete CPR operation count is $N_{\mathrm{CPR}} = N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} + N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}$; a favorable value of ω_{rec} controls only one structural contribution to the complete analysis, and a polynomial-time conclusion requires uniform polynomial control of every term. Let $\mathcal{J} = \{\mathrm{Build}, \mathrm{Reduce}, \mathrm{Transition}, \mathrm{Reconstruct}, \mathrm{Verify}, \mathrm{Output}\}$ be the runtime-index set.

Let $\mathcal{F} = \{P_x : x \in \{0,1\}^*\}$ be a uniformly encoded family of finite decision-problem instances with associated decision language $L_{\mathcal{F}} = \{x \in \{0,1\}^* : P_x \text{ is a yes-instance}\}$. A complete uniform CPR decision model requires one fixed decision task τ, one fixed uniform reconstruction model $\mathfrak{M}$, one deterministic preprocessing algorithm $A_{\mathrm{Preprocess}}$ containing an ordering-supply component A_π, as specified in Chapter 5, and six fixed polynomial functions $\{p_j\}_{j \in \mathcal{J}}$.

Soundness and completeness are already required uniformly through BC3 and therefore do not constitute a separately quantified object.

For every encoding x, let

$$\mathcal{D}_x = A_{\text{Preprocess}}(x)$$

denote the complete uniformly constructed CPR representation of P_x, and let

$$\pi_x = A_\pi(x) \in \Pi_{\text{adm}}(P_x, \tau; \mathfrak{M})$$

denote the ordering supplied by the ordering component of $A_{\text{Preprocess}}$.

Every elementary preprocessing operation is assigned exactly once to either N_{Build} or N_{Reduce}, according to the disjoint accounting convention of Chapters 4–5. In particular, the operation count of the ordering-supply component A_π is included in N_{Build}.

The complete uniform tractability requirement is

$$\Big[\exists \tau\, \exists \mathfrak{M}\, \exists A_{\text{Preprocess}}\, \exists \{p_j\}_{j\in\mathcal{J}}\, \forall x \in \{0,1\}^* : \mathcal{D}_x = A_{\text{Preprocess}}(x) \wedge \pi_x = A_\pi(x) \in \Pi_{\text{adm}}(P_x, \tau; \mathfrak{M}) \wedge \bigwedge_{k=1}^{4} \text{BC}_k(P_x, \pi_x, \tau; \mathfrak{M}) \wedge \bigwedge_{j\in\mathcal{J}} [N_j(P_x, \pi_x, \tau; \mathfrak{M}) \le p_j(|x|)]\Big] \Longrightarrow L_{\mathcal{F}} \in \mathbf{P}.$$

Here A_π denotes the ordering-supply component of $A_{\text{Preprocess}}$ and is not separately quantified. The order of quantifiers 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 $\mathbf{P}_{\text{CPR}}$ in Chapter 5, which includes BC1–BC4, uniform ordering supply, and polynomial bounds for all six runtime components,

$$\mathbf{P}_{\text{CPR}} \subseteq \mathbf{P}.$$

This inclusion is not a complexity-class collapse statement. For an NP-complete language L^*, the implication

$$L^* \in \mathbf{P}_{\text{CPR}} \Longrightarrow \mathbf{P} = \mathbf{NP}$$

remains strictly conditional, and no NP-complete language is proved in this work to belong to $\mathbf{P}_{\text{CPR}}$.

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 and peak-memory data, exact brute-force cross-checks on small adder instances, and a structural interface-growth experiment on random 3-SAT instances. 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, while a measured runtime claim requires a separate implementation, hardware description, software environment, benchmark protocol, and reproducible measurement procedure. The remaining sections formulate the principal mathematical, algorithmic, and experimental research problems of CPR-WIDTH.

9.1 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.

9.1.1 Optimal Reconstruction Orderings

Let $\Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})$ denote the set of admissible reconstruction orderings, assumed nonempty. The minimum normalized CPR-Width is $\omega_{\mathrm{rec}}(P,\tau;\mathfrak{M}) = \min_{\pi\in\Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})} \omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})$, with corresponding unnormalized minimum $\widehat{\omega}_{\mathrm{rec}}(P,\tau;\mathfrak{M}) = \min_{\pi} \widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})$. The definition does not imply that a minimizing ordering can be found efficiently; the exact ordering problem is to determine $\pi^* \in \Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})$ achieving this minimum. A complete algorithmic analysis should distinguish: whether the exact minimum width can be computed in polynomial time for a specified uniformly encoded family; whether one can determine whether $\omega_{\mathrm{rec}} \le k$ for a declared integer k; whether a polynomial-time approximation algorithm exists when exact minimization is hard; whether fixed-parameter algorithms are possible with respect to $\widehat{\omega}_{\mathrm{rec}}$, $q(P)$, 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.

9.1.2 Width Ratio

Let $\widetilde{\pi} \in \Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})$ be a declared admissible ordering; the tilde marks an arbitrary declared ordering symbol here and must not be confused with the hat accent used elsewhere for unnormalized width quantities such as $\widehat{\omega}_{\mathrm{rec}}$. Define the unnormalized width ratio $\alpha_{\mathrm{width}}(P,\widetilde{\pi},\tau;\mathfrak{M}) = (\widehat{\omega}_{\mathrm{rec}}(P,\widetilde{\pi},\tau;\mathfrak{M})+1)/(\widehat{\omega}_{\mathrm{rec}}(P,\tau;\mathfrak{M})+1)$; adding one guarantees a positive denominator while preserving the exact optimality criterion. This is a structural width ratio, not an operation-count ratio or a measured runtime overhead.

Proposition 9.1 (Width-Ratio Bound)**.** *For every admissible ordering $\widetilde{\pi}$, $\alpha_{\mathrm{width}}(P,\widetilde{\pi},\tau;\mathfrak{M}) \ge 1$, with equality if and only if $\widehat{\omega}_{\mathrm{rec}}(P,\widetilde{\pi},\tau;\mathfrak{M}) = \widehat{\omega}_{\mathrm{rec}}(P,\tau;\mathfrak{M})$.*

Proof. By definition of the minimum unnormalized width,

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

adding one to both sides and dividing by the strictly positive denominator gives

$$\alpha_{\mathrm{width}}(P,\widetilde{\pi},\tau;\mathfrak{M}) \ge 1,$$

with equality exactly when the two unnormalized width values are equal. □

9.1.3 Efficient Semantic-State Construction

At stage i, the semantic quotient is $S_i(P,\pi,\tau;\mathfrak{M}) = U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})/\equiv^{\tau,\mathfrak{M}}_{P,\pi,i}$, with primary equivalence $u \equiv^{\tau,\mathfrak{M}}_{P,\pi,i} v \Leftrightarrow \widehat{\mathrm{Sig}}^{\pi}_{\tau,i}(u) = \widehat{\mathrm{Sig}}^{\pi}_{\tau,i}(v)$, where $\widehat{\mathrm{Sig}}^{\pi}_{\tau,i} : U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}) \to \Sigma^{\pi}_{\tau,i}$ is the induced reconstruction-safe interface-signature map: the equivalence relation is defined by reconstruction-safe signatures, not generally by literal equality of complete 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, preserves S1–S3, and the declared task requires that stronger information.

The principal algorithmic question is whether semantic classes can be represented by a finite efficiently computable canonical encoding. Let $\chi_i : U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}) \to \Gamma_i(P,\pi,\tau;\mathfrak{M})$ be a finite encoding map (the symbol χ_i avoids conflict with the stage-wise branching number λ_i of Chapter 4 and the admissible next-value sets $\Lambda_i^{\pi}(r)$). The encoding is exact if $\chi_i(u) = \chi_i(v) \Leftrightarrow u \equiv^{\tau,\mathfrak{M}}_{P,\pi,i} v$; under this condition, the fibers of χ_i coincide exactly with the semantic equivalence classes, giving a natural bijection $S_i(P,\pi,\tau;\mathfrak{M}) \cong \chi_i(U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}))$. 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 $\chi_i(u) = \chi_i(v) \Rightarrow u \equiv^{\tau,\mathfrak{M}}_{P,\pi,i} v$, 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, so $|\chi_i(U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}))| \geq |S_i(P,\pi,\tau;\mathfrak{M})|$. If this implication fails, the encoding may merge states whose reconstruction-safe future behavior differs, which can destroy soundness, completeness, admissible-next-value preservation, or successor congruence. 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 above remains open for arbitrary uniformly encoded families, but 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, so at every bit position the semantic state belongs to the fixed set $\{0,1\}$. The successor signature is obtained locally from the current carry and the current input bits via the incremental update $\sigma_{j+1} = \Phi_{\mathrm{add}}(\sigma_j, a_j, b_j)$ of Chapter 7, which specializes the abstract incremental signature-update mechanism of Chapter 4. Consequently, the quotient state set has constant cardinality and its stage-wise update is local; over an n-bit ripple adder, the complete sequence of quotient-state updates is therefore constructible in $O(n)$ elementary operations, subject to the operation-cost conventions and the model-specific caveat established in Chapter 7 (Proposition 7.4): this is a resolved instance of the construction problem for one declared family, not a general polynomial-construction theorem for arbitrary CPR models.

9.1.4 Width-Bound Slack

The sharp operational Width–State Bound is $S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \leq q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}$. Since $S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \geq 1$, define the sharp width-bound slack factor $\kappa_{\mathrm{bound}}(P,\pi,\tau;\mathfrak{M}) = q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} / S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M})$.

Proposition 9.2 (Width-Bound Slack)**.** *For every finite problem instance, declared task, reconstruction model, and admissible ordering,* $\kappa_{\mathrm{bound}}(P,\pi,\tau;\mathfrak{M}) \geq 1$.

Proof. By the Width–State Bound, $S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \leq q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}$; dividing the upper bound by the positive semantic-state count gives $\kappa_{\mathrm{bound}}(P,\pi,\tau;\mathfrak{M}) \geq 1$. □

A value 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 much smaller state space than the raw width bound. The slack factor is not a measured compression speedup, operation-count gain, or wall-clock gain.

9.1.5 Relationship to Classical Width Parameters

CPR-Width is not defined as treewidth, pathwidth, branchwidth, clique-width, or hypertree width. Nevertheless, specific relationships may exist for explicitly declared problem representations and reconstruction tasks. Let $G_P = \mathcal{G}(P)$ be the explicitly declared graph representation associated with P (a primal graph, incidence graph, constraint graph, circuit graph, or another specified construction, depending on the problem class), and let $\mathrm{tw}(G_P)$, $\mathrm{pw}(G_P)$, $\mathrm{bw}(G_P)$, $\mathrm{cw}(G_P)$ denote treewidth, pathwidth, branchwidth, and clique-width. Future research should investigate whether inequalities of the form $\omega_{\mathrm{rec}}(P,\tau;\mathfrak{M}) \leq f(\mathrm{tw}(G_P))$ or $\mathrm{tw}(G_P) \leq g(\omega_{\mathrm{rec}}(P,\tau;\mathfrak{M}))$ hold for explicitly defined functions $f, g : \mathbb{N}_0 \to \mathbb{N}_0$; similar questions may be posed for pathwidth, branchwidth, clique-width, hypertree width, trellis width, contraction width, and other problem-class-specific width notions. No universal comparison is claimed: any theorem relating CPR-Width to a classical width parameter must fix the problem representation, the declared task, the reconstruction model, the admissible-ordering family, the active-interface rule, the reconstruction-safe signature family, and the graph or relational representation $\mathcal{G}(P)$, since the same underlying graph may induce different CPR-Width values

for decision, counting, enumeration, optimization, or exact reconstruction tasks. This open problem, together with the informal discussion of treewidth and pathwidth in Chapter 1, is the precise sense in which a formal structural comparison between CPR-Width and classical width parameters is left for future work rather than established in this manuscript. The general open problem above may be formulated for the normalized width ω_{rec}, consistent with its use in the boundary conditions of Chapter 5. The concrete conditional comparison with vertex separation below is instead stated for the unnormalized width $\widehat{\omega}_{\mathrm{rec}}$, because vertex separation is itself an unnormalized boundary cardinality. The two width forms must therefore be distinguished according to the normalization convention of Chapter 2:

$$\omega_{\mathrm{rec}} = \max\{0, \widehat{\omega}_{\mathrm{rec}} - 1\}.$$

Every admissible reconstruction ordering π in this manuscript is a *linear* ordering of components: components are processed strictly one after another. For a graph-representation-based realization (Section 6.3), the *structural candidate interface* $\widetilde{B}_i^{\pi}$ associated with such a linear ordering (Chapter 6: processed vertices still adjacent to an unprocessed vertex) is naturally comparable to the boundary of that linear vertex ordering of G_P, and the minimum, over all linear vertex orderings, of the maximum boundary size is a classical quantity known as the *vertex separation number* of G_P; Kinnersley's theorem [4] identifies the vertex separation number with $\mathrm{pw}(G_P)$, the pathwidth of G_P. The structural candidate interface $\widetilde{B}_i^{\pi}$ must not be identified with the certified CPR interface $B_i^{\pi,\tau}$: the latter is, by definition, a cardinality-*minimal* interface that is proved task-sufficient for the declared signature family, whereas $\widetilde{B}_i^{\pi}$ is only a structural candidate. For any fixed admissible linear ordering π whose structural candidate boundary is proved task-sufficient for the declared task, minimality of $B_i^{\pi,\tau}$ gives

$$|B_i^{\pi,\tau}| \le |\widetilde{B}_i^{\pi}|$$

at every stage, and therefore, for every such π,

$$\widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) \;\le\; \max_i |\widetilde{B}_i^{\pi}|.$$

Consequently, if there exists an ordering $\pi^* \in \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M})$ that attains the vertex-separation number of G_P (i.e., $\max_i |\widetilde{B}_i^{\pi^*}| = \mathrm{pw}(G_P)$) and whose structural candidate boundary is task-sufficient at every stage, then

$$\widehat{\omega}_{\mathrm{rec}}(P, \tau; \mathfrak{M}) \;\le\; \mathrm{pw}(G_P),$$

since $\widehat{\omega}_{\mathrm{rec}}(P, \tau; \mathfrak{M})$ minimizes over all of $\Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M})$, including π^*. More generally, and without assuming π^* is admissible, every admissible ordering whose structural candidate boundary is task-sufficient yields the weaker per-ordering bound $\widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) \le \max_i |\widetilde{B}_i^{\pi}|$ stated above. Since the vertex separation number is a maximum boundary *cardinality* (an unnormalized quantity, in the sense of Chapter 2), the natural point of comparison is the unnormalized width $\widehat{\omega}_{\mathrm{rec}}$, not the normalized $\omega_{\mathrm{rec}} = \max\{0, \widehat{\omega}_{\mathrm{rec}} - 1\}$ used in the boundary conditions of Chapter 5. Equality is not claimed in general: it is expected to fail whenever the declared task admits a task-sufficient CPR interface strictly smaller than the raw structural boundary, since the cardinality-minimality defining $B_i^{\pi,\tau}$ can only tighten the bound obtained from $\widetilde{B}_i^{\pi}$, never loosen it. This is a separate effect from subsequent semantic quotienting (Chapter 3), which acts afterward on the already-selected interface's reachable states, $U_i^{\mathrm{reach}} \to S_i$, and does not retroactively change the cardinality of an interface already fixed at the $B_i^{\pi,\tau}$ stage. Under the running example of Chapter 2 (the path P_4, where the tracked structural candidate boundary has maximum size 1 at every stage and is task-sufficient, so $\widehat{\omega}_{\mathrm{rec}} = 1$ and $\omega_{\mathrm{rec}} = 0$), the proposed comparison specializes to

$$\widehat{\omega}_{\mathrm{rec}}(P_4) \le \mathrm{pw}(P_4) = 1,$$

and equality holds in this instance; this single instance is, of course, only a sanity check and not evidence beyond Evidence Level E1. Treewidth, by contrast, is naturally associated with *tree-shaped* elimination structures rather than linear ones; since every admissible ordering considered in this manuscript is linear, a direct CPR-Width bound in terms of $\mathrm{tw}(G_P)$ rather

than $\mathrm{pw}(G_P)$ may require either a different comparison argument or a generalization of admissible reconstruction structures from linear sequences to tree-shaped elimination or decomposition structures, which is itself a natural direction for future extensions of the framework rather than a change to be made within the present linear-ordering formulation. The pathwidth comparison above is therefore a conditional relation under the explicitly stated task-sufficiency and admissibility assumptions, not a universal CPR-Width theorem. The broader treewidth connection and comparisons beyond these conditions remain open and conjectural.

9.1.6 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

$$\pi_x \in \Pi_{\mathrm{adm}}(P_x, \tau; \mathfrak{M}),$$

together with fixed polynomials

$$p_j : \mathbb{N} \to \mathbb{N}, \qquad j \in \mathcal{J},$$

such that

$$N_j(P_x, \pi_x, \tau; \mathfrak{M}) \le p_j(|x|)$$

for every encoding x and every $j \in \mathcal{J}$. 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 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 requirement remains the uniform tractability condition above.

Most compelling presently supported candidate. Among the realizations presently studied, bounded-state sequential arithmetic systems are the strongest current candidate for controlled or constant CPR-Width. The ripple-adder family provides the concrete prototype: reconstruction proceeds through a fixed local carry state, the interface cardinality does not grow with the bit length, the successor signature is locally computable, 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, for SAT, or for general CSP, none of which presently admits a comparable family-level BC1–BC4 certification in this manuscript; it identifies bounded-state sequential arithmetic and, more generally, finite-state transducer-like reconstruction families as the most compelling presently supported direction for identifying further CPR-tractable families.

9.2 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 process.

9.2.1 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

$$\mathcal{R}_{\mathrm{core}}(P,\pi,\tau;\mathfrak{M}) = \big(\widehat{\omega}_{\mathrm{rec}}, \omega_{\mathrm{rec}}, S_{\mathrm{CPR}}, V_{\mathrm{sem}}, E_{\mathrm{sem}}, m_{\mathrm{stage}},$$
$$N_{\mathrm{Build}}, N_{\mathrm{Reduce}}, N_{\mathrm{Transition}}, N_{\mathrm{Reconstruct}}, N_{\mathrm{Verify}}, N_{\mathrm{Output}}\big),$$

where $V_{\mathrm{sem}}(P,\pi,\tau;\mathfrak{M}) = \sum_{i=0}^{m_{\mathrm{stage}}(P,\pi)} |S_i(P,\pi,\tau;\mathfrak{M})|$ is the total stage-indexed semantic-state count and $E_{\mathrm{sem}}(P,\pi,\tau;\mathfrak{M}) = \sum_{i=0}^{m_{\mathrm{stage}}(P,\pi)-1} |\mathcal{E}_i^{\mathrm{sem}}(P,\pi,\tau;\mathfrak{M})|$ the total semantic-transition count, $\mathcal{E}_i^{\mathrm{sem}}$ denoting the reachable labeled semantic quotient transitions from stage i to stage $i+1$ (interpreted as 0 when $m_{\mathrm{stage}}(P,\pi) = 0$); the superscript "unif" of Chapter 7 denotes their family-level specialization for the uniform ripple-adder transducer. A complete measured implementation record should additionally include

$$\mathcal{R}_{\mathrm{measured}} = (T_{\mathrm{CPR}}, M_{\mathrm{peak}}, L_{\mathrm{Output}}, I_{\mathrm{CPR}}, H, S_{\mathrm{env}}, \mathcal{P}_{\mathrm{measure}}),$$

the measured wall-clock time, peak memory use, encoded output length, implementation identifier, hardware identifier, software-environment identifier, and measurement-protocol identifier, all referring 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: parse and validate the encoded instance; construct the complete initial CPR representation; construct or read an admissible reconstruction ordering; generate the processed-prefix and unresolved-suffix sequence; generate the reachable prefix-state spaces under the declared reachability and transition rules; compute, retrieve, or certify the reconstruction-safe prefix signatures; determine the family of task-sufficient interfaces at every stage; select a cardinality-minimal task-sufficient interface whenever the exact CPR-Width is required; construct the raw interface-state spaces; project reachable prefix states onto the selected interfaces; construct the induced interface-signature maps; form the semantic quotients induced by the reconstruction-safe signatures (not generally a future-extension-set quotient); construct and process the semantic quotient transitions; perform task-relevant reconstruction; verify soundness and completeness for the declared task; materialize the required output; and record all structural quantities, operation counts, memory use, and measured runtime data. For every stage i, define the stage record $\mathcal{A}_i(P,\pi,\tau;\mathfrak{M}) = (b_i^{\pi,\tau}(P), |U_i(P,\pi,\tau;\mathfrak{M})|, |U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})|, |S_i(P,\pi,\tau;\mathfrak{M})|)$, and the complete stage sequence $\mathcal{A}(P,\pi,\tau;\mathfrak{M}) = (\mathcal{A}_0,\ldots,\mathcal{A}_{m_{\mathrm{stage}}(P,\pi)})$; this record separates interface width, raw interface-state growth, reachability filtering, semantic quotienting, transition cost, reconstruction cost, verification cost, and output cost.

9.2.2 Comparison with Classical Baselines and Width Parameters

Every comparative claim requires a declared classical baseline. Let $N_{\mathrm{Classic}}(P,\tau;A_{\mathrm{Classic}})$ denote the complete elementary-operation count of a declared classical algorithm A_{Classic}, $T_{\mathrm{Classic}}(P,\tau;I_{\mathrm{Classic}})$ the measured wall-clock time of a declared classical implementation, and $T_{\mathrm{CPR}}(P,\pi,\tau;I_{\mathrm{CPR}}) > 0$ the measured wall-clock time of the declared CPR implementation. Assuming $N_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) > 0$, the count-level gain and the measured wall-clock gain are

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

$$G_{\mathrm{time}}(P,\pi,\tau;I_{\mathrm{CPR}},I_{\mathrm{Classic}}) = \frac{T_{\mathrm{Classic}}(P,\tau;I_{\mathrm{Classic}})}{T_{\mathrm{CPR}}(P,\pi,\tau;I_{\mathrm{CPR}})};$$

the two gains remain distinct, $G_{\mathrm{count}} > 1 \not\Rightarrow G_{\mathrm{time}} > 1$. A valid comparison must fix the problem representation, the task, the required output, the correctness criterion, all preprocessing, the stopping condition, the classical algorithm, the CPR model, both implementations, the hardware

and software environment, and the measurement protocol.

For graph-based or hypergraph-based benchmarks, the experimental record should include all available classical width parameters of the declared representation $G_P = \mathcal{G}(P)$, $\mathcal{W}(P, \pi, \tau; \mathfrak{M}, \mathcal{G}) = (\omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}), \mathrm{tw}(G_P), \mathrm{pw}(G_P), \mathrm{bw}(G_P), \mathrm{cw}(G_P))$, not to infer a theorem from empirical correlation but to identify candidate upper bounds, candidate lower bounds, separation examples, ordering effects, and representation dependence – the experimental counterpart of the formal open problem in Section 9.1.5.

9.2.3 Boundary-Condition Audit

The CPR audit must distinguish finite instance records from uniform family-level certification. For one finite benchmark instance P, the instance-level audit record $\mathcal{A}_{\mathrm{BC}}^{\mathrm{inst}}(P) = (\sigma_1^{\mathrm{inst}}(P), \ldots, \sigma_4^{\mathrm{inst}}(P))$ has each component taking a value in $\{\mathrm{verified}, \mathrm{failed}, \mathrm{undetermined}\}$, recording respectively whether the required structural objects are explicitly available for the instance (BC1), whether the finite construction process and its measured operation count are completely recorded (BC2), whether soundness and completeness are verified for the declared finite task and benchmark instance (BC3), and whether the complete stage-indexed semantic-state and transition record is available (BC4). An instance-level record does not prove polynomial family behavior.

For a uniformly encoded family $\mathcal{F}$, the family-level certification record is

$$\mathcal{A}_{\mathrm{BC}}^{\mathrm{fam}}(\mathcal{F}) = (\sigma_1^{\mathrm{fam}}(\mathcal{F}), \ldots, \sigma_4^{\mathrm{fam}}(\mathcal{F})),$$

with each component taking a value in $\{\mathrm{proved}, \mathrm{disproved}, \mathrm{open}\}$; only the family-level record may support CPR-tractability certification. In particular, BC2 and BC4 are asymptotic uniform conditions and cannot be proved by one finite benchmark instance; BC4 controls the total number of stage-indexed semantic-state occurrences and the total number of reachable semantic transition edges, while the elementary operation cost of processing those states and edges is recorded separately in $N_{\mathrm{Transition}}$. If any family-level boundary condition remains open or is disproved, the family is not certified as CPR-tractable.

9.2.4 Benchmark Families and Ordering Ablation

A future benchmark corpus should contain favorable, neutral, and unfavorable instances, not only examples with visibly small candidate interfaces, so as to determine where CPR-WIDTH produces small active interfaces, where semantic quotienting is effective, where quotient construction dominates the runtime, where reconstruction or output dominates the runtime, and where CPR-WIDTH offers no advantage. Candidate benchmark groups include Boolean satisfiability instances with controlled clause-interaction structure; finite CSP instances with varying domain size, constraint arity, and primal-graph structure; graph-coloring instances with varying density and separator structure; arithmetic systems including full adders, ripple adders, and larger finite circuit blocks; tensor-structured systems with explicitly defined reconstruction components and active-index interfaces; routing and scheduling systems with bounded and unbounded interaction boundaries; coding and decoding systems with controlled trellis structure; and negative-control instances designed to produce large active interfaces or expensive semantic signatures. For each benchmark family, the instance-generation procedure, input 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 $\Pi_{\mathrm{test}}(P, \tau; \mathfrak{M}) = \{\pi_1, \ldots, \pi_r\} \subseteq \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M})$ while keeping every other model component fixed, recording for each π_j the tuple

$$\big(\widehat{\omega}_{\mathrm{rec}}(P, \pi_j, \tau; \mathfrak{M}),\ \omega_{\mathrm{rec}}(P, \pi_j, \tau; \mathfrak{M}),\ S_{\mathrm{CPR}}(P, \pi_j, \tau; \mathfrak{M}),$$
$$N_{\mathrm{CPR}}(P, \pi_j, \tau; \mathfrak{M}),\ T_{\mathrm{CPR}}(P, \pi_j, \tau; I_{\mathrm{CPR}})\big).$$

Only the ordering may vary; the problem 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: $\omega_{\mathrm{rec}}(P, \pi_a, \tau; \mathfrak{M}) < \omega_{\mathrm{rec}}(P, \pi_b, \tau; \mathfrak{M}) \not\Rightarrow T_{\mathrm{CPR}}(P, \pi_a, \tau; I_{\mathrm{CPR}}) < T_{\mathrm{CPR}}(P, \pi_b, \tau; I_{\mathrm{CPR}})$.

9.2.5 Evidence Classification and Future Research Sequence

CPR-WIDTH results should be classified by the specific evidence categories supporting them; the categories are not a single linear hierarchy, and a result may belong to more than one. The classification is: E0, Definition Evidence, a symbol, object, relation, operator, or structural quantity introduced formally; E1, Finite Verification Evidence, a claim checked exhaustively or directly on one explicitly stated finite example; E2, Structural Pattern Evidence, a recurring structural pattern identified across several models or examples; E3, Operation-Count Evidence, a finite or symbolic elementary-operation comparison relative to a declared baseline and accounting convention; E4, Asymptotic Theorem Evidence, a theorem proved uniformly for an encoded family under explicit assumptions and boundary conditions; and E5, Measured Computational Evidence, a runtime or memory claim supported by an implementation, hardware description, software environment, benchmark protocol, and reproducible measurements. The categories satisfy the non-implications E1 $\not\Rightarrow$ E4, E3 $\not\Rightarrow$ E5, E4 $\not\Rightarrow$ E5, and E5 $\not\Rightarrow$ E4: a finite verification does not establish an asymptotic theorem, an operation-count advantage does not establish measured acceleration, an asymptotic theorem does not establish a practical wall-clock advantage, and measured performance on a benchmark corpus does not establish a uniform asymptotic theorem. Every result should therefore state its complete evidence profile; a result may simultaneously possess E1, E3, and E5 support while having no E4 theorem.

On this basis, the concrete realizations and verification models developed in Chapters 6–8 fall into three groups.

First, mathematically established results include the isolated full adder, verified exhaustively on its complete finite input space and therefore carrying Evidence Level E1 (Chapter 8), and the ripple-adder family, established by a uniform proof for all n and therefore carrying Evidence Level E4 (Chapter 7), with additional measured E5 runtime and memory evidence from Chapter 8.

Second, finite graph-coloring realizations provide instance-level evidence. The C_4 realization gives a complete finite raw/admissible/reconstruction calculation, while the C_5 realization of Section 6.3.2 provides an explicit stage-by-stage record of interface growth and ordering dependence. Both are finite E1 records; neither constitutes a family-level E4 theorem.

Third, Boolean satisfiability, constraint satisfaction, and tensor representations remain theoretically formulated realizations: structural candidate interfaces and reconstruction templates are given, but no general family-level CPR-tractability theorem is established for these classes in this manuscript.

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 and does not constitute E4 evidence for random 3-SAT as a family. It is a negative structural observation on the tested candidate interfaces, not a CPR-Width measurement.

A disciplined CPR-WIDTH research program may proceed through: complete exact mathematical characterization of CPR-Width for selected finite models; implementation of exact reachable-prefix, signature, interface, and semantic-quotient construction; recording BC1–BC4 evidence at the instance level; proving BC2 and BC4 at the uniform family level whenever an asymptotic tractability claim is intended; publishing complete component-wise operation-count records; comparing multiple admissible reconstruction orderings while keeping all other model and implementation conditions fixed; comparing CPR-WIDTH with declared classical baselines; extending finite analysis to uniformly encoded families; deriving formal relations among interface width, semantic-state growth, transition structure, and complete operation count; investigating exact and approximate relations to classical width parameters; determining controlled or constant CPR-Width bounds for additional problem-class subclasses; 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.

9.3 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 $\boldsymbol{\omega_{\mathbf{rec}}(P, \pi, \tau; \mathfrak{M}) \to S_{\mathbf{CPR}}(P, \pi, \tau; \mathfrak{M}) \to N_{\mathbf{CPR}}(P, \pi, \tau; \mathfrak{M})}$: the first arrow expresses the influence of the active reconstruction interface on the raw and semantic state spaces, the second the contribution of semantic-state processing to the complete CPR operation count, and 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; determination of 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 $\mathrm{BC}_1 \wedge \mathrm{BC}_2 \wedge \mathrm{BC}_3 \wedge \mathrm{BC}_4$, but the full uniform quantified requirement stated at the start of this chapter; a polynomial-time conclusion is permitted only after one fixed decision task, one fixed uniform reconstruction model, one deterministic preprocessing algorithm (with its ordering-supply component, Chapter 5), and six fixed polynomial component bounds have been established for every encoded instance, together with BC1–BC4 (which already require sound and complete reconstruction uniformly).

No NP-complete language is proved in this work 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, transition cost, reconstruction cost, verification cost, output cost, operation-count gain, and measured runtime gain. Every conclusion must be limited to the specific evidence category or evidence profile supporting it: finite verification supports finite conclusions, structural patterns support structural conjectures and model comparisons, operation-count evidence supports count-level conclusions, asymptotic tractability requires a uniformly encoded family with all four boundary conditions and polynomial control of all six cost components, and measured acceleration requires a separate reproducible implementation and benchmark protocol.

CPR-Width is therefore useful not only when it is small. A small value identifies a reconstruction model with limited simultaneously active information; variation across orderings, as quantified by the Width Ratio above, measures structural sensitivity to the choice of ordering; and, as the random 3-SAT record of Chapter 8 illustrates, a persistently large structural candidate interface under tested representations and orderings provides negative diagnostic evidence that the selected reconstruction model does not expose a narrow active boundary, without thereby establishing the exact minimum CPR-Width of those instances. In this sense, CPR-Width functions simultaneously as a candidate tractability parameter, a comparative structural parameter across orderings and representations, and a diagnostic structural parameter for a selected representation.

Thus, CPR-WIDTH remains a formal framework for controlled partial reconstruction, semantic-state compression, and width-based runtime analysis. Its broader complexity-theoretic significance depends on the future identification of nontrivial uniformly encoded families satisfying

the complete structural, semantic, algorithmic, and runtime requirements.

A Core Definitions and Dependency Map

This appendix provides a compact structural reference for the principal objects of the CPR-WIDTH framework.

Unlike a conventional symbol register, the purpose is not merely to list notation. The appendix records the dependency order in which the central objects of the framework are defined and clarifies the distinction between interface cardinality, semantic-state complexity, and computational cost.

No new assumptions or theoretical claims are introduced here. All definitions refer to the corresponding constructions in the main text.

A.1 Declared Reconstruction Setting

A CPR analysis is relative to four basic objects:

$$(P, \tau, \mathfrak{M}, \pi),$$

where

- P is a finite encoded problem instance,
- τ is the declared computational task,
- $\mathfrak{M}$ is the declared CPR reconstruction model,
- π is an admissible reconstruction ordering.

The ordering must satisfy

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

Consequently, CPR-Width is not an invariant of the underlying combinatorial object alone. It is relative to the representation, task, reconstruction model, and admissible ordering.

When τ and $\mathfrak{M}$ are fixed unambiguously by context, they may be suppressed from the notation.

A.2 Definition Dependency Chain

The central definitions occur in the following non-circular order:

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

Here,

$$\mathcal{R}_i^{\pi}(P)$$

denotes the reachable prefix states at reconstruction stage i, and

$$E_i^{\pi}(r)$$

denotes the admissible future-extension behavior of a reachable state r.

The reconstruction-safe signature

$$\mathrm{Sig}_{\tau,i}^{\pi}(r)$$

retains exactly the information required by the declared task and reconstruction model.

A task-sufficient interface

$$B_i^{\pi,\tau}$$

is then selected so that its projected information is sufficient to determine the required task-relevant future behavior.

The interface used in the width definition is cardinality-minimal among the task-sufficient interfaces admitted by the declared model.

A.3 Operational and Normalized Width

Let

$$b_i^{\pi,\tau}(P) = |B_i^{\pi,\tau}|$$

denote the cardinality of a minimum task-sufficient interface at stage i.

The unnormalized reconstruction width is

$$\widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = \max_i b_i^{\pi,\tau}(P).$$

This quantity has a direct operational interpretation:

$$\boxed{\widehat{\omega}_{\mathrm{rec}} = \text{maximum number of simultaneously retained task-relevant interface components.}}$$

The normalized CPR-Width is

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

Hence

$$\widehat{\omega}_{\mathrm{rec}} \leq \omega_{\mathrm{rec}} + 1.$$

The normalization must not be interpreted as removing an operational component. In particular,

$$\omega_{\mathrm{rec}} = 0$$

may correspond to

$$\widehat{\omega}_{\mathrm{rec}} = 1.$$

For the declared task and model, the ordering-independent quantities are obtained by minimizing over admissible orderings:

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

and

$$\omega_{\mathrm{rec}}(P, \tau; \mathfrak{M}) = \min_{\pi \in \Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})} \omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}).$$

A.4 Three Distinct State Levels

The CPR-WIDTH framework distinguishes interface cardinality from state-space cardinality.

For a fixed stage i, let

$$\mathcal{U}_i$$

denote the raw interface-state space and let

$$\mathcal{U}_i^{\mathrm{reach}} \subseteq \mathcal{U}_i$$

denote the subset reachable under the declared reconstruction model.
Semantic equivalence is defined through the reconstruction-safe signature:

$$u \equiv_{\tau,\mathfrak{M}} v \quad \Longleftrightarrow \quad \widehat{\mathrm{Sig}}_{\tau,i}(u) = \widehat{\mathrm{Sig}}_{\tau,i}(v).$$

The semantic quotient is

$$\mathcal{S}_i = \mathcal{U}_i^{\mathrm{reach}} / \equiv_{\tau,\mathfrak{M}}.$$

Therefore,

$$|\mathcal{S}_i| \le |\mathcal{U}_i^{\mathrm{reach}}| \le |\mathcal{U}_i|.$$

These quantities must not be identified with the interface size

$$|B_i^{\pi,\tau}|.$$

Interface minimization determines how many components must be retained. Semantic quotienting determines how many task-distinct states remain after an interface has been selected.

A.5 Width–State–Runtime Dependency

Suppose every active component has at most $q(P)$ possible local values.
Then

$$|\mathcal{U}_i| \le q(P)^{|B_i^{\pi,\tau}|}.$$

Consequently,

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

and therefore

$$\boxed{S_{\mathrm{CPR}} \le q(P)^{\widehat{\omega}_{\mathrm{rec}}} \le q(P)^{\omega_{\mathrm{rec}}+1}.}$$

This gives the structural dependency

$$\boxed{\widehat{\omega}_{\mathrm{rec}} \longrightarrow \omega_{\mathrm{rec}} \longrightarrow S_{\mathrm{CPR}} \longrightarrow N_{\mathrm{CPR}}.}$$

The arrows denote dependency, not equality.
In particular,

$$S_{\mathrm{CPR}} \in \mathrm{poly}(|P|)$$

does not by itself imply

$$N_{\mathrm{CPR}} \in \mathrm{poly}(|P|).$$

The complete runtime model must also control all declared operation categories.

A.6 Complete Runtime Decomposition

The total CPR operation count is decomposed as

$$\boxed{N_{\mathrm{CPR}} = N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} + N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}.}$$

The terms respectively account for

1. construction of the declared representation,
2. reduction and preprocessing,
3. semantic-state transitions,
4. reconstruction operations,
5. verification of reconstructed objects,
6. production of the required output.

A small CPR-Width or polynomial semantic-state count therefore does not alone establish polynomial-time tractability.

A.7 Task Relativity

CPR-Width is explicitly task-relative.

For the same encoded problem instance P, different tasks may require different task-sufficient interfaces and therefore different widths.

For example, it may occur that

$$B_i^{\pi,\mathrm{dec}} \neq B_i^{\pi,\mathrm{count}},$$

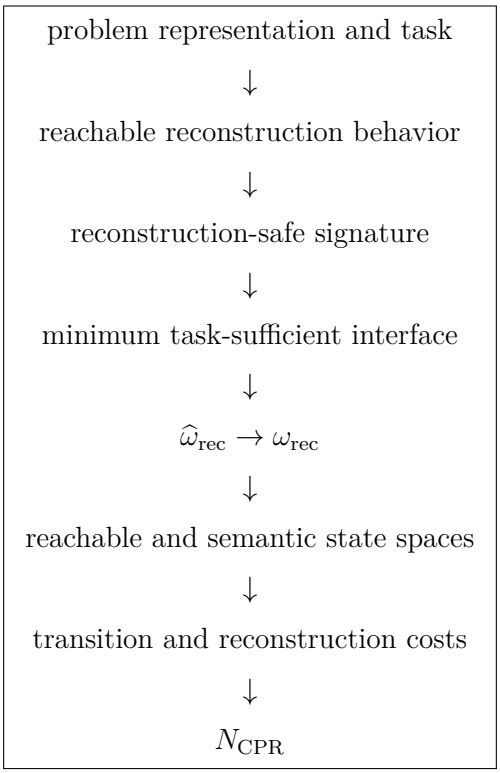
and consequently

$$\omega_{\mathrm{rec}}(P, \pi, \mathrm{dec}; \mathfrak{M}) \neq \omega_{\mathrm{rec}}(P, \pi, \mathrm{count}; \mathfrak{M}).$$

Accordingly, a width result established for an evaluation task, decision task, counting task, or enumeration task must not be transferred to another task without a separate task-sufficiency argument.

A.8 Interpretive Summary

The logical architecture of CPR-WIDTH can be summarized as



The framework therefore separates three questions that should not be conflated:

1. How many components must remain simultaneously active?
2. How many task-distinct semantic states remain over those components?
3. What computational cost is required to construct, update, reconstruct, verify, and output the corresponding objects?

This separation is the central structural distinction underlying the CPR-WIDTH framework.

B Core Definition Register

This appendix summarizes the principal definitions of the CPR-WIDTH framework in compact form.

The register is intended as a reference only. The complete definitions, assumptions, scope conditions, and proofs are given in the corresponding chapters of the manuscript.

Unless explicitly suppressed, all reconstruction quantities are relative to the finite problem instance P, the declared computational task τ, the reconstruction model $\mathfrak{M}$, and an admissible reconstruction ordering π.

Table B.1: Core definitions of CPR-WIDTH.

Definition / Object	**Formula**	**Meaning**
Finite component set	$V(P) = \{v_1, \ldots, v_n\}$	Finite set of reconstruction components of the problem instance P.

Continued on the next page

Definition / Object	Formula	Meaning
Declared CPR reconstruction model	$\mathfrak{M}(P,\tau) = \big(\mathrm{Adm}_{P,\tau}, \mathfrak{R}, \Lambda, \delta, \mathrm{Sig}_\tau\big)$	Declared reconstruction model fixing admissible orderings, reachable-state rules, stage transitions, and the reconstruction-safe signature family.
Admissible reconstruction ordering	$\pi = (v_{\pi(1)}, \dots, v_{\pi(n)}) \in \Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})$ with $\mathrm{Adm}_{P,\tau}(\pi) = 1$	A permutation of the reconstruction components satisfying the admissibility predicate of the declared CPR reconstruction model.
Admissible-ordering set	$\Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M}) = \{\pi \in \mathrm{Sym}(V(P)) : \mathrm{Adm}_{P,\tau}(\pi) = 1\}$	Set of all reconstruction orderings admissible under the declared task and reconstruction model.
Processed prefix	$V_i^\pi = \{v_{\pi(1)}, \dots, v_{\pi(i)}\}$	Set of components processed after reconstruction stage i.
Unresolved suffix	$\overline{V}_i^\pi = V(P) \setminus V_i^\pi$	Set of components not yet processed after stage i.
Reachable prefix-state space	$\mathcal{R}_i^\pi(P) \subseteq \prod_{v \in V_i^\pi} D_v$	Prefix assignments that can actually arise under the declared transition and admissibility rules.
Feasible continuation set	$E_i^\pi(r) = \left\{ s \in \prod_{v \in \overline{V}_i^\pi} D_v : r \cup s \in \mathrm{Sol}(P) \right\}$	Complete feasible suffix assignments extending the reachable prefix state r.
Task-relative future value	$\mathrm{Ans}_{\tau,i}^\pi(r) = \Theta_{\tau,i}^\pi\big(r, E_i^\pi(r)\big)$	Value required by the declared task at reconstruction stage i.
Admissible next-value set	$\Lambda_i^\pi(r) = \big\{a \in D_{w_i^\pi} : \delta_i^\pi(r,a) \in \mathcal{R}_{i+1}^\pi(P)\big\}$	Next component values that produce reachable successor prefix states.
Stage transition	$\delta_i^\pi(r,a) = r \cup \{w_i^\pi \mapsto a\}$	Successor reachable prefix state obtained by processing the next component with admissible value a.
Reconstruction-safe signature	$\mathrm{Sig}_{\tau,i}^\pi : \mathcal{R}_i^\pi(P) \longrightarrow \Sigma_{\tau,i}^\pi$	Task-relative signature satisfying the reconstruction-safety conditions S1–S3: task preservation, preservation of admissible next values, and successor congruence.
Task-sufficient interface	$B \subseteq V_i^\pi, \quad r\|_B = r'\|_B \Longrightarrow \mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r')$ for all $r, r' \in \mathcal{R}_i^\pi(P)$	A set of processed components whose retained values determine the complete reconstruction-safe signature at stage i.
Family of task-sufficient interfaces	$\mathfrak{B}_i^{\pi,\tau}(P) = \{B \subseteq V_i^\pi : B \text{ is task-sufficient}\}$	All task-sufficient reconstruction interfaces available at stage i.

Continued on the next page

Definition / Object	Formula	Meaning
Minimum interface size	$b_i^{\pi,\tau}(P) = \min\{\lvert B\rvert : B \in \mathfrak{B}_i^{\pi,\tau}(P)\}$	Minimum number of processed components whose retained values are sufficient at stage i.
Selected active reconstruction interface	$B_i^{\pi,\tau} \in \underset{B\in\mathfrak{B}_i^{\pi,\tau}(P)}{\arg\min} \lvert B\rvert$ with $\lvert B_i^{\pi,\tau}\rvert = b_i^{\pi,\tau}(P)$	One selected cardinality-minimal task-sufficient active interface at stage i. The minimizing set need not be unique.
Interface sequence	$B(P,\pi,\tau) = \left(B_0^{\pi,\tau},\dots,B_n^{\pi,\tau}\right)$	Complete selected active-interface sequence induced by the reconstruction ordering.
Unnormalized CPR interface width	$\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = \max_{0\le i\le n} b_i^{\pi,\tau}(P)$	Actual maximum cardinality of a minimum task-sufficient active interface during the reconstruction process.
Controlled Partial Reconstruction Width	$\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = \max\{0, \widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) - 1\}$	Normalized CPR-Width induced by the admissible ordering.
Minimum unnormalized CPR width	$\widehat{\omega}_{\mathrm{rec}}(P,\tau;\mathfrak{M}) = \min_{\pi\in\Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})} \widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})$	Smallest unnormalized active-interface width attainable over all admissible reconstruction orderings.
Minimum CPR-Width	$\omega_{\mathrm{rec}}(P,\tau;\mathfrak{M}) = \min_{\pi\in\Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})} \omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})$	Smallest normalized CPR-Width attainable over all admissible reconstruction orderings.
Maximum local domain size	$q(P) = \max_{v\in V(P)} \lvert D_v\rvert$	Largest local component-domain cardinality.
Raw interface-state space	$U_i(P,\pi,\tau;\mathfrak{M}) = \prod_{v\in B_i^{\pi,\tau}} D_v$	All syntactically possible assignments to the selected active interface.
Reachable interface-state space	$U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}) = \rho_i^{\pi,\tau}\left(\mathcal{R}_i^{\pi}(P)\right)$	Interface assignments that arise from reachable prefix states.
Induced interface signature	$\widehat{\mathrm{Sig}}_{\tau,i}^{\pi} : U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}) \longrightarrow \Sigma_{\tau,i}^{\pi}$ with $\mathrm{Sig}_{\tau,i}^{\pi} = \widehat{\mathrm{Sig}}_{\tau,i}^{\pi} \circ \rho_i^{\pi,\tau}$	Representative-independent reconstruction-safe signature induced on reachable interface states.
Reconstruction-safe semantic equivalence	$u \equiv_{P,\pi,i}^{\tau,\mathfrak{M}} v \iff \widehat{\mathrm{Sig}}_{\tau,i}^{\pi}(u) = \widehat{\mathrm{Sig}}_{\tau,i}^{\pi}(v)$	Two reachable interface states are semantically equivalent exactly when they induce the same reconstruction-safe task-relative signature. Equality of literal continuation sets is not required in general.

Continued on the next page

Definition / Object	Formula	Meaning
Semantic quotient	$S_i(P,\pi,\tau;\mathfrak{M}) = U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}) \big/ \equiv_{P,\pi,i}^{\tau,\mathfrak{M}}$	Stage-wise semantic state space obtained by merging reachable interface states with identical reconstruction-safe signatures.
Maximum semantic-state count	$S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) = \max_{0\le i\le n} \lvert S_i(P,\pi,\tau;\mathfrak{M})\rvert$	Maximum number of semantic quotient states occurring at any reconstruction stage.
Binary semantic-state dimension	$D_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) = \lceil \log_2 \max\{1, S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M})\}\rceil$	Minimum fixed-length binary index size sufficient to distinguish the maximum number of stage-wise semantic classes.
Width–state bound	$S_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \le q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} \le q(P)^{\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}$	Structural upper bound linking active-interface width to the maximum semantic-state count.
Complete CPR operation count	$N_{\mathrm{CPR}} = N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} + N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}$	Complete elementary-operation count. The six terms are disjoint accounting categories.
Build cost	N_{Build}	Operations required to construct the initial explicit CPR representation and build data.
Reduction cost	N_{Reduce}	Operations required for task-sufficient structural reduction, interface construction, signature preparation, and semantic quotient construction assigned to reduction.
Semantic transition cost	$N_{\mathrm{Transition}}$	Operations required to process all reachable stage-to-stage semantic transitions.
Reconstruction cost	$N_{\mathrm{Reconstruct}}$	Operations required to recover the task-relevant reconstructed result.
Verification cost	N_{Verify}	Operations required to verify correctness of the reconstructed result for the declared task.
Output cost	N_{Output}	Operations required to materialize the required output.
Post-transition cost	$R_{\mathrm{post}} = N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}$	Compact notation for reconstruction, verification, and output costs.
Count-level gain	$G_{\mathrm{count}} = \dfrac{N_{\mathrm{Classic}}}{N_{\mathrm{CPR}}}$	Elementary-operation-count comparison relative to a declared classical baseline solving the same task and output requirement.

Continued on the next page

Definition / Object	Formula	Meaning
Measured wall-clock gain	$G_{\text{time}} = \dfrac{T_{\text{Classic}}}{T_{\text{CPR}}}$	Measured execution-time comparison relative to a declared classical implementation under a stated experimental protocol.
Uniform encoded family	$\mathcal{F} = \{P_x : x \in \{0,1\}^*\}$	Uniformly encoded family of finite problem instances.
Associated decision language	$L_{\mathcal{F}} = \{x \in \{0,1\}^* : P_x \text{ is a yes-instance for } \tau\}$	Decision language associated with the uniformly encoded family.
Uniform preprocessing algorithm	$\mathcal{D}_x = A_{\text{Preprocess}}(x),$ $\pi_x = A_\pi(x) \in \Pi_{\text{adm}}(P_x, \tau; \mathfrak{M})$	One deterministic preprocessing algorithm used uniformly for the family. A_π denotes its ordering-supply component and is not a second independently quantified preprocessing algorithm.
Boundary conditions	BC1, BC2, BC3, BC4	Respectively: explicit uniform structural representation; polynomial uniform constructibility; task-relative soundness and completeness; and polynomial semantic-state and transition-structure control.
Runtime-component index set	$\mathcal{J} = \{\text{Build}, \text{Reduce}, \text{Transition},$ $\text{Reconstruct}, \text{Verify}, \text{Output}\}$	Index set for the six complete CPR runtime components.
Uniform CPR tractability requirement	$\exists\tau\, \exists\mathfrak{M}\, \exists A_{\text{Preprocess}}\, \exists\{p_j\}_{j\in\mathcal{J}}\, \forall x \in \{0,1\}^* :$ $\pi_x = A_\pi(x) \in \Pi_{\text{adm}}(P_x, \tau; \mathfrak{M}),$ $\bigwedge_{k=1}^{4} \text{BC}_k(P_x, \pi_x, \tau; \mathfrak{M}),$ $\bigwedge_{j\in\mathcal{J}} [N_j(P_x, \pi_x, \tau; \mathfrak{M}) \le p_j(\lvert x\rvert)]$	Complete uniform CPR certification requirement. The task, model, preprocessing algorithm including its ordering-supply component, and all six polynomial bounds are fixed before the individual encoding x is selected.
CPR-tractable decision family	$\mathcal{F}$ satisfies the uniform requirement above	A uniformly encoded decision family satisfying BC1–BC4 and fixed polynomial bounds for all six complete runtime components.
CPR-certified language class	$\mathbf{P}_{\text{CPR}} = \big\{L \subseteq \{0,1\}^* : L = L_{\mathcal{F}}$ for some uniformly encoded CPR-tractable decision family $\mathcal{F}\big\}$	Class of decision languages certified by the complete uniform CPR-tractability conditions.
Certified-language consequence	$\mathbf{P}_{\text{CPR}} \subseteq \mathbf{P}$	Every decision language satisfying the complete CPR certification requirements is deterministically polynomial-time decidable.

Continued on the next page

Definition / Object	Formula	Meaning
Conditional NP-complete consequence	L^* NP-complete $\wedge$ $L^* \in \mathbf{P}_{\mathrm{CPR}} \Longrightarrow \mathbf{P} = \mathbf{NP}$	Strictly conditional complexity consequence. This manuscript does not establish the premise for any NP-complete language.

Interpretive note. The register preserves the principal distinction of CPR-WIDTH: structural width, semantic-state cardinality, and complete computational cost are related but are not interchangeable quantities. In particular,

$$\widehat{\omega}_{\mathrm{rec}} \longrightarrow S_{\mathrm{CPR}} \longrightarrow N_{\mathrm{CPR}}$$

is a structural and accounting dependency chain, not a universal polynomial-time implication. Likewise, a small normalized ω_{rec} alone does not establish small runtime, and semantic equivalence is determined by the declared reconstruction-safe signature rather than by equality of literal continuation sets.

C Proof Details

This appendix records compact proof details for selected statements used in the main text. The purpose is to make the principal structural and runtime implications directly checkable without repeating the full development of Chapters 2–4.

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

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

C.1 Interface Bound

By definition,

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

Since the selected interface $B_i^{\pi,\tau}$ is cardinality-minimal,

$$|B_i^{\pi,\tau}| = b_i^{\pi,\tau}(P).$$

Therefore, for every stage i,

$$|B_i^{\pi,\tau}| \le \widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}).$$

Using

$$\omega_{\mathrm{rec}} = \max\{0, \widehat{\omega}_{\mathrm{rec}} - 1\},$$

one also has

$$\widehat{\omega}_{\mathrm{rec}} \le \omega_{\mathrm{rec}} + 1.$$

Hence

$$\boxed{|B_i^{\pi,\tau}| \le \widehat{\omega}_{\mathrm{rec}} \le \omega_{\mathrm{rec}} + 1}.$$

C.2 Raw State Bound

The raw interface-state space is

$$U_i = \prod_{v \in B_i^{\pi,\tau}} D_v.$$

With

$$q(P) = \max_{v \in V(P)} |D_v|,$$

it follows that

$$|U_i| = \prod_{v \in B_i^{\pi,\tau}} |D_v| \leq q(P)^{|B_i^{\pi,\tau}|}.$$

Applying the Interface Bound gives

$$\boxed{|U_i| \leq q(P)^{\widehat{\omega}_{\mathrm{rec}}} \leq q(P)^{\omega_{\mathrm{rec}}+1}}.$$

C.3 Reachable State Bound

Because the reachable interface states form a subset of the raw interface-state space,

$$U_i^{\mathrm{reach}} \subseteq U_i.$$

Consequently,

$$\boxed{|U_i^{\mathrm{reach}}| \leq |U_i| \leq q(P)^{\widehat{\omega}_{\mathrm{rec}}} \leq q(P)^{\omega_{\mathrm{rec}}+1}}.$$

Reachability filtering can therefore reduce, but never increase, the width-based state bound.

C.4 Semantic-State Bound

The semantic state space is the quotient

$$S_i = U_i^{\mathrm{reach}} / \equiv_{P,\pi,i}^{\tau,\mathfrak{M}}.$$

Since the canonical quotient map is surjective,

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

Therefore,

$$|S_i| \leq q(P)^{\widehat{\omega}_{\mathrm{rec}}}.$$

Taking the maximum over all stages yields

$$\boxed{S_{\mathrm{CPR}} \leq q(P)^{\widehat{\omega}_{\mathrm{rec}}} \leq q(P)^{\omega_{\mathrm{rec}}+1}}.$$

This is a state-cardinality bound; it does not by itself bound the cost of constructing the semantic quotient.

C.5 Total Operation Bound

At reconstruction stage i, at most $|S_i|$ semantic states are processed. If each such state has at most λ_i admissible outgoing transitions, the number of processed transition edges is bounded by

$$|S_i|\lambda_i.$$

Let c_{edge} bound the cost of processing one admissible transition edge and let c_{local} bound the additional state-local work performed once per semantic state. Then

$$N_{\mathrm{Transition},i} \leq |S_i| \left(\lambda_i c_{\mathrm{edge}} + c_{\mathrm{local}}\right).$$

Using

$$|S_i| \leq S_{\mathrm{CPR}}, \qquad \lambda_i \leq \lambda_{\max},$$

and summing over the $m_{\mathrm{stage}}(P,\pi)$ nonterminal stages gives

$$N_{\mathrm{Transition}} \leq m_{\mathrm{stage}} S_{\mathrm{CPR}} \left(\lambda_{\max} c_{\mathrm{edge}} + c_{\mathrm{local}}\right).$$

Together with

$$S_{\mathrm{CPR}} \le q(P)^{\widehat{\omega}_{\mathrm{rec}}},$$

this yields

$$\boxed{N_{\mathrm{Transition}} \le m_{\mathrm{stage}}\, q(P)^{\widehat{\omega}_{\mathrm{rec}}}\, (\lambda_{\max} c_{\mathrm{edge}} + c_{\mathrm{local}})}.$$

Equivalently, under a declared per-state accounting convention,

$$N_{\mathrm{Transition}} \le m_{\mathrm{stage}} S_{\mathrm{CPR}} c_{\mathrm{state}}.$$

The per-edge and per-state conventions are alternatives and are not added together.

This bound is a *sufficient* condition for a polynomial $N_{\mathrm{Transition}}$: it applies when m_{stage}, $q(P)^{\widehat{\omega}_{\mathrm{rec}}}$, $\lambda_{\max}$, c_{edge}, and c_{local} are each polynomially bounded in the input encoding length. It is *not* asserted to follow from small CPR-Width alone.

The complete CPR operation count is

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

Hence, if fixed polynomials p_j satisfy

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

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 hypothesis is *not* implied by BC2 or BC4 alone: BC2 bounds only the preprocessing work assigned to N_{Build} and the static portion of N_{Reduce} (Definition 5.2), and BC4 bounds only the semantic-state and transition-structure cardinalities entering $N_{\mathrm{Transition}}$. Neither provides a polynomial bound on $N_{\mathrm{Reconstruct}}$, N_{Verify}, or N_{Output}, nor on the dynamic (non-static) portion of N_{Reduce}, by itself. The remaining components require the same additional polynomial-bound assumptions used in the Uniform CPR Tractability theorem (Theorem 5.1).

Thus the width-dependent state bound contributes to the transition bound, but polynomial total runtime requires polynomial control of all six runtime components; small CPR-Width alone does not imply it.

C.6 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' of u, 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 then 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,

$$\mathrm{Sig}^{\pi}_{\tau,i+1}\big(\delta^{\pi}_{i}(r,a)\big) = \mathrm{Sig}^{\pi}_{\tau,i+1}\big(\delta^{\pi}_{i}(r',a)\big).$$

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.

Table D.1: Compact CPR-WIDTH audit checklist.

Audit item	**Required check**
Problem model	Declare P, τ, $\mathfrak{M}$, component domains, and the required output.
Ordering	Verify $\pi \in \Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})$.
Reachability	Declare the reachable prefix spaces $\mathcal{R}^{\pi}_{i}(P)$, admissible next values Λ^{π}_{i}, and transitions δ^{π}_{i}.
Signature safety	Verify S1–S3 for $\mathrm{Sig}^{\pi}_{\tau,i}$.
Task-sufficient interface	Verify $r\|_B = r'\|_B \Rightarrow \mathrm{Sig}^{\pi}_{\tau,i}(r) = \mathrm{Sig}^{\pi}_{\tau,i}(r')$.
Minimal interface	Distinguish structural candidate interfaces from the cardinality-minimal task-sufficient interface $B^{\pi,\tau}_{i}$.
Width	Record $\widehat{\omega}_{\mathrm{rec}} = \max_i b^{\pi,\tau}_{i}(P)$, $\omega_{\mathrm{rec}} = \max\{0, \widehat{\omega}_{\mathrm{rec}} - 1\}$.
Semantic quotient	Verify $u \equiv^{\tau,\mathfrak{M}}_{P,\pi,i} v \iff \widehat{\mathrm{Sig}}^{\pi}_{\tau,i}(u) = \widehat{\mathrm{Sig}}^{\pi}_{\tau,i}(v)$.
State bound	Verify $S_{\mathrm{CPR}} \le q(P)^{\widehat{\omega}_{\mathrm{rec}}} \le q(P)^{\omega_{\mathrm{rec}}+1}$.
Runtime accounting	Account disjointly for $N_{\mathrm{Build}}, N_{\mathrm{Reduce}}, N_{\mathrm{Transition}}, N_{\mathrm{Reconstruct}}, N_{\mathrm{Verify}}, N_{\mathrm{Output}}$.
Transition cost	Control semantic-state count, branching, and per-state or per-edge processing cost.
BC1–BC4	Verify explicit representation, polynomial constructibility, task-relative correctness, and polynomial semantic-state/transition control.

Audit item	Required check
Uniformity	For family claims, use one fixed τ, $\mathfrak{M}$, $A_{\mathrm{Preprocess}}$ with ordering component A_π, and fixed polynomial bounds for all six runtime components.
Evidence status	Distinguish finite verification, sampled empirical evidence, family-level theorems, and measured runtime evidence.
Scope	Do not infer polynomial runtime from small CPR-Width or a small semantic quotient alone.

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.

E Figure Register

This appendix records the scientific figures used in the manuscript. All figures are generated natively in LaTeX (TikZ) directly inside the chapter files in which they occur; none depend on an externally maintained image file, so no figure can silently fall out of sync with the mathematics it illustrates as the manuscript is revised.

Table E.1: Figure Register for CPR-WIDTH.

Figure	Page	Description
Figure 2.1	9	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	10	Chapter 2. Semantic-runtime architecture of CPR-WIDTH: $\omega_{\mathrm{rec}} \longrightarrow S_{\mathrm{CPR}} \longrightarrow N_{\mathrm{CPR}}$.
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	22	Chapter 4. Complete runtime framework of CPR-WIDTH: the six disjoint operation-count components summing to 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] Marek Cygan et al. *Parameterized Algorithms*. Springer, 2015.

[3] Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Springer, 2013.

[4] Nancy G. Kinnersley. "The Vertex Separation Number of a Graph Equals Its Path-Width". In: *Information Processing Letters* 42.6 (1992), pp. 345–350.

[5] Neil Robertson and Paul D. Seymour. "Graph Minors. II. Algorithmic Aspects of Tree-Width". In: *Journal of Algorithms* 7.3 (1986), pp. 309–322.