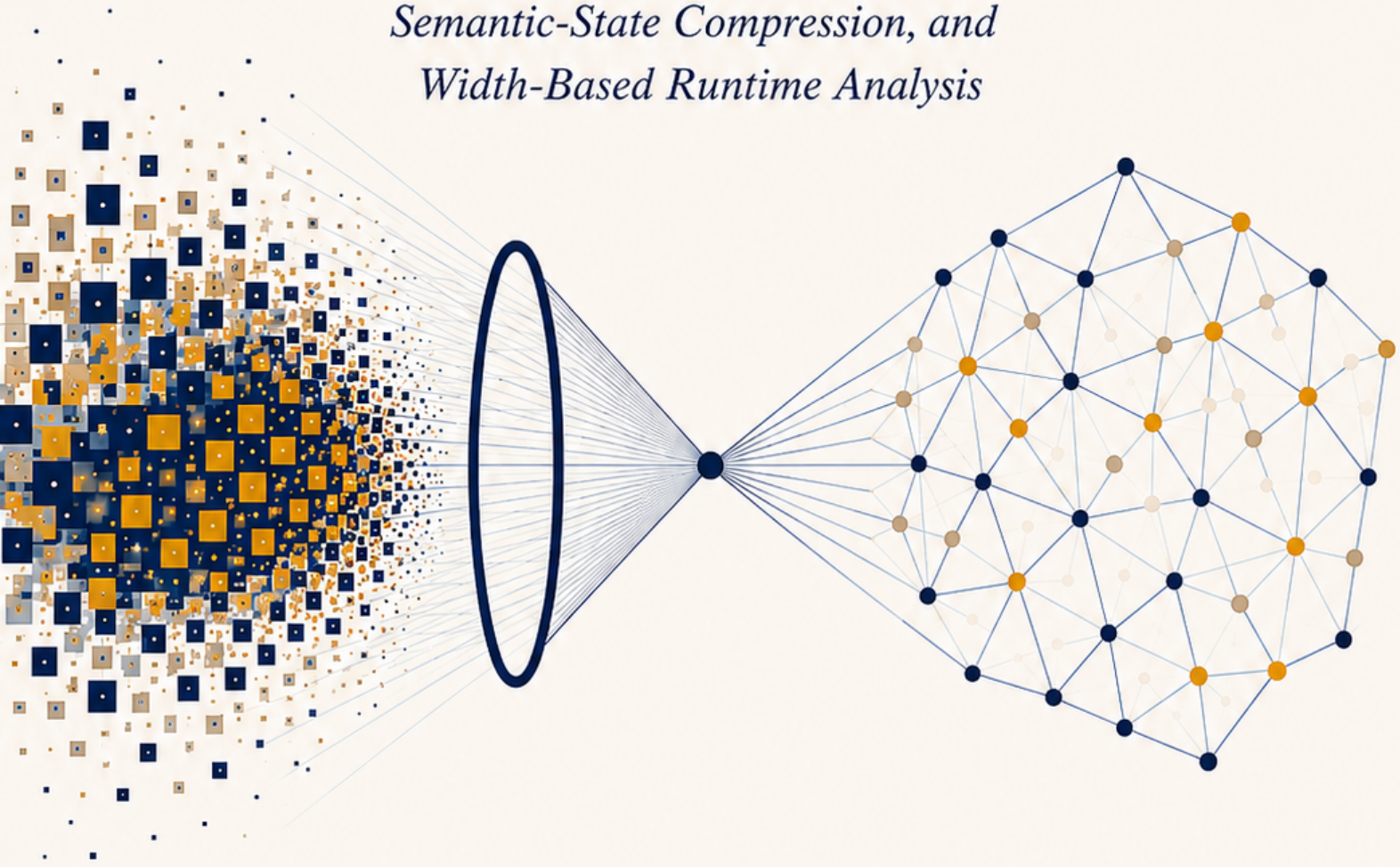
CPR-WIDTH

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-V1

Abstract

Controlled Partial Reconstruction Width (*CPR-Width*) is introduced as 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. CPR-Width is defined from the maximum cardinality of this interface over the complete reconstruction process and is minimized over admissible orderings.

The framework distinguishes raw interface states, reachable states, and task-relative semantic states obtained through future-extension equivalence. This yields the structural chain

$$\boldsymbol{\omega_{\mathrm{rec}}(P,\pi) \longrightarrow S_{\mathrm{CPR}}(P,\pi) \longrightarrow N_{\mathrm{CPR}}(P,\pi)} \tag{0.1}$$

connecting reconstruction width, semantic-state growth, and complete computational cost.

A full runtime model is developed that 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. Exact finite verification is provided for the full adder and ripple-adder models.

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; future-extension 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 information that must remain active and information that may safely be discarded is therefore fundamental.

The present manuscript studies the minimum number of processed components that must remain simultaneously active during a sound, complete, and task-relative reconstruction.

Rather than measuring the size of the complete search space, the present theory measures the minimum active reconstruction interface required during reconstruction.

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 [14, 2]. Parameterized complexity extends this perspective by separating the influence of the input size from the influence of additional structural parameters [8, 4]. 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.

Its value depends on

- the declared reconstruction model,
- the chosen reconstruction ordering,
- the active reconstruction interface,
- the declared computational task,
- and the reconstruction-safe future-signature family.

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

$$\boldsymbol{\omega_{\mathrm{rec}}(P,\pi) \longrightarrow S_{\mathbf{CPR}}(P,\pi) \longrightarrow N_{\mathbf{CPR}}(P,\pi)} \tag{1.1}$$

The first quantity measures the maximal active reconstruction interface.

The second describes the corresponding semantic-state space.

The third represents the computational effort required by the declared reconstruction model.

The formal construction of these quantities is developed progressively in Chapters 2–3.

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, admissible reconstruction orderings, reconstruction-safe future signatures, task-sufficient interfaces, and the formal definition of Controlled Partial Reconstruction Width.

Chapter 3 establishes the complete runtime model.

Chapter 4 derives the boundary conditions required for tractability.

Chapter 5 illustrates the theory by representative problem classes.

Chapter 6 develops the adder and arithmetic reconstruction realization, its finite verification records, and the resulting complexity-theoretic scope, open problems, and future research directions.

2 Controlled Partial Reconstruction Width

This chapter introduces the central structural parameter of the CPR-WIDTH framework.

The purpose is to determine 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:

$$\mathcal{R}_i^{\pi} \longrightarrow E_i^{\pi}(r) \longrightarrow \mathrm{Sig}_{\tau,i}^{\pi}(r) \longrightarrow B_i^{\pi,\tau} \longrightarrow \omega_{\mathrm{rec}}(P,\pi,\tau). \tag{2.1}$$

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

2.1 Declared CPR Reconstruction Model

Let P be a finite problem instance equipped with a finite reconstruction-component representation

$$V(P) = \{v_1, \ldots, v_n\}. \tag{2.2}$$

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

$$\varnothing \neq D_v, \qquad |D_v| < \infty. \tag{2.3}$$

Let

$$q(P) = \max_{v \in V(P)} |D_v| \tag{2.4}$$

denote the maximum local domain size of P.

The complete assignment space is

$$\mathcal{X}(P) = \prod_{v \in V(P)} D_v. \tag{2.5}$$

Let

$$\mathrm{Sol}(P) \subseteq \mathcal{X}(P) \tag{2.6}$$

denote the set of complete feasible reconstructions.

A declared computational task is denoted by

$$\tau. \tag{2.7}$$

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

$$\mathfrak{M}(P, \tau) = (\mathrm{Adm}_{P,\tau}, \mathcal{R}, \Lambda, \delta, \mathrm{Sig}_\tau)\,, \tag{2.8}$$

where:

- $\mathrm{Adm}_{P,\tau}$ is the admissibility predicate for reconstruction orderings;
- $\mathcal{R}$ specifies the reachable prefix-state spaces;
- Λ specifies the admissible next reconstruction values;
- δ specifies the stage-to-stage transition maps;
- Sig_τ specifies a reconstruction-safe, task-relative future-signature family.

The components $\mathcal{R}$, Λ, δ, and Sig_τ denote parameterized families indexed jointly by the admissible ordering π and the reconstruction stage i — that is, $\mathcal{R} = (\mathcal{R}_i^\pi(P))_{\pi,i}$, $\Lambda = (\Lambda_i^\pi)_{\pi,i}$, and $\delta = (\delta_i^\pi)_{\pi,i}$ — rather than single fixed objects independent of the ordering.

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

Unless stated otherwise, every subsequent construction is understood relative to these fixed declarations.

2.2 Admissible Reconstruction Orderings

A reconstruction ordering is a permutation

$$\pi = \big(v_{\pi(1)}, v_{\pi(2)}, \ldots, v_{\pi(n)}\big). \tag{2.9}$$

Definition 2.2 (Admissible Reconstruction Ordering)**.** An ordering π is admissible for the model $\mathfrak{M}(P, \tau)$ if

$$\mathrm{Adm}_{P,\tau}(\pi) = 1. \tag{2.10}$$

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

$$\Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M}) = \{\pi \in \mathrm{Sym}(V(P)) : \mathrm{Adm}_{P,\tau}(\pi) = 1\}\,. \tag{2.11}$$

Assume throughout that

$$\Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M}) \neq \varnothing. \tag{2.12}$$

When both the task τ and reconstruction model $\mathfrak{M}$ are fixed throughout a section, the shorter notation

$$\Pi_{\mathrm{adm}}(P) \tag{2.13}$$

may be used as a local abbreviation.

2.3 Processed Prefix and Unresolved Suffix

Let

$$\pi \in \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M}) \tag{2.14}$$

be fixed.

After reconstruction stage i, the processed prefix is

$$V_i^\pi = \{v_{\pi(1)}, \dots, v_{\pi(i)}\}, \qquad 0 \le i \le n. \tag{2.15}$$

The unresolved suffix is

$$\overline{V_i^\pi} = V(P) \setminus V_i^\pi. \tag{2.16}$$

The boundary cases are

$$V_0^\pi = \varnothing, \qquad V_n^\pi = V(P), \tag{2.17}$$

and

$$\overline{V_0^\pi} = V(P), \qquad \overline{V_n^\pi} = \varnothing. \tag{2.18}$$

The complete prefix-assignment space at stage i is

$$\mathcal{A}_i^\pi(P) = \prod_{v \in V_i^\pi} D_v. \tag{2.19}$$

The empty Cartesian product is interpreted as the singleton containing the empty assignment:

$$\mathcal{A}_0^\pi(P) = \{\varnothing\}. \tag{2.20}$$

The complete suffix-assignment space at stage i is

$$\mathcal{C}_i^\pi(P) = \prod_{v \in \overline{V_i^\pi}} D_v. \tag{2.21}$$

Accordingly,

$$\mathcal{C}_n^\pi(P) = \{\varnothing\}. \tag{2.22}$$

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.

Reachability is determined exclusively by the declared transition system and not by global feasibility.

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

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

Since $V_i^\pi \cap \overline{V_i^\pi} = \varnothing$, the union $r \cup s$ is a well-defined complete assignment.

At a fixed reconstruction stage i, all sets $E_i^\pi(r)$ are subsets of the same continuation universe $\mathcal{C}_i^\pi(P)$.

The continuation set itself is independent of the computational task. Task dependence enters only through the evaluation map.

The declared task τ determines a stage-relative evaluation map

$$\Theta_{\tau,i}^\pi : \mathcal{R}_i^\pi(P) \times \mathcal{P}\big(\mathcal{C}_i^\pi(P)\big) \longrightarrow Y_{\tau,i}.$$

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 a counting task, one may use $\mathrm{Ans}_{\mathrm{count},i}^\pi(r) = |E_i^\pi(r)|$.

For an enumeration task, one may use $\mathrm{Ans}_{\mathrm{enum},i}^\pi(r) = \{r \cup s : s \in E_i^\pi(r)\}$.

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

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

For every admissible next value $a \in \Lambda_i^\pi(r)$, the transition map is

$$\delta_i^\pi(r, a) = r \cup \{w_i^\pi \mapsto a\}. \tag{2.23}$$

Thus,

$$\delta_i^\pi(r, a) \in \mathcal{R}_{i+1}^\pi(P). \tag{2.24}$$

2.7 Reconstruction-Safe Future Signatures

A task value alone may be too coarse for a stage-wise reconstruction process.

For example, 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.

For every admissible ordering and reconstruction stage, let $\Sigma_{\tau,i}^\pi$ denote the abstract signature space associated with the declared reconstruction model.

Definition 2.4 (Reconstruction-Safe Future-Signature Family)**.** A family of maps

$$\mathrm{Sig}_{\tau,i}^\pi : \mathcal{R}_i^\pi(P) \longrightarrow \Sigma_{\tau,i}^\pi, \qquad 0 \le i \le n, \tag{2.25}$$

is called reconstruction-safe if the following conditions hold.

S1 — Task preservation.

For all reachable states $r, r' \in \mathcal{R}_i^\pi(P)$,

$$\mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r') \tag{2.26}$$

implies

$$\mathrm{Ans}_{\tau,i}^\pi(r) = \mathrm{Ans}_{\tau,i}^\pi(r'). \tag{2.27}$$

S2 — Preservation of admissible next values.
For every $i < n$,

$$\mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r') \tag{2.28}$$

implies

$$\Lambda_i^\pi(r) = \Lambda_i^\pi(r'). \tag{2.29}$$

S3 — Successor congruence.
If

$$\mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r') \tag{2.30}$$

and

$$a \in \Lambda_i^\pi(r) = \Lambda_i^\pi(r'), \tag{2.31}$$

then

$$\mathrm{Sig}_{\tau,i+1}^\pi(\delta_i^\pi(r,a)) = \mathrm{Sig}_{\tau,i+1}^\pi(\delta_i^\pi(r',a)). \tag{2.32}$$

The reconstruction-safe signature therefore defines a task-relative transition congruence rather than merely an equality relation over feasible continuation sets.

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

$$\mathrm{Sig}_{\tau,i}^\pi(r) = r \tag{2.33}$$

is always reconstruction-safe.

Therefore, 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.

2.8 Task-Sufficient Active Reconstruction Interface

Definition 2.5 (Task-Sufficient Interface)**.** Let i be a reconstruction stage.
A set

$$B \subseteq V_i^\pi \tag{2.34}$$

is called a task-sufficient reconstruction interface if

$$\forall r, r' \in \mathcal{R}_i^\pi(P): \qquad r|_B = r'|_B \Longrightarrow \mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r'). \tag{2.35}$$

This condition guarantees that the complete reconstruction-safe future signature is determined by the values retained on B.

Therefore, 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

$$\mathfrak{B}_i^{\pi,\tau}(P) = \left\{ B \subseteq V_i^\pi : \begin{array}{l} \forall r, r' \in \mathcal{R}_i^\pi(P), \\ r|_B = r'|_B \Rightarrow \mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r') \end{array} \right\}. \tag{2.36}$$

Lemma 2.1 (Existence of a Task-Sufficient Interface)**.** *For every reconstruction stage i,*

$$\mathfrak{B}_i^{\pi,\tau}(P) \neq \varnothing. \tag{2.37}$$

Proof. The complete processed prefix V_i^π is task-sufficient.

Indeed, if

$$r|_{V_i^\pi} = r'|_{V_i^\pi}, \tag{2.38}$$

then

$$r = r'. \tag{2.39}$$

Consequently,

$$\mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r'). \tag{2.40}$$

Therefore,

$$V_i^\pi \in \mathfrak{B}_i^{\pi,\tau}(P). \tag{2.41}$$

□

2.9 Interface Factorization

For

$$B \subseteq V_i^\pi, \tag{2.42}$$

define the coordinate projection

$$\rho_{i,B}^\pi : \mathcal{R}_i^\pi(P) \longrightarrow \prod_{v \in B} D_v \tag{2.43}$$

by

$$\rho_{i,B}^\pi(r) = r|_B. \tag{2.44}$$

Proposition 2.1 (Interface Factorization)**.** *A set $B \subseteq V_i^\pi$ is task-sufficient if and only if there exists a unique map on the projected reachable interface-state space*

$$\overline{\mathrm{Sig}}_{\tau,i,B}^\pi : \rho_{i,B}^\pi\big(\mathcal{R}_i^\pi(P)\big) \longrightarrow \Sigma_{\tau,i}^\pi \tag{2.45}$$

such that

$$\mathrm{Sig}_{\tau,i}^\pi = \overline{\mathrm{Sig}}_{\tau,i,B}^\pi \circ \rho_{i,B}^\pi. \tag{2.46}$$

Proof. Assume first that B is task-sufficient.
For

$$z \in \rho^{\pi}_{i,B}(\mathcal{R}^{\pi}_{i}(P)), \tag{2.47}$$

choose a reachable state r satisfying

$$\rho^{\pi}_{i,B}(r) = z \tag{2.48}$$

and define

$$\overline{\mathrm{Sig}}^{\pi}_{\tau,i,B}(z) = \mathrm{Sig}^{\pi}_{\tau,i}(r). \tag{2.49}$$

If r' is another reachable state satisfying

$$\rho^{\pi}_{i,B}(r') = z, \tag{2.50}$$

then

$$r|_B = r'|_B. \tag{2.51}$$

Task sufficiency gives

$$\mathrm{Sig}^{\pi}_{\tau,i}(r) = \mathrm{Sig}^{\pi}_{\tau,i}(r'). \tag{2.52}$$

Hence, the map is well defined: the definition is independent of the chosen representative.
Conversely, assume that the factorization exists.
If

$$r|_B = r'|_B, \tag{2.53}$$

then

$$\rho^{\pi}_{i,B}(r) = \rho^{\pi}_{i,B}(r'). \tag{2.54}$$

Therefore,

$$\begin{aligned} \mathrm{Sig}^{\pi}_{\tau,i}(r) &= \overline{\mathrm{Sig}}^{\pi}_{\tau,i,B}(\rho^{\pi}_{i,B}(r)) && (2.55)\\ &= \overline{\mathrm{Sig}}^{\pi}_{\tau,i,B}(\rho^{\pi}_{i,B}(r')) && (2.56)\\ &= \mathrm{Sig}^{\pi}_{\tau,i}(r'). && (2.57) \end{aligned}$$

Thus, B is task-sufficient.
Uniqueness follows because every element of the domain is the projection of at least one reachable prefix state.

□

Corollary 2.1 (Well-Defined Admissible Next-Value Set on the Interface)**.** *Let $B = B_i^{\pi,\tau}$ be a task-sufficient active interface at a nonterminal stage $i < n$. For $u \in \rho^{\pi}_{i,B}(\mathcal{R}^{\pi}_{i}(P))$, define*

$$\widehat{\Lambda}^{\pi,\tau}_{i}(u) = \Lambda^{\pi}_{i}(r). \tag{2.58}$$

for any $r \in \mathcal{R}^{\pi}_{i}(P)$ with $r|_B = u$. This is independent of the choice of r.

Proof. Let $r, r' \in \mathcal{R}^{\pi}_{i}(P)$ satisfy $r|_B = r'|_B = u$. By the Interface Factorization Proposition, $\mathrm{Sig}^{\pi}_{\tau,i}(r) = \mathrm{Sig}^{\pi}_{\tau,i}(r')$. By S2 (Preservation of Admissible Next Values), $\Lambda^{\pi}_{i}(r) = \Lambda^{\pi}_{i}(r')$.

□

The set $\widehat{\Lambda}_i^{\pi,\tau}(u)$ is therefore a well-defined function of the interface state u alone, independently of which reachable prefix state projects onto it.

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

$$b_i^{\pi,\tau}(P) = \min\left\{|B| : B \in \mathfrak{B}_i^{\pi,\tau}(P)\right\}. \tag{2.59}$$

The minimum exists because

$$\mathfrak{B}_i^{\pi,\tau}(P) \neq \varnothing \tag{2.60}$$

and V_i^{π} is finite.

One may select any cardinality-minimal task-sufficient set as the active reconstruction interface:

$$B_i^{\pi,\tau} \in \operatorname*{arg\,min}_{B \in \mathfrak{B}_i^{\pi,\tau}(P)} |B|. \tag{2.61}$$

The minimizing set need not be unique, but every minimizing set has the same cardinality:

$$|B_i^{\pi,\tau}| = b_i^{\pi,\tau}(P). \tag{2.62}$$

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.

Whenever multiple cardinality-minimal interfaces exist, $B_i^{\pi,\tau}$ denotes one selected representative from the corresponding arg min family.

The complete selected interface sequence is

$$\mathcal{B}(P,\pi,\tau) = (B_0^{\pi,\tau}, B_1^{\pi,\tau}, \ldots, B_n^{\pi,\tau}). \tag{2.63}$$

Since

$$\mathcal{R}_0^{\pi}(P) = \{\varnothing\}, \tag{2.64}$$

one has

$$b_0^{\pi,\tau}(P) = 0 \tag{2.65}$$

and the initial interface may be selected as

$$B_0^{\pi,\tau} = \varnothing. \tag{2.66}$$

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 the 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

$$r, r' \in \mathcal{R}_i^\pi(P) \tag{2.67}$$

such that

$$r|_{V_i^\pi \setminus \{v\}} = r'|_{V_i^\pi \setminus \{v\}}, \tag{2.68}$$

but

$$\mathrm{Sig}_{\tau,i}^\pi(r) \neq \mathrm{Sig}_{\tau,i}^\pi(r'). \tag{2.69}$$

Then every task-sufficient interface contains v.

Proof. Let B be task-sufficient and suppose that

$$v \notin B. \tag{2.70}$$

Then

$$B \subseteq V_i^\pi \setminus \{v\}. \tag{2.71}$$

Consequently,

$$r|_B = r'|_B. \tag{2.72}$$

Task sufficiency would imply

$$\mathrm{Sig}_{\tau,i}^\pi(r) = \mathrm{Sig}_{\tau,i}^\pi(r'), \tag{2.73}$$

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

$$\mathcal{R}_i^\pi(P) = \{00, 11\}, \tag{2.74}$$

with

$$\mathrm{Sig}_{\tau,i}^\pi(00) \neq \mathrm{Sig}_{\tau,i}^\pi(11). \tag{2.75}$$

There are no two reachable states that differ only in a, and there are no two reachable states 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

$$\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = \max_{0\le i\le n} b_i^{\pi,\tau}(P). \tag{2.76}$$

The value

$$\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \tag{2.77}$$

is the actual minimum number of components that must remain simultaneously active at the widest reconstruction stage.

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

$$\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = \max\left\{0,\, \widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) - 1\right\}. \tag{2.78}$$

The subtraction by one is an optional normalization convention chosen to facilitate comparison with classical width parameters, including treewidth and pathwidth. The primary CPR invariant is the unnormalized quantity $\widehat{\omega}_{\mathrm{rec}}$.

It is not forced by the reconstruction model.

Accordingly,

$$\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = 0 \tag{2.79}$$

may still mean that one component must remain active.

The unnormalized quantity $\widehat{\omega}_{\mathrm{rec}}$ is the primary structural invariant, whereas ω_{rec} is a derived normalized convention.

When τ and $\mathfrak{M}$ are fixed, the shorter forms

$$\widehat{\omega}_{\mathrm{rec}}(P,\pi) \tag{2.80}$$

and

$$\omega_{\mathrm{rec}}(P,\pi) \tag{2.81}$$

may be used.

2.13 Interface Bound

Lemma 2.3 (Interface Bound)**.** *For every stage i,*

$$b_i^{\pi,\tau}(P) \le \widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}). \tag{2.82}$$

Moreover,

$$\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \le \omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) + 1, \tag{2.83}$$

with equality whenever $\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) \ge 1$; the inequality is strict only at the boundary case $\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = 0$.

Consequently, for every selected minimum active interface,

$$|B_i^{\pi,\tau}| \le \omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) + 1. \tag{2.84}$$

Proof. The first inequality follows immediately from

$$\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}) = \max_{0\le j\le n} b_j^{\pi,\tau}(P). \tag{2.85}$$

Let

$$M = \widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}). \tag{2.86}$$

If $M = 0$, then by definition

$$\omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = 0. \tag{2.87}$$

Hence,

$$M = 0 \leq \omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) + 1. \tag{2.88}$$

If $M \geq 1$, then

$$\omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = M - 1, \tag{2.89}$$

and therefore

$$M = \omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) + 1. \tag{2.90}$$

Finally,

$$|B_i^{\pi,\tau}| = b_i^{\pi,\tau}(P), \tag{2.91}$$

because $B_i^{\pi,\tau}$ is cardinality-minimal.

□

2.14 Minimum CPR-Width over Admissible Orderings

Because

$$\Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M}) \neq \varnothing, \tag{2.92}$$

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

$$\omega_{\mathrm{rec}}(P, \tau; \mathfrak{M}) = \min_{\pi \in \Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})} \omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}). \tag{2.93}$$

The corresponding unnormalized minimum is

$$\widehat{\omega}_{\mathrm{rec}}(P, \tau; \mathfrak{M}) = \min_{\pi \in \Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})} \widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}). \tag{2.94}$$

When the task and reconstruction model are fixed, these quantities may be written as

$$\omega_{\mathrm{rec}}(P) \tag{2.95}$$

and

$$\widehat{\omega}_{\mathrm{rec}}(P). \tag{2.96}$$

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*

$$\omega_{\mathrm{rec}}(P, \pi_1, \tau; \mathfrak{M}) \neq \omega_{\mathrm{rec}}(P, \pi_2, \tau; \mathfrak{M}). \tag{2.97}$$

Proof. Let

$$V(P) = \{a, b, c\} \tag{2.98}$$

with binary domains

$$D_a = D_b = D_c = \{0, 1\}. \tag{2.99}$$

Impose the constraints

$$a = c \tag{2.100}$$

and

$$b = c. \tag{2.101}$$

Thus,

$$\mathrm{Sol}(P) = \{000, 111\}. \tag{2.102}$$

Let the declared task be decision.

A partial assignment is reachable if every constraint whose complete scope has already been processed is satisfied.

Reachability in this model therefore refers to transition admissibility and not to existence of a complete feasible extension.

Constraints involving unprocessed components are not evaluated at this stage; therefore a reachable prefix may later become infeasible after additional components are reconstructed.

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

$$\pi_1 = (a, b, c). \tag{2.103}$$

After stage 2, neither equality constraint has been completely processed, because both contain c.

Therefore,

$$\mathcal{R}_2^{\pi_1}(P) = \{00, 01, 10, 11\}. \tag{2.104}$$

The admissible next-value sets for c are

$$\Lambda_2^{\pi_1}(00) = \{0\}, \tag{2.105}$$

$$\Lambda_2^{\pi_1}(11) = \{1\}, \tag{2.106}$$

and

$$\Lambda_2^{\pi_1}(01) = \Lambda_2^{\pi_1}(10) = \varnothing. \tag{2.107}$$

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,

$$b_2^{\pi_1,\mathrm{dec}}(P) = 2. \tag{2.108}$$

At all other stages, an interface of size at most one is sufficient. Therefore,

$$\widehat{\omega}_{\mathrm{rec}}(P, \pi_1, \mathrm{dec}; \mathfrak{M}) = 2 \tag{2.109}$$

and

$$\omega_{\mathrm{rec}}(P, \pi_1, \mathrm{dec}; \mathfrak{M}) = 1. \tag{2.110}$$

Now consider the ordering

$$\pi_2 = (a, c, b). \tag{2.111}$$

After stage 2, the constraint $a = c$ has already been processed. The inconsistent prefixes are therefore removed, and

$$\mathcal{R}_2^{\pi_2}(P) = \{00, 11\}. \tag{2.112}$$

The admissible next-value sets for b are

$$\Lambda_2^{\pi_2}(00) = \{0\} \tag{2.113}$$

and

$$\Lambda_2^{\pi_2}(11) = \{1\}. \tag{2.114}$$

Either component a or component c distinguishes these two reachable states.

Thus,

$$b_2^{\pi_2,\mathrm{dec}}(P) = 1. \tag{2.115}$$

At every stage, an interface of size at most one is task-sufficient. Hence,

$$\widehat{\omega}_{\mathrm{rec}}(P, \pi_2, \mathrm{dec}; \mathfrak{M}) = 1 \tag{2.116}$$

and

$$\omega_{\mathrm{rec}}(P, \pi_2, \mathrm{dec}; \mathfrak{M}) = 0. \tag{2.117}$$

Therefore,

$$\omega_{\mathrm{rec}}(P, \pi_1, \mathrm{dec}; \mathfrak{M}) \neq \omega_{\mathrm{rec}}(P, \pi_2, \mathrm{dec}; \mathfrak{M}). \tag{2.118}$$

□

2.16 Structural Interpretation

Controlled Partial Reconstruction Width does not measure the total size of the complete candidate space.

The unnormalized quantity

$$\widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) \tag{2.119}$$

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

$$\omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) \tag{2.120}$$

is obtained by the adopted minus-one convention.

A task-sufficient interface satisfies

$$r|_B = r'|_B \Longrightarrow \mathrm{Sig}^{\pi}_{\tau,i}(r) = \mathrm{Sig}^{\pi}_{\tau,i}(r'). \tag{2.121}$$

Equivalently, the future-signature map factors through the interface projection:

$$\mathrm{Sig}^{\pi}_{\tau,i} = \overline{\mathrm{Sig}}^{\pi}_{\tau,i,B} \circ \rho^{\pi}_{i,B}. \tag{2.122}$$

This factorization is the formal statement that all jointly forgotten components are semantically and transitionally unnecessary for the declared reconstruction task.

CPR-Width is therefore dependent 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,

$$B_i^{\pi,\mathrm{dec}} \neq B_i^{\pi,\mathrm{count}} \tag{2.123}$$

and

$$\omega_{\mathrm{rec}}(P, \pi, \mathrm{dec}; \mathfrak{M}) \neq \omega_{\mathrm{rec}}(P, \pi, \mathrm{count}; \mathfrak{M}) \tag{2.124}$$

may occur.

CPR-Width counts active components. It does not directly measure their bit-level information content.

Consequently, CPR-Width is invariant under any representation-preserving renaming of component identifiers.

Local domain sizes, projected interface-state spaces, reachable interface states, and semantic quotient states are developed in the following chapter.

Chapter Summary

The constructions of this chapter follow one non-circular chain. The raw prefix-state space $\mathcal{A}_i^\pi(P)$ is first restricted to the reachable prefix-state space $\mathcal{R}_i^\pi(P)$ by the declared admissibility and transition rules. The reconstruction-safe future-signature map $\mathrm{Sig}_{\tau,i}^\pi$ then identifies reachable states that are indistinguishable for the declared task and admissible transitions (conditions S1–S3), yielding a task-sufficient active interface $B_i^{\pi,\tau}$ of minimum cardinality. Controlled Partial Reconstruction Width, $\omega_{\mathrm{rec}}(P,\pi)$, records the minimum interface cardinality required at the widest reconstruction stage; the semantic-state count $S_{\mathrm{CPR}}(P,\pi;\tau)$ (formally introduced in Chapter 3), where S_{CPR} denotes the number of equivalence classes induced by the reconstruction-safe semantic quotient, records the largest number of task-relevant signature classes encountered along the same reconstruction. Neither quantity requires literal equality of continuation sets, only preservation of the declared task value and of the admissible transition structure. This yields the structural relation

$$\omega_{\mathrm{rec}}(P,\pi) \quad \text{and} \quad S_{\mathrm{CPR}}(P,\pi;\tau),$$

where the first quantity measures active interface size and the second measures semantic-state compression, which the following chapter extends to the complete operation-count model $N_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M})$.

2.17 Future Continuations and Task Signatures

The purpose of semantic-state compression is not to preserve every syntactic difference between reachable interface states.

Instead, it preserves exactly the normalized information required by the declared computational task.

Let

$$u \in \mathcal{U}_i^{\mathrm{reach}}(P,\pi)$$

be a reachable interface state at reconstruction stage i.

Let

$$\overline{V}_i^\pi = V(P) \setminus V_i^\pi$$

denote the unresolved suffix.

Definition 2.10 (Continuation Universe)**.** For a fixed problem instance P, admissible reconstruction ordering π, and reconstruction stage i, let

$$\mathcal{C}_i(P,\pi) = \prod_{v \in \overline{V}_i^\pi} D_v \tag{2.125}$$

denote the common universe of complete assignments to the unresolved suffix $\overline{V}_i^\pi$.

For every reachable state

$$u \in \mathcal{U}_i^{\mathrm{reach}}(P,\pi),$$

the admissible future-continuation set of u is a subset

$$E_i(u) \subseteq \mathcal{C}_i(P,\pi). \tag{2.126}$$

The use of the common continuation universe $\mathcal{C}_i(P,\pi)$ ensures that all continuation sets $E_i(u)$ belong to the same power set

$$\mathcal{P}(\mathcal{C}_i(P,\pi)).$$

Consequently, they may be evaluated by one common task map.

Definition 2.11 (Task-Evaluation Map)**.** Let τ be a declared computational task.

A task-evaluation map at stage i is a function

$$\Theta^{\tau}_{P,i} : \mathcal{P}(\mathcal{C}_i(P,\pi)) \longrightarrow \mathcal{Y}^{\tau}_{P,i}, \tag{2.127}$$

where

$$\mathcal{Y}^{\tau}_{P,i}$$

is the task-dependent evaluation space.
The value

$$\Theta^{\tau}_{P,i}(E_i(u)) \tag{2.128}$$

records the information extracted from the continuation set $E_i(u)$ that is relevant to the declared task τ.
The task-evaluation map must preserve exactly the information required by the declared task and no less: $\Theta^{\tau}_{P,i}$ must be task-sufficient in the sense of the task-sufficiency requirement of Chapter 2.

The task-evaluation map need not preserve the complete continuation set.
For a decision task, one may define

$$\Theta^{\mathrm{dec}}_{P,i}(E) = \mathbf{1}[E \neq \varnothing]. \tag{2.129}$$

For a counting task, one may define

$$\Theta^{\mathrm{count}}_{P,i}(E) = |E|. \tag{2.130}$$

For an enumeration task, one may use the identity evaluation

$$\Theta^{\mathrm{enum}}_{P,i}(E) = E. \tag{2.131}$$

For an optimization task, the evaluation map may preserve a declared optimum value, a set of optimal continuations, or another explicitly specified optimization certificate.

Definition 2.12 (Signature Normalization)**.** Let

$$\mathcal{Y}^{\tau}_{P,i}$$

be the task-evaluation space.
A signature-normalization map is a function

$$\nu^{\tau}_{P,i} : \mathcal{Y}^{\tau}_{P,i} \longrightarrow \widehat{\mathcal{Y}}^{\tau}_{P,i} \tag{2.132}$$

that assigns a canonical representative to every task-evaluation value.
The normalization is required to be idempotent on its image:

$$\nu^{\tau}_{P,i}(\nu^{\tau}_{P,i}(y)) = \nu^{\tau}_{P,i}(y) \tag{2.133}$$

whenever the expression is defined.

Normalization removes representational differences that do not alter the declared task value.
When the task-evaluation space already has a canonical representation, one may take

$$\nu^{\tau}_{P,i} = \mathrm{id}_{\mathcal{Y}^{\tau}_{P,i}}. \tag{2.134}$$

Definition 2.13 (Normalized Future Signature)**.** For every reachable state

$$u \in \mathcal{U}^{\mathrm{reach}}_i(P,\pi),$$

the normalized future signature of u for the declared task τ is

$$\widehat{\sigma}^{\tau}_{P,i}(u) = \nu^{\tau}_{P,i}\left(\Theta^{\tau}_{P,i}(E_i(u))\right). \tag{2.135}$$

Thus,

$$\widehat{\sigma}^{\tau}_{P,i} : \mathcal{U}_i^{\mathrm{reach}}(P,\pi) \longrightarrow \widehat{\mathcal{Y}}^{\tau}_{P,i}. \tag{2.136}$$

The normalized future signature is the object used for semantic comparison.
It is not, in general, identical to the complete continuation set.

2.18 Task-Relative Signature Equivalence

Throughout this section, P, π, i, and τ are assumed fixed. The dependence on these parameters is therefore omitted from the notation $\equiv^{\tau}_{P,i}$ when no ambiguity arises; in particular, the relation depends on the reconstruction ordering π through the reachable state space $\mathcal{U}_i^{\mathrm{reach}}(P,\pi)$ and the continuation sets $E_i(u)$.

Definition 2.14 (Task-Relative Signature Equivalence)**.** For reachable states

$$u, v \in \mathcal{U}_i^{\mathrm{reach}}(P,\pi),$$

define

$$u \equiv^{\tau}_{P,i} v \quad \Longleftrightarrow \quad \widehat{\sigma}^{\tau}_{P,i}(u) = \widehat{\sigma}^{\tau}_{P,i}(v). \tag{2.137}$$

If the declared task τ is fixed throughout the discussion, the superscript may be omitted and the relation may be written as

$$u \equiv_{P,i} v. \tag{2.138}$$

The relation compares normalized task signatures.
It does not compare continuation sets directly.

Proposition 2.3 (Signature Equivalence Is an Equivalence Relation)**.** *For every fixed problem instance P, admissible reconstruction ordering π, stage i, and declared task τ, the relation*

$$\equiv^{\tau}_{P,i}$$

is an equivalence relation on

$$\mathcal{U}_i^{\mathrm{reach}}(P,\pi).$$

Proof. Reflexivity follows because, for every reachable state u,

$$\widehat{\sigma}^{\tau}_{P,i}(u) = \widehat{\sigma}^{\tau}_{P,i}(u).$$

Symmetry follows because, if

$$\widehat{\sigma}^{\tau}_{P,i}(u) = \widehat{\sigma}^{\tau}_{P,i}(v),$$

then

$$\widehat{\sigma}^{\tau}_{P,i}(v) = \widehat{\sigma}^{\tau}_{P,i}(u).$$

Transitivity follows because, if

$$\widehat{\sigma}^{\tau}_{P,i}(u) = \widehat{\sigma}^{\tau}_{P,i}(v)$$

and

$$\widehat{\sigma}^{\tau}_{P,i}(v) = \widehat{\sigma}^{\tau}_{P,i}(w),$$

then

$$\widehat{\sigma}^{\tau}_{P,i}(u) = \widehat{\sigma}^{\tau}_{P,i}(w).$$

Therefore,

$$\equiv^{\tau}_{P,i}$$

is reflexive, symmetric, and transitive.

□

Remark 2.2 (Signature Equality Does Not Imply Set Equality)**.** In general,

$$u \equiv^{\tau}_{P,i} v \tag{2.139}$$

implies only

$$\widehat{\sigma}^{\tau}_{P,i}(u) = \widehat{\sigma}^{\tau}_{P,i}(v). \tag{2.140}$$

It does not imply

$$E_i(u) = E_i(v). \tag{2.141}$$

Indeed, two different continuation sets may have the same decision value, the same cardinality, or the same optimum value.

Equality of continuation sets follows only under an additional injectivity hypothesis on the normalized task-evaluation map.

Proposition 2.4 (Condition for Exact Continuation-Set Recovery)**.** *Define the reachable continuation family*

$$\mathfrak{E}_i(P,\pi) = \left\{ E_i(u) : u \in \mathcal{U}_i^{\mathrm{reach}}(P,\pi) \right\}, \qquad \mathfrak{E}_i(P,\pi) \subseteq \mathcal{P}\big(\mathcal{C}_i(P,\pi)\big). \tag{2.142}$$

Assume that the composition

$$\nu^{\tau}_{P,i} \circ \Theta^{\tau}_{P,i} : \mathfrak{E}_i(P,\pi) \longrightarrow \widehat{\mathcal{Y}}^{\tau}_{P,i} \tag{2.143}$$

is injective on $\mathfrak{E}_i(P,\pi)$.

Then, for all reachable states u *and* v*,*

$$u \equiv^{\tau}_{P,i} v \quad \Longleftrightarrow \quad E_i(u) = E_i(v). \tag{2.144}$$

Proof. If

$$E_i(u) = E_i(v),$$

then applying the same task-evaluation and normalization maps yields

$$\widehat{\sigma}^{\tau}_{P,i}(u) = \widehat{\sigma}^{\tau}_{P,i}(v),$$

and hence

$$u \equiv^{\tau}_{P,i} v.$$

Conversely, assume

$$u \equiv^{\tau}_{P,i} v.$$

Then

$$\nu^{\tau}_{P,i}\left(\Theta^{\tau}_{P,i}(E_i(u))\right) = \nu^{\tau}_{P,i}\left(\Theta^{\tau}_{P,i}(E_i(v))\right).$$

By injectivity on $\mathfrak{E}_i(P,\pi)$, it follows that

$$E_i(u) = E_i(v).$$

□

For the enumeration task with identity evaluation and identity normalization, the injectivity condition is satisfied.

For decision, counting, and many optimization tasks, it generally is not.

The equivalence relation is task-relative.

Two states may be equivalent for a decision task while remaining distinct for counting, enumeration, or optimization.

Therefore, in general,

$$\equiv^{\mathrm{dec}}_{P,i}, \qquad \equiv^{\mathrm{count}}_{P,i}, \qquad \equiv^{\mathrm{enum}}_{P,i}, \qquad \equiv^{\mathrm{opt}}_{P,i} \tag{2.145}$$

need not coincide.

A semantic compression that is sound for one declared task may be unsound or incomplete for another.

2.19 Semantic Quotient

For a fixed declared task τ, define the semantic state space at stage i by

$$\mathcal{S}^{\tau}_i(P,\pi) = \mathcal{U}^{\mathrm{reach}}_i(P,\pi)/\equiv^{\tau}_{P,i}. \tag{2.146}$$

If the declared task is fixed throughout the discussion, one may write

$$\mathcal{S}_i(P,\pi) = \mathcal{U}^{\mathrm{reach}}_i(P,\pi)/\equiv_{P,i}. \tag{2.147}$$

The quotient map is

$$q^{\tau}_i : \mathcal{U}^{\mathrm{reach}}_i(P,\pi) \twoheadrightarrow \mathcal{S}^{\tau}_i(P,\pi), \qquad q^{\tau}_i(u) = [u]_{\equiv^{\tau}_{P,i}}. \tag{2.148}$$

Proposition 2.5 (Signature Representation of the Semantic Quotient)**.** *The semantic quotient is canonically bijective to the image of the normalized future-signature map:*

$$\mathcal{S}^{\tau}_i(P,\pi) \cong \widehat{\sigma}^{\tau}_{P,i}\left(\mathcal{U}^{\mathrm{reach}}_i(P,\pi)\right). \tag{2.149}$$

Proof. Define

$$\overline{\sigma}^{\tau}_{P,i} : \mathcal{S}^{\tau}_i(P,\pi) \longrightarrow \widehat{\sigma}^{\tau}_{P,i}\left(\mathcal{U}^{\mathrm{reach}}_i(P,\pi)\right)$$

by

$$\overline{\sigma}^{\tau}_{P,i}\left([u]_{\equiv^{\tau}_{P,i}}\right) = \widehat{\sigma}^{\tau}_{P,i}(u). \tag{2.150}$$

The map is well-defined because all members of one equivalence class have the same normalized future signature.

It is surjective by construction.

It is injective because two equivalence classes with the same signature must coincide by Definition 2.14. Therefore,

$$\overline{\sigma}^{\tau}_{P,i}$$

is a bijection.

□

For every nonterminal stage $i < m_{\mathrm{stage}}(P,\pi)$, the raw transition $\delta^{\pi}_i(r,a) = r \cup \{w^{\pi}_i \mapsto a\}$ of Chapter 2 propagates to a transition on semantic states only if the normalized future signature is compatible with it in the following sense.

Definition 2.15 (Successor Congruence for the Normalized Future Signature)**.** The normalized future-signature family $\{\widehat{\sigma}^{\tau}_{P,i}\}_{0\le i\le m_{\mathrm{stage}}(P,\pi)}$ satisfies successor congruence if, for all $r, r' \in \mathcal{R}^{\pi}_i(P)$ with

$$\widehat{\sigma}^{\tau}_{P,i}(r|_{B^{\pi,\tau}_i}) = \widehat{\sigma}^{\tau}_{P,i}(r'|_{B^{\pi,\tau}_i}),$$

and for every $a \in \widehat{\Lambda}^{\pi,\tau}_i(r|_{B^{\pi,\tau}_i})$ (Corollary 2.1 of Chapter 2),

$$\widehat{\sigma}^{\tau}_{P,i+1}\Big(\delta^{\pi}_i(r,a)\big|_{B^{\pi,\tau}_{i+1}}\Big) = \widehat{\sigma}^{\tau}_{P,i+1}\Big(\delta^{\pi}_i(r',a)\big|_{B^{\pi,\tau}_{i+1}}\Big).$$

This is the interface-level counterpart of condition S3 (Successor Congruence) required of $\mathrm{Sig}^{\pi}_{\tau,i}$ in Chapter 2, and it is required of $\widehat{\sigma}^{\tau}_{P,i}$ for the same reason: without it, the semantic quotient could not be propagated consistently from one reconstruction stage to the next. A declared CPR model is required to satisfy successor congruence for its normalized future-signature family.

Definition 2.16 (Quotient Transition)**.** Assume successor congruence for $\{\widehat{\sigma}^{\tau}_{P,i}\}_{0\le i\le m_{\mathrm{stage}}(P,\pi)}$. For a semantic state $[u]_{\equiv^{\tau}_{P,i}} \in \mathcal{S}^{\tau}_i(P,\pi)$ and an admissible next value $a \in \widehat{\Lambda}^{\pi,\tau}_i(u)$, choose any $r \in \mathcal{R}^{\pi}_i(P)$ with $r|_{B^{\pi,\tau}_i} = u$, and define the quotient transition by

$$\overline{\delta}^{\pi,\tau}_i\big([u]_{\equiv^{\tau}_{P,i}}, a\big) = \Big[\delta^{\pi}_i(r,a)\big|_{B^{\pi,\tau}_{i+1}}\Big]_{\equiv^{\tau}_{P,i+1}}. \tag{2.151}$$

Proposition 2.6 (Well-Definedness of the Quotient Transition)**.** *Under successor congruence,* $\overline{\delta}^{\pi,\tau}_i\big([u]_{\equiv^{\tau}_{P,i}}, a\big)$ *does not depend on the choice of the representative* r *of the class* $[u]_{\equiv^{\tau}_{P,i}}$.

Proof. Let $r, r' \in \mathcal{R}^{\pi}_i(P)$ with $r|_{B^{\pi,\tau}_i} \equiv^{\tau}_{P,i} r'|_{B^{\pi,\tau}_i}$, that is, $\widehat{\sigma}^{\tau}_{P,i}(r|_{B^{\pi,\tau}_i}) = \widehat{\sigma}^{\tau}_{P,i}(r'|_{B^{\pi,\tau}_i})$ (Definition 2.14). By successor congruence, $\widehat{\sigma}^{\tau}_{P,i+1}(\delta^{\pi}_i(r,a)|_{B^{\pi,\tau}_{i+1}}) = \widehat{\sigma}^{\tau}_{P,i+1}(\delta^{\pi}_i(r',a)|_{B^{\pi,\tau}_{i+1}})$, that is, $\delta^{\pi}_i(r,a)|_{B^{\pi,\tau}_{i+1}} \equiv^{\tau}_{P,i+1} \delta^{\pi}_i(r',a)|_{B^{\pi,\tau}_{i+1}}$, so the two choices of representative yield the same class in $\mathcal{S}^{\tau}_{i+1}(P,\pi)$.

□

Thus,

$$\overline{\delta}^{\pi,\tau}_i : \mathcal{S}^{\tau}_i(P,\pi) \longrightarrow \mathcal{S}^{\tau}_{i+1}(P,\pi) \tag{2.152}$$

is well defined, and realizes the stage-to-stage semantic transition used in the runtime accounting of Chapter 4.

Definition 2.17 (Maximum Semantic-State Count)**.** For a fixed declared task τ, define

$$S_{\mathrm{CPR}}(P,\pi;\tau) = \max_{0\le i\le m_{\mathrm{stage}}(P,\pi)} \left|\mathcal{S}^{\tau}_i(P,\pi)\right|. \tag{2.153}$$

If τ is fixed throughout the manuscript, one may write

$$S_{\mathrm{CPR}}(P,\pi) = S_{\mathrm{CPR}}(P,\pi;\tau). \tag{2.154}$$

At every reconstruction stage,

$$|\mathcal{S}_i^{\tau}(P,\pi)| \le \left|\mathcal{U}_i^{\mathrm{reach}}(P,\pi)\right| \le |\mathcal{U}_i(P,\pi)|\,. \tag{2.155}$$

2.20 Width–State Bound

Theorem 2.1 (Width–State Bound)**.** *For every finite problem instance P, every admissible reconstruction ordering π, and every fixed declared task τ,*

$$S_{\mathrm{CPR}}(P,\pi;\tau) \le q(P)^{\omega_{\mathrm{rec}}(P,\pi)+1}. \tag{2.156}$$

Proof. For every stage i,

$$|\mathcal{S}_i^{\tau}(P,\pi)| \le \left|\mathcal{U}_i^{\mathrm{reach}}(P,\pi)\right|.$$

The reachable state space is contained in the raw interface-state space $\mathcal{U}_i(P,\pi,\tau;\mathfrak{M}) = \prod_{v\in B_i^{\pi,\tau}} D_v$ of assignments to the minimum active interface $B_i^{\pi,\tau}$ (Section 2.23). By the Interface Bound (Lemma 2.3) of Chapter 2, $|B_i^{\pi,\tau}| \le \omega_{\mathrm{rec}}(P,\pi)+1$, and therefore

$$\left|\mathcal{U}_i^{\mathrm{reach}}(P,\pi)\right| \le |\mathcal{U}_i(P,\pi,\tau;\mathfrak{M})| = q(P)^{|B_i^{\pi,\tau}|} \le q(P)^{\omega_{\mathrm{rec}}(P,\pi)+1}.$$

Hence,

$$|\mathcal{S}_i^{\tau}(P,\pi)| \le q(P)^{\omega_{\mathrm{rec}}(P,\pi)+1}.$$

Taking the maximum over all stages i yields

$$S_{\mathrm{CPR}}(P,\pi;\tau) \le q(P)^{\omega_{\mathrm{rec}}(P,\pi)+1}.$$

□

The theorem depends only on the fact that the semantic state space is a quotient of the reachable state space.

It does not require equality of continuation sets inside one semantic class.

2.21 Binary Semantic-State Dimension

Definition 2.18 (Binary Semantic-State Dimension)**.** For a fixed declared task τ, define

$$D_{\mathrm{CPR}}(P,\pi;\tau) = \left\lceil \log_2 \max\left\{1, S_{\mathrm{CPR}}(P,\pi;\tau)\right\}\right\rceil. \tag{2.157}$$

For $q(P) \ge 2$, the Width–State Bound implies

$$D_{\mathrm{CPR}}(P,\pi;\tau) \le \left\lceil \left(\omega_{\mathrm{rec}}(P,\pi)+1\right)\log_2 q(P)\right\rceil. \tag{2.158}$$

If $q(P) = 1$, every local domain is a singleton, and therefore

$$S_{\mathrm{CPR}}(P,\pi;\tau) = 1 \tag{2.159}$$

and

$$D_{\mathrm{CPR}}(P,\pi;\tau) = 0. \tag{2.160}$$

The binary semantic-state dimension gives the information-theoretic number of bits required to index one semantic class among the maximum number of possible classes.
It does not imply that the complete semantic quotient is polynomially constructible.

2.22 Task Dependence and Semantic Safety

Future-signature equivalence depends on the declared computational task.
For decision, it may be sufficient to preserve whether at least one admissible continuation exists.
For counting, the number of admissible continuations must be preserved.
For enumeration, the complete continuation set must be preserved.
For optimization, the declared optimum value or optimization certificate must be preserved.
Consequently, in general, the relations

$$\equiv_{P,i}^{\mathrm{dec}}, \qquad \equiv_{P,i}^{\mathrm{count}}, \qquad \equiv_{P,i}^{\mathrm{enum}}, \qquad \equiv_{P,i}^{\mathrm{opt}} \tag{2.161}$$

need not coincide. This indicates a possible distinction between the relations; it is not asserted to hold for every individual problem instance.

Proposition 2.7 (Task-Relative Semantic Safety)**.** *Let*

$$u, v \in \mathcal{U}_i^{\mathrm{reach}}(P, \pi).$$

If

$$u \equiv_{P,i}^{\tau} v, \tag{2.162}$$

then

$$\nu_{P,i}^{\tau}\left(\Theta_{P,i}^{\tau}(E_i(u))\right) = \nu_{P,i}^{\tau}\left(\Theta_{P,i}^{\tau}(E_i(v))\right). \tag{2.163}$$

Therefore, replacing u by v preserves the normalized result of the declared task τ.

Proof. By Definition 2.14,

$$u \equiv_{P,i}^{\tau} v$$

holds exactly when

$$\widehat{\sigma}_{P,i}^{\tau}(u) = \widehat{\sigma}_{P,i}^{\tau}(v).$$

By Definition 2.13,

$$\widehat{\sigma}_{P,i}^{\tau}(u) = \nu_{P,i}^{\tau}\left(\Theta_{P,i}^{\tau}(E_i(u))\right)$$

and

$$\widehat{\sigma}_{P,i}^{\tau}(v) = \nu_{P,i}^{\tau}\left(\Theta_{P,i}^{\tau}(E_i(v))\right).$$

Substitution yields the stated equality.

□

Remark 2.3 (Scope of Semantic Safety)**.** Proposition 2.7 establishes preservation of the declared normalized task value.
It does not establish that

$$E_i(u) = E_i(v),$$

that every individual continuation is preserved, or that the two states are interchangeable for a different computational task.

For an enumeration task with identity evaluation and identity normalization, signature equivalence does imply equality of the enumerated continuation sets.

For decision, counting, or non-injective optimization signatures, that conclusion is not available.

2.23 Structural Interpretation

For a fixed declared task τ, the state spaces developed in this chapter are related through reachability filtering and signature-based quotienting.

The raw interface-state space is

$$\mathcal{U}_i(P,\pi,\tau;\mathfrak{M}) = \prod_{v\in B_i^{\pi,\tau}} D_v, \tag{2.164}$$

the set of all assignments to the minimum active reconstruction interface $B_i^{\pi,\tau}$ introduced in Chapter 2. By the Interface Bound (Lemma 2.3) of Chapter 2,

$$|\mathcal{U}_i(P,\pi,\tau;\mathfrak{M})| = q(P)^{|B_i^{\pi,\tau}|} \le q(P)^{\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1}. \tag{2.165}$$

When τ and $\mathfrak{M}$ are fixed, the shorter notation $\mathcal{U}_i(P,\pi)$ may be used.

The reachable state space

$$\mathcal{U}_i^{\mathrm{reach}}(P,\pi) \subseteq \mathcal{U}_i(P,\pi)$$

removes assignments that cannot occur under the declared reconstruction rules.

The normalized future-signature map

$$\widehat{\sigma}_{P,i}^{\tau} : \mathcal{U}_i^{\mathrm{reach}}(P,\pi) \longrightarrow \widehat{\mathcal{Y}}_{P,i}^{\tau}$$

assigns to every reachable state its canonical task-relevant future description.

The semantic state space is then the quotient

$$\mathcal{S}_i^{\tau}(P,\pi) = \mathcal{U}_i^{\mathrm{reach}}(P,\pi)/\equiv_{P,i}^{\tau}. \tag{2.166}$$

Equivalently, the quotient is represented by the signature image:

$$\mathcal{S}_i^{\tau}(P,\pi) \cong \widehat{\sigma}_{P,i}^{\tau}\left(\mathcal{U}_i^{\mathrm{reach}}(P,\pi)\right). \tag{2.167}$$

There is a surjective quotient map

$$q_i^{\tau} : \mathcal{U}_i^{\mathrm{reach}}(P,\pi) \twoheadrightarrow \mathcal{S}_i^{\tau}(P,\pi). \tag{2.168}$$

Consequently,

$$|\mathcal{S}_i^{\tau}(P,\pi)| \le \left|\mathcal{U}_i^{\mathrm{reach}}(P,\pi)\right| \le |\mathcal{U}_i(P,\pi)|. \tag{2.169}$$

The complete semantic construction is therefore

$$\mathcal{U}_i(P,\pi) \supseteq \mathcal{U}_i^{\mathrm{reach}}(P,\pi) \xrightarrow{\widehat{\sigma}_{P,i}^{\tau}} \widehat{\sigma}_{P,i}^{\tau}\left(\mathcal{U}_i^{\mathrm{reach}}(P,\pi)\right) \cong \mathcal{S}_i^{\tau}(P,\pi). \tag{2.170}$$

At the global level, the resulting structural chain remains

$$\omega_{\mathrm{rec}}(P,\pi) \longrightarrow S_{\mathrm{CPR}}(P,\pi;\tau). \tag{2.171}$$

The arrow denotes an upper-bound control relationship, established by the Width–State Bound, and not a causal or algorithmic dependency.

The first quantity controls the maximum active-interface size.

The second counts the maximum number of normalized task-signature classes encountered during reconstruction.

No equality of future-continuation sets is required for this cardinality bound.

Because $S_{\mathrm{CPR}}(P, \pi; \tau)$ is a maximum over stages while the total stage-indexed semantic-state count $N_{\mathrm{sem,state}}(P, \pi, \tau; \mathfrak{M})$ of Chapter 5 is a sum over the same stages, it follows immediately that

$$S_{\mathrm{CPR}}(P, \pi; \tau) \leq N_{\mathrm{sem,state}}(P, \pi, \tau; \mathfrak{M}). \tag{2.172}$$

The following chapter extends this structural chain from semantic-state growth to the complete computational cost of Controlled Partial Reconstruction.

3 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 this chapter, let P be a finite problem instance, let τ be the declared computational task, let $\mathfrak{M}$ be the fixed CPR reconstruction model, and let

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

be an admissible reconstruction ordering.

All runtime quantities are therefore relative to

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

When no ambiguity can arise, the dependence on τ and $\mathfrak{M}$ may be suppressed.

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.

3.1 Complete Cost Model

Definition 3.1 (Complete CPR Operation Count)**.** The complete CPR operation count is

$$\begin{aligned} N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) = {} & N_{\mathrm{Build}}(P, \pi, \tau; \mathfrak{M}) \\ & + N_{\mathrm{Reduce}}(P, \pi, \tau; \mathfrak{M}) \\ & + N_{\mathrm{Transition}}(P, \pi, \tau; \mathfrak{M}) \\ & + N_{\mathrm{Reconstruct}}(P, \pi, \tau; \mathfrak{M}) \\ & + N_{\mathrm{Verify}}(P, \pi, \tau; \mathfrak{M}) \\ & + N_{\mathrm{Output}}(P, \pi, \tau; \mathfrak{M}). \end{aligned} \tag{3.1}$$

The six terms are defined as disjoint accounting categories.

Every elementary operation of the declared CPR procedure must be assigned to exactly one of these categories.

No elementary operation may be counted simultaneously in two different cost components.

In particular,

- construction of a semantic class belongs either to N_{Reduce} or to $N_{\text{Transition}}$, according to the declared accounting rule, but not to both;
- writing an output object belongs to N_{Output}, not again to $N_{\text{Reconstruct}}$;
- verification steps belong to N_{Verify}, unless correctness follows directly from a proved construction and no separate verification operation is executed.

The accounting convention must be fixed before any count-level comparison is performed.

3.2 Build Cost

Definition 3.2 (Build Cost)**.** The build cost $N_{\text{Build}}(P, \pi, \tau; \mathfrak{M})$ is the number of elementary operations required to construct the initial CPR representation before semantic-state processing begins.

Depending on the declared model, the build cost may include

- parsing and validating the encoded problem instance;
- constructing the finite component representation $V(P)$;
- constructing the local domains D_v;
- constructing or receiving the admissible ordering π;
- evaluating the admissibility predicate $\text{Adm}_{P,\tau}(\pi)$;
- initializing reachable-prefix representations;
- constructing auxiliary indices, incidence structures, or constraint tables;
- and allocating the data structures required by the reconstruction process.

The build cost does not include later semantic quotienting or stage-to-stage state 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.

3.3 Reduction Cost

Definition 3.3 (Reduction Cost)**.** The reduction cost $N_{\text{Reduce}}(P, \pi, \tau; \mathfrak{M})$ is the number of elementary operations required to construct the task-sufficient structural reductions used by the CPR model.

This cost may include

- determining task-sufficient candidate interfaces;
- testing or certifying interface sufficiency according to the declared model;
- selecting cardinality-minimal interfaces when this is performed algorithmically;
- computing the values $b_i^{\pi,\tau}(P)$;
- projecting reachable prefix states onto active interfaces;
- constructing induced interface signatures;
- forming semantic equivalence classes;
- canonicalizing semantic-state representatives;
- and constructing static quotient data used in later transitions.

The reduction cost must include the work required to construct the semantic quotient.

The semantic quotient construction includes the computation of the equivalence relation induced by

$$u \equiv_{P,i}^{\tau} v$$

and the resulting quotient representation

$$\mathcal{S}_i^{\tau}(P, \pi).$$

The existence of a small quotient does not imply that the quotient is inexpensive to compute.

If a semantic signature is precomputed once and reused during all stages, 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.

The distinction between reduction and transition cost is determined by the temporal role of the operation. Operations that construct semantic objects before reconstruction starts, including the computation of normalized signatures, equivalence classes, and quotient representatives, belong to N_{Reduce}. Operations that evaluate, update, or traverse already constructed semantic states during the reconstruction process belong to $N_{\mathrm{Transition}}$. Therefore, a precomputed semantic quotient is counted once in N_{Reduce}, whereas repeated semantic comparisons or successor-state computations performed during execution are counted in $N_{\mathrm{Transition}}$. No operation contributing to the same semantic decision may be counted in both categories.

In the notation of the preceding chapters, this cost accounts for the construction of the task-sufficient active interface $B_i^{\pi,\tau}$ of Chapter 2, the normalized future signature $\widehat{\sigma}_{P,i}^{\tau}$ of Chapter 3, and the induced semantic quotient $\mathcal{S}_i^{\tau}(P,\pi)$ of Chapter 3.

The reduction cost is exactly the cost of realizing, at every reconstruction stage i, the construction

$$B_i^{\pi,\tau} \longrightarrow \widehat{\sigma}_{P,i}^{\tau} \longrightarrow \mathcal{S}_i^{\tau}(P,\pi). \tag{3.2}$$

3.4 Reconstruction Stages and Branching

The reconstruction ordering processes the component set $V(P)$.

For the component-by-component reconstruction model introduced in Chapter 2, the number of nonterminal transition stages is

$$m_{\mathrm{stage}}(P,\pi) = |V(P)|. \tag{3.3}$$

The symbol m_{stage} is used instead of $b(P,\pi)$ in order to avoid conflict with the minimum interface size $b_i^{\pi,\tau}(P)$.

For a reachable interface state

$$u \in U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}),$$

let $\widehat{\Lambda}_i^{\pi,\tau}(u)$ denote the well-defined admissible next-value set introduced in Chapter 2.

Define the stage-wise branching number by

$$\lambda_i(P,\pi,\tau;\mathfrak{M}) = \begin{cases} \max\limits_{u \in U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})} \left|\widehat{\Lambda}_i^{\pi,\tau}(u)\right|, & U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}) \neq \varnothing, \\ 0, & U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}) = \varnothing. \end{cases} \tag{3.4}$$

The maximum branching number is defined by

$$\lambda_{\max}(P,\pi,\tau;\mathfrak{M}) = \begin{cases} \max\limits_{0 \le i < m_{\mathrm{stage}}(P,\pi)} \lambda_i(P,\pi,\tau;\mathfrak{M}), & m_{\mathrm{stage}}(P,\pi) > 0, \\ 0, & m_{\mathrm{stage}}(P,\pi) = 0. \end{cases} \tag{3.5}$$

Thus, both branching quantities remain well defined when a reachable state set is empty or when the component representation itself is empty.

If every stage is deterministic, then

$$\lambda_{\max}(P, \pi, \tau; \mathfrak{M}) \leq 1. \tag{3.6}$$

3.5 Semantic Transition Cost

Once the semantic quotient has been constructed, the subsequent CPR execution operates on semantic states rather than on raw interface assignments.

A transition is an admissible move from one semantic state at stage i to one semantic state at stage $i+1$, that is, from an element of $\mathcal{S}_i^\tau(P, \pi)$ to an element of $\mathcal{S}_{i+1}^\tau(P, \pi)$ (Chapter 3).

The quotient transition $\overline{\delta}_i^{\pi,\tau}$ was proved well defined in Chapter 3, under the successor-congruence requirement on the normalized future-signature family, and realizes exactly this stage-to-stage map

$$\mathcal{S}_i^\tau(P, \pi) \longrightarrow \mathcal{S}_{i+1}^\tau(P, \pi). \tag{3.7}$$

Definition 3.4 (Per-Edge Transition Cost)**.** Let $c_{\text{edge}}(P, \pi, \tau; \mathfrak{M})$ be an upper bound on the elementary-operation cost of processing one admissible semantic transition edge.

This cost may include

- generating one successor candidate;
- testing local admissibility;
- evaluating or retrieving its induced signature;
- locating or constructing the successor semantic class;
- merging duplicate successor representatives;
- and updating transition data.

Whether the construction of a successor's induced signature and semantic class is charged to $N_{\text{Transition}}$ or has already been charged to N_{Reduce} is governed by the same accounting convention declared in Section 3.3: if the semantic quotient is precomputed once for every stage, these two bullet items reduce to a lookup charged here, and their construction cost has already been counted in N_{Reduce}; if classes are instead constructed on demand while processing transitions, their construction cost is charged here and must not also be counted in N_{Reduce}. Exactly one of the two conventions applies for a given declared model, so no operation is counted twice.

Definition 3.5 (Local State-Processing Cost)**.** Let $c_{\text{local}}(P, \pi, \tau; \mathfrak{M})$ be an upper bound on the state-local cost incurred once for one semantic state during one reconstruction stage, independently of the number of outgoing transition edges.

This cost may include state initialization, local bookkeeping, retrieval of cached data, state-level canonicalization, or other work performed once per state.

Definition 3.6 (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

$$\begin{aligned} c_{\text{state}}(P, \pi, \tau; \mathfrak{M}) \leq \lambda_{\max}(P, \pi, \tau; \mathfrak{M})\, c_{\text{edge}}(P, \pi, \tau; \mathfrak{M}) \\ + c_{\text{local}}(P, \pi, \tau; \mathfrak{M}). \end{aligned} \tag{3.8}$$

The manuscript may use either the per-edge convention or the per-state convention, but the selected convention must be stated explicitly.

Definition 3.7 (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,

$$N_{\text{Transition}}(P,\pi,\tau;\mathfrak{M}) \le \sum_{i=0}^{m_{\text{stage}}(P,\pi)-1} |\mathcal{S}_i^{\tau}(P,\pi;\mathfrak{M})| \cdot c_{\text{state}}(P,\pi,\tau;\mathfrak{M}). \tag{3.9}$$

If $m_{\text{stage}}(P,\pi) = 0$, the sum is interpreted as the empty sum and therefore has value 0.
Consequently,

$$N_{\text{Transition}}(P,\pi,\tau;\mathfrak{M}) \le m_{\text{stage}}(P,\pi)\, S_{\text{CPR}}(P,\pi,\tau;\mathfrak{M}) \cdot c_{\text{state}}(P,\pi,\tau;\mathfrak{M}). \tag{3.10}$$

Using the per-edge convention, including the local cost incurred once per semantic state,

$$N_{\text{Transition}}(P,\pi,\tau;\mathfrak{M}) \le m_{\text{stage}}(P,\pi)\, S_{\text{CPR}}(P,\pi,\tau;\mathfrak{M}) \cdot \Big[\lambda_{\max}(P,\pi,\tau;\mathfrak{M})\, c_{\text{edge}}(P,\pi,\tau;\mathfrak{M}) + c_{\text{local}}(P,\pi,\tau;\mathfrak{M})\Big]. \tag{3.11}$$

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

3.6 Width-Based Transition Bound

By the Width–State Bound established in Chapter 3,

$$S_{\text{CPR}}(P,\pi,\tau;\mathfrak{M}) \le q(P)^{\widehat{\omega}_{\text{rec}}(P,\pi,\tau;\mathfrak{M})}. \tag{3.12}$$

Consequently, the per-state transition bound becomes

$$N_{\text{Transition}}(P,\pi,\tau;\mathfrak{M}) \le m_{\text{stage}}(P,\pi)\, q(P)^{\widehat{\omega}_{\text{rec}}(P,\pi,\tau;\mathfrak{M})} \cdot c_{\text{state}}(P,\pi,\tau;\mathfrak{M}). \tag{3.13}$$

Using the normalized CPR-Width,

$$N_{\text{Transition}}(P,\pi,\tau;\mathfrak{M}) \le m_{\text{stage}}(P,\pi)\, q(P)^{\omega_{\text{rec}}(P,\pi,\tau;\mathfrak{M})+1} \cdot c_{\text{state}}(P,\pi,\tau;\mathfrak{M}). \tag{3.14}$$

Under the per-edge convention,

$$N_{\text{Transition}}(P,\pi,\tau;\mathfrak{M}) \le m_{\text{stage}}(P,\pi)\, q(P)^{\widehat{\omega}_{\text{rec}}(P,\pi,\tau;\mathfrak{M})} \cdot \Big[\lambda_{\max}(P,\pi,\tau;\mathfrak{M})\, c_{\text{edge}}(P,\pi,\tau;\mathfrak{M}) + c_{\text{local}}(P,\pi,\tau;\mathfrak{M})\Big]. \tag{3.15}$$

Equivalently, using the normalized width,

$$N_{\text{Transition}}(P,\pi,\tau;\mathfrak{M}) \le m_{\text{stage}}(P,\pi)\, q(P)^{\omega_{\text{rec}}(P,\pi,\tau;\mathfrak{M})+1} \cdot \Big[\lambda_{\max}(P,\pi,\tau;\mathfrak{M})\, c_{\text{edge}}(P,\pi,\tau;\mathfrak{M}) + c_{\text{local}}(P,\pi,\tau;\mathfrak{M})\Big]. \tag{3.16}$$

The bound using $\widehat{\omega}_{\text{rec}}$ is the sharper operational form.

The normalized form is retained for consistency with the principal CPR-Width notation.

3.7 Reconstruction Cost

Definition 3.8 (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, reconstruction may require only an acceptance value or one witness.

For counting, it may require the propagation and recovery of exact multiplicities.

For enumeration, it may require the explicit recovery of every required solution.

For optimization, it may require an optimum value, an optimal witness, all optimal witnesses, or another declared certificate.

The reconstruction cost includes operations used to trace predecessor information, expand semantic representatives, recover component values, or assemble complete outputs.

It does not include the physical writing of the final outputs; that work belongs to N_{Output}.

In the notation of the preceding chapters, reconstruction begins at the terminal semantic quotient $\mathcal{S}^{\tau}_{m_{\mathrm{stage}}(P,\pi)}(P, \pi)$ reached after the last transition stage, and recovers from it the declared output required by τ; materializing that output is accounted separately by N_{Output}.

3.8 Verification Cost

Definition 3.9 (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.

Verification may be omitted only when correctness follows directly from the construction and this implication has been proved.

A statement that an output is valid is not sufficient to eliminate the verification term unless no separate verification procedure is actually executed.

3.9 Output Cost

Definition 3.10 (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 an enumeration task, 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.

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

$$N_{\mathrm{Output}}(P,\pi,\tau;\mathfrak{M}) \geq c_{\mathrm{out}}\, L_{\mathrm{Output}}(P,\pi,\tau;\mathfrak{M}). \tag{3.17}$$

No runtime statement may ignore this unavoidable output-size lower bound.

3.10 Post-Transition Cost

For compact notation, define the post-transition cost by

$$\begin{aligned} R_{\mathrm{post}}(P,\pi,\tau;\mathfrak{M}) = {} & N_{\mathrm{Reconstruct}}(P,\pi,\tau;\mathfrak{M}) \\ & + N_{\mathrm{Verify}}(P,\pi,\tau;\mathfrak{M}) \\ & + N_{\mathrm{Output}}(P,\pi,\tau;\mathfrak{M}). \end{aligned} \tag{3.18}$$

The complete cost model may then be written as

$$\begin{aligned} N_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) = {} & N_{\mathrm{Build}}(P,\pi,\tau;\mathfrak{M}) \\ & + N_{\mathrm{Reduce}}(P,\pi,\tau;\mathfrak{M}) \\ & + N_{\mathrm{Transition}}(P,\pi,\tau;\mathfrak{M}) \\ & + R_{\mathrm{post}}(P,\pi,\tau;\mathfrak{M}). \end{aligned} \tag{3.19}$$

This is only a notational abbreviation.

The six-component decomposition in Definition 3.1 remains the primary complete accounting model.

3.11 Complete Width-Based Runtime Bound

Theorem 3.1 (Complete Width-Based Runtime Bound)**.** *Under the per-state transition-cost convention,*

$$\begin{aligned} N_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \leq {} & N_{\mathrm{Build}}(P,\pi,\tau;\mathfrak{M}) \\ & + N_{\mathrm{Reduce}}(P,\pi,\tau;\mathfrak{M}) \\ & + m_{\mathrm{stage}}(P,\pi)\, q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} \\ & \quad \cdot c_{\mathrm{state}}(P,\pi,\tau;\mathfrak{M}) \\ & + R_{\mathrm{post}}(P,\pi,\tau;\mathfrak{M}). \end{aligned} \tag{3.20}$$

Consequently,

$$\begin{aligned} N_{\mathrm{CPR}}(P,\pi,\tau;\mathfrak{M}) \leq {} & N_{\mathrm{Build}}(P,\pi,\tau;\mathfrak{M}) \\ & + N_{\mathrm{Reduce}}(P,\pi,\tau;\mathfrak{M}) \\ & + m_{\mathrm{stage}}(P,\pi)\, q(P)^{\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})+1} \\ & \quad \cdot c_{\mathrm{state}}(P,\pi,\tau;\mathfrak{M}) \\ & + R_{\mathrm{post}}(P,\pi,\tau;\mathfrak{M}). \end{aligned} \tag{3.21}$$

Proof. By the complete six-component decomposition in Equation (3.1) and the definition of the post-transition cost in Equation (3.18), one obtains the abbreviated complete cost model

$$N_{\mathrm{CPR}} = N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} + R_{\mathrm{post}}. \tag{3.22}$$

Equivalently, this is the identity stated in Equation (3.19).

The per-state transition bound gives

$$N_{\text{Transition}} \leq m_{\text{stage}}\, S_{\text{CPR}}\, c_{\text{state}}. \tag{3.23}$$

By the Width–State Bound established in Chapter 3,

$$S_{\text{CPR}} \leq q(P)^{\widehat{\omega}_{\text{rec}}}. \tag{3.24}$$

Substitution yields

$$N_{\text{Transition}} \leq m_{\text{stage}}\, q(P)^{\widehat{\omega}_{\text{rec}}}\, c_{\text{state}}. \tag{3.25}$$

Substituting this inequality into Equation (3.19) proves Equation (3.20).
Chapter 2 established

$$q(P) \geq 1 \tag{3.26}$$

and

$$\widehat{\omega}_{\text{rec}} \leq \omega_{\text{rec}} + 1. \tag{3.27}$$

Since the function $x \mapsto q(P)^x$ is nondecreasing for $q(P) \geq 1$, it follows that

$$q(P)^{\widehat{\omega}_{\text{rec}}} \leq q(P)^{\omega_{\text{rec}}+1}. \tag{3.28}$$

Therefore, the normalized bound in Equation (3.21) follows.

□

Theorem 3.2 (Complete Edge-Based Runtime Bound)**.** *Under the per-edge transition-cost convention,*

$$\begin{aligned} N_{\text{CPR}}(P,\pi,\tau;\mathfrak{M}) \leq\ & N_{\text{Build}}(P,\pi,\tau;\mathfrak{M}) \\ & + N_{\text{Reduce}}(P,\pi,\tau;\mathfrak{M}) \\ & + m_{\text{stage}}(P,\pi)\, q(P)^{\widehat{\omega}_{\text{rec}}(P,\pi,\tau;\mathfrak{M})} \\ & \quad \cdot \Big[\lambda_{\max}(P,\pi,\tau;\mathfrak{M})\, c_{\text{edge}}(P,\pi,\tau;\mathfrak{M}) \\ & \qquad + c_{\text{local}}(P,\pi,\tau;\mathfrak{M})\Big] \\ & + R_{\text{post}}(P,\pi,\tau;\mathfrak{M}). \end{aligned} \tag{3.29}$$

Proof. At every reconstruction stage, at most

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

semantic states are processed.
For each semantic state, the state-local work costs at most $c_{\text{local}}(P,\pi,\tau;\mathfrak{M})$.
Each state has at most $\lambda_{\max}(P,\pi,\tau;\mathfrak{M})$ admissible outgoing transitions, and every transition edge costs at most $c_{\text{edge}}(P,\pi,\tau;\mathfrak{M})$.
Therefore,

$$\begin{aligned} N_{\text{Transition}}(P,\pi,\tau;\mathfrak{M}) \leq\ & m_{\text{stage}}(P,\pi)\, S_{\text{CPR}}(P,\pi,\tau;\mathfrak{M}) \\ & \cdot \Big[\lambda_{\max}(P,\pi,\tau;\mathfrak{M})\, c_{\text{edge}}(P,\pi,\tau;\mathfrak{M}) \\ & \quad + c_{\text{local}}(P,\pi,\tau;\mathfrak{M})\Big]. \end{aligned} \tag{3.30}$$

Using

$$S_{\text{CPR}}(P,\pi,\tau;\mathfrak{M}) \leq q(P)^{\widehat{\omega}_{\text{rec}}(P,\pi,\tau;\mathfrak{M})} \tag{3.31}$$

and substituting the resulting transition bound into Equation (3.19) proves Equation (3.29).

□

3.12 Polynomial Tractability 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.

Corollary 3.1 (Constant-Domain Logarithmic-Width Criterion)**.** *Assume that there exists a constant $q_0 > 1$ such that*

$$1 \le q(P) \le q_0. \tag{3.32}$$

The lower bound $q(P) \ge 1$ includes deterministic domains, where the state-space contribution may collapse to a constant factor. The upper bound q_0 controls the maximal local domain size.
Assume further that

$$\widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = O(\log \ell(P)). \tag{3.33}$$

Then there exists a constant $c > 0$ such that

$$q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} \le \ell(P)^c \tag{3.34}$$

for all sufficiently large inputs.
Equivalently,

$$q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} = \ell(P)^{O(1)}. \tag{3.35}$$

Proof. Since $q(P) \le q_0$ and

$$\widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = O(\log \ell(P)),$$

there exists a constant $C > 0$ such that

$$\widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) \le C \log \ell(P)$$

for all sufficiently large instances.
Therefore,

$$q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})} \le q_0^{C \log \ell(P)} \tag{3.36}$$

$$= \ell(P)^{C \log q_0}. \tag{3.37}$$

Since q_0 and C are constants, the exponent $C \log q_0$ is constant.
Hence, the right-hand side is polynomially bounded in $\ell(P)$.

□

The corollary controls only the width-dependent state factor.
A complete polynomial-time result additionally requires polynomials

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

such that

$$N_j(P, \pi, \tau; \mathfrak{M}) \le p_j(\ell(P)) \tag{3.38}$$

for every cost component

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

In particular, one must also control $m_{\text{stage}}(P, \pi)$, $\lambda_{\max}(P, \pi, \tau; \mathfrak{M})$, and either $c_{\text{state}}(P, \pi, \tau; \mathfrak{M})$ or $c_{\text{edge}}(P, \pi, \tau; \mathfrak{M})$ together with $c_{\text{local}}(P, \pi, \tau; \mathfrak{M})$, according to the selected transition-cost convention.

3.13 Sufficient Complete Polynomial-Cost Condition

Theorem 3.3 (Sufficient Complete Polynomial-Cost Condition)**.** *Assume that there exist polynomials*

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

such that

$$N_j(P, \pi, \tau; \mathfrak{M}) \leq p_j(\ell(P)) \tag{3.39}$$

for every one of the six cost components.
Then there exists a polynomial p_{CPR} *such that*

$$N_{\text{CPR}}(P, \pi, \tau; \mathfrak{M}) \leq p_{\text{CPR}}(\ell(P)). \tag{3.40}$$

Proof. By Equation (3.1),

$$\begin{aligned} N_{\text{CPR}} = {} & N_{\text{Build}} + N_{\text{Reduce}} + N_{\text{Transition}} \\ & + N_{\text{Reconstruct}} + N_{\text{Verify}} + N_{\text{Output}}. \end{aligned} \tag{3.41}$$

Using the assumed bounds gives

$$\begin{aligned} N_{\text{CPR}} \leq {} & p_{\text{Build}}(\ell(P)) + p_{\text{Reduce}}(\ell(P)) \\ & + p_{\text{Transition}}(\ell(P)) + p_{\text{Reconstruct}}(\ell(P)) \\ & + p_{\text{Verify}}(\ell(P)) + p_{\text{Output}}(\ell(P)). \end{aligned} \tag{3.42}$$

A finite sum of polynomials is a polynomial.
Therefore, a polynomial p_{CPR} satisfying Equation (3.40) exists.

□

3.14 Classical Baseline and Count-Level Gain

Let

$$N_{\text{Classic}}(P, \tau; \mathcal{A}_{\text{Classic}})$$

denote the complete elementary-operation count of a declared reference algorithm $\mathcal{A}_{\text{Classic}}$ 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 that

$$N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) > 0. \tag{3.43}$$

Definition 3.11 (Count-Level Gain)**.** The count-level gain is an elementary-operation-count metric rather than a measured execution-time metric. The count-level gain of the CPR model relative to the declared classical baseline is

$$G_{\mathrm{count}}(P, \pi, \tau; \mathfrak{M}, \mathcal{A}_{\mathrm{Classic}}) = \frac{N_{\mathrm{Classic}}(P, \tau; \mathcal{A}_{\mathrm{Classic}})}{N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M})}. \tag{3.44}$$

A favorable count-level regime exists when

$$N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) < N_{\mathrm{Classic}}(P, \tau; \mathcal{A}_{\mathrm{Classic}}), \tag{3.45}$$

equivalently,

$$G_{\mathrm{count}}(P, \pi, \tau; \mathfrak{M}, \mathcal{A}_{\mathrm{Classic}}) > 1. \tag{3.46}$$

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 it is not the only admissible classical baseline.

3.15 Measured Wall-Clock Gain

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

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

denote the measured wall-clock time of a declared CPR implementation $\mathcal{I}_{\mathrm{CPR}}$.

Assume that

$$T_{\mathrm{CPR}}(P, \pi, \tau; \mathcal{I}_{\mathrm{CPR}}) > 0. \tag{3.47}$$

Definition 3.12 (Measured Wall-Clock Gain)**.** The measured wall-clock gain is

$$G_{\mathrm{time}}(P, \pi, \tau; \mathcal{I}_{\mathrm{CPR}}, \mathcal{I}_{\mathrm{Classic}}) = \frac{T_{\mathrm{Classic}}(P, \tau; \mathcal{I}_{\mathrm{Classic}})}{T_{\mathrm{CPR}}(P, \pi, \tau; \mathcal{I}_{\mathrm{CPR}})}. \tag{3.48}$$

A count-level gain does not imply a wall-clock gain:

$$G_{\mathrm{count}} > 1 \not\Longrightarrow G_{\mathrm{time}} > 1. \tag{3.49}$$

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.

Therefore, operation-count claims and wall-clock claims must remain separate.

3.16 Structural Runtime Chain

The theory developed in Chapters 2–3 yields the central structural runtime chain

$$\widehat{\boldsymbol{\omega}}_{\mathbf{rec}}(\boldsymbol{P},\boldsymbol{\pi},\boldsymbol{\tau};\boldsymbol{\mathfrak{M}}) \longrightarrow \boldsymbol{S}_{\mathbf{CPR}}(\boldsymbol{P},\boldsymbol{\pi},\boldsymbol{\tau};\boldsymbol{\mathfrak{M}}) \longrightarrow \boldsymbol{N}_{\mathbf{CPR}}(\boldsymbol{P},\boldsymbol{\pi},\boldsymbol{\tau};\boldsymbol{\mathfrak{M}}). \tag{3.50}$$

Using the normalized width, this may also be written as

$$\boldsymbol{\omega}_{\mathbf{rec}}(\boldsymbol{P},\boldsymbol{\pi},\boldsymbol{\tau};\boldsymbol{\mathfrak{M}}) \longrightarrow \boldsymbol{S}_{\mathbf{CPR}}(\boldsymbol{P},\boldsymbol{\pi},\boldsymbol{\tau};\boldsymbol{\mathfrak{M}}) \longrightarrow \boldsymbol{N}_{\mathbf{CPR}}(\boldsymbol{P},\boldsymbol{\pi},\boldsymbol{\tau};\boldsymbol{\mathfrak{M}}). \tag{3.51}$$

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.

3.17 Runtime Non-Implications

Proposition 3.1 (Width Does Not Determine Runtime Alone)**.** *The inequality*

$$\omega_{\mathrm{rec}}(P,\pi_1,\tau;\mathfrak{M}) < \omega_{\mathrm{rec}}(P,\pi_2,\tau;\mathfrak{M}) \tag{3.52}$$

does not by itself imply

$$N_{\mathrm{CPR}}(P,\pi_1,\tau;\mathfrak{M}) < N_{\mathrm{CPR}}(P,\pi_2,\tau;\mathfrak{M}). \tag{3.53}$$

It also does not imply

$$T_{\mathrm{CPR}}(P,\pi_1,\tau;\mathcal{I}_{\mathrm{CPR}}) < T_{\mathrm{CPR}}(P,\pi_2,\tau;\mathcal{I}_{\mathrm{CPR}}), \tag{3.54}$$

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;
- more costly semantic signatures;
- larger branching;
- more expensive quotient transitions;
- greater reconstruction overhead;
- greater verification cost;
- or less favorable memory behavior.

Therefore, width alone does not determine either the complete operation count or the measured wall-clock time.

□

Proposition 3.2 (Semantic-State Count Does Not Determine Runtime Alone)**.** *The inequality*

$$S_{\mathrm{CPR}}(P, \pi_1, \tau; \mathfrak{M}) < S_{\mathrm{CPR}}(P, \pi_2, \tau; \mathfrak{M}) \tag{3.55}$$

does not by itself imply

$$N_{\mathrm{CPR}}(P, \pi_1, \tau; \mathfrak{M}) < N_{\mathrm{CPR}}(P, \pi_2, \tau; \mathfrak{M}). \tag{3.56}$$

Proof. A smaller semantic quotient may be more expensive to construct.

It may also require more expensive canonicalization, equivalence tests, or reconstruction steps.

The complete operation count depends on all six cost components and not only on the maximum semantic-state count.

□

3.18 Conclusion

Controlled Partial Reconstruction Width is a structural control parameter.

It does not determine runtime by itself.

The complete CPR operation count is

$$\begin{aligned} N_{\mathrm{CPR}} = {} & N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} \\ & + N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}. \end{aligned} \tag{3.57}$$

The six cost components are disjoint accounting categories.

Every elementary operation must be assigned to exactly one category.

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,

$$N_{\mathrm{Transition}} \leq m_{\mathrm{stage}}\, S_{\mathrm{CPR}}\, \left(\lambda_{\max} c_{\mathrm{edge}} + c_{\mathrm{local}}\right). \tag{3.58}$$

Under the per-state convention,

$$N_{\mathrm{Transition}} \leq m_{\mathrm{stage}}\, S_{\mathrm{CPR}}\, c_{\mathrm{state}}. \tag{3.59}$$

The width-based bound controls the semantic-state contribution through

$$S_{\mathrm{CPR}} \leq q(P)^{\widehat{\omega}_{\mathrm{rec}}} \leq q(P)^{\omega_{\mathrm{rec}}+1}. \tag{3.60}$$

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

$$\boldsymbol{\omega_{\mathrm{rec}}} \longrightarrow \boldsymbol{S_{\mathrm{CPR}}} \longrightarrow \boldsymbol{N_{\mathrm{CPR}}}. \tag{3.61}$$

A refined runtime-accounting chain is

$$\boldsymbol{\omega_{\mathrm{rec}}} \longrightarrow \boldsymbol{S_{\mathrm{CPR}}} \longrightarrow \boldsymbol{N_{\mathrm{Transition}}} \longrightarrow \boldsymbol{N_{\mathrm{CPR}}}. \tag{3.62}$$

The refined chain makes explicit that the semantic-state count enters the complete operation count primarily through transition processing.

It does not imply that the remaining cost components are determined by the transition cost or by the reconstruction width.

The semantic-state count S_{CPR} and the binary semantic-state dimension D_{CPR} of Chapter 3 provide structural bounds on the number and representation size of semantic states. They do not directly equal the total operation count N_{CPR}. The complete runtime analysis additionally requires the transition, verification, reconstruction, and output costs defined in this chapter.

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

Chapter Summary

The complete CPR pipeline decomposes into six accounted components: construction of the declared representation (N_{Build}), formation of active interfaces and reduced representations (N_{Reduce}), semantic-transition processing ($N_{\mathrm{Transition}}$), reconstruction of task-relevant output ($N_{\mathrm{Reconstruct}}$), correctness verification (N_{Verify}), and output materialization (N_{Output}). Their sum is the complete operation count $N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M})$. The width-based and edge-based runtime bounds developed above express $N_{\mathrm{Transition}}$ in terms of the semantic-state count S_{CPR}, which is itself controlled by the reconstruction width $\omega_{\mathrm{rec}}(P, \pi)$ via the Width–State Bound. The classical baseline count N_{Classic} and the resulting count-level gain G_{count} are declared separately from measured wall-clock performance: neither reconstruction width, nor semantic-state control, nor count-level gain alone establishes a measured runtime advantage. The following chapter formalizes the boundary conditions BC1–BC4 under which this operation-count model yields a uniform polynomial-time guarantee.

4 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 certify that Controlled Partial Reconstruction admits a polynomially bounded execution over an entire uniformly encoded problem family. This chapter formulates those conditions and establishes the resulting tractability criterion.

Throughout this chapter, all elementary-operation counts are interpreted in one fixed deterministic bit-cost computation model.

For definiteness, the model may be taken to be a deterministic multi-tape Turing machine.

Alternatively, 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 when 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

$$\ell(P_x) = |x| \tag{4.1}$$

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.

4.1 Overview

Polynomial-time tractability requires more than a small reconstruction width.

The reconstruction representation must be explicitly specified.

It must be 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.

Finally, every component of the complete runtime model introduced in Chapter 3 must be polynomially bounded.

These requirements are organized through four boundary conditions:

1. BC1: explicit uniform structural representation;
2. BC2: polynomial uniform constructibility;
3. BC3: task-relative soundness and completeness;
4. BC4: polynomial semantic-state and transition-structure control.

The boundary conditions are required components of the declared CPR tractability certificate.

They do not, when considered in isolation, replace the complete six-component runtime analysis.

4.2 Boundary Condition BC1

Definition 4.1 (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 $V(P_x)$;
- the local domains D_v;
- the admissibility predicate for reconstruction orderings;
- the representation of an admissible ordering when the ordering is supplied as part of the encoded instance, or one uniform procedure for constructing such an ordering;
- the reachable prefix-state spaces;
- the declared active reconstruction interfaces together with their task-sufficiency certification;
- 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.

The reconstruction-safe signature family $\mathrm{Sig}_{\tau,i}^{\pi_x}$ corresponds to the normalized future-signature mapping $\widehat{\sigma}_{P,i}^{\tau}$ defined in Chapter 3.

BC1 is a representation condition.

It requires the mathematical objects on which the CPR analysis is based to be explicitly declared.

It does not assert that these objects are inexpensive to construct.

It also does not assert that the resulting reconstruction process is sound, complete, or polynomially bounded.

4.3 Boundary Condition BC2

Definition 4.2 (BC2 – Polynomial Uniform Constructibility)**.** A uniformly encoded family $\mathcal{F} = \{P_x : x \in \{0,1\}^*\}$ satisfies BC2 for task τ and reconstruction model $\mathfrak{M}$ if there exists one deterministic preprocessing algorithm

$$\mathcal{A}_{\text{Preprocess}}$$

and one polynomial $p_{\text{BC2}} : \mathbb{N} \to \mathbb{N}$ such that, for every encoding x,

$$N_{\mathcal{A}_{\text{Preprocess}}}(x) \leq p_{\text{BC2}}(|x|), \tag{4.2}$$

where $N_{\mathcal{A}_{\text{Preprocess}}}(x)$ denotes the number of elementary operations executed by the preprocessing algorithm.

The algorithm constructs or reads all static 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, indexing, and constraint structures;
- and all other data assigned to N_{Build} or to the static portion of N_{Reduce}.

The ordering-supply procedure is part of $\mathcal{A}_{\text{Preprocess}}$.

It either reads an admissible ordering from the declared input representation or constructs one uniformly from x.

In both cases, the resulting ordering must satisfy

$$\pi_x = \mathcal{A}_\pi(x) \in \Pi_{\text{adm}}(P_x, \tau; \mathfrak{M}), \tag{4.3}$$

where $\mathcal{A}_\pi$ denotes the ordering-supply component of $\mathcal{A}_{\text{Preprocess}}$.

The operation count of $\mathcal{A}_\pi$ is included in N_{Build}.

It is not counted as an additional runtime component.

The quantity $N_{\mathcal{A}_{\text{Preprocess}}}$ is an auxiliary preprocessing count and does not introduce a seventh component into the complete runtime model.

Every elementary preprocessing operation must be assigned exactly once according to the disjoint accounting convention fixed in Chapter 3.

BC2 requires polynomial constructibility of every structural object that the declared model assigns to preprocessing.

The precise allocation between N_{Build} and N_{Reduce} must follow the disjoint accounting convention fixed in Chapter 3.

BC2 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.

4.4 Boundary Condition BC3

Definition 4.3 (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 solution-set representation is applicable.

Decision

For a decision task, let

$$\mathrm{Dec}_\tau(P_x) \in \{0, 1\}$$

denote the correct task-relative decision value, and define

$$\mathrm{CPRDecision}(P_x, \pi_x, \tau; \mathfrak{M}) \in \{0, 1\}.$$

Soundness and completeness require

$$\mathrm{CPRDecision}(P_x, \pi_x, \tau; \mathfrak{M}) = \mathrm{Dec}_\tau(P_x). \tag{4.4}$$

For an existential feasibility decision task, this specializes to

$$\mathrm{Sol}_\tau(P_x) \neq \varnothing \quad \Longleftrightarrow \quad \mathrm{CPRDecision}(P_x, \pi_x, \tau; \mathfrak{M}) = 1. \tag{4.5}$$

Thus, 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.

Enumeration

For an enumeration task, soundness and completeness require

$$\mathrm{Output}_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) = \mathrm{Sol}_\tau(P_x). \tag{4.6}$$

Equivalently, soundness requires

$$\mathrm{Output}_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) \subseteq \mathrm{Sol}_\tau(P_x), \tag{4.7}$$

and completeness requires

$$\mathrm{Sol}_\tau(P_x) \subseteq \mathrm{Output}_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}). \tag{4.8}$$

Counting

For an exact counting task, soundness and completeness require

$$\mathrm{CPRCount}(P_x, \pi_x, \tau; \mathfrak{M}) = \left|\mathrm{Sol}_\tau(P_x)\right|. \tag{4.9}$$

Optimization

For an optimization task, let $\mathrm{Opt}_\tau(P_x)$ denote the exact task-required optimum object.
Depending on the declared task, this object may be

- the optimum value;
- one optimal witness;
- all optimal witnesses;
- or another explicitly declared optimality certificate.

The output requirement must be fixed before the CPR model is evaluated, so that no weaker optimization objective may be substituted after the fact.

BC3 requires

$$\mathrm{CPROpt}(P_x, \pi_x, \tau; \mathfrak{M}) = \mathrm{Opt}_\tau(P_x). \tag{4.10}$$

No weaker output object may silently replace the declared task requirement.

Task-Evaluation Form

More generally, if the task is represented by the evaluation map Θ_τ, then BC3 requires exact preservation of the declared task value:

$$\Theta_\tau^{\mathrm{CPR}}(P_x, \pi_x; \mathfrak{M}) = \Theta_\tau(P_x). \tag{4.11}$$

The decision, enumeration, counting, and optimization conditions above are special cases of Equation (4.11).

BC3 is a correctness condition.

It does not by itself provide a polynomial runtime bound.

4.5 Boundary Condition BC4

For each encoded instance, define the total number of semantic-state occurrences by

$$N_{\mathrm{sem,state}}(P_x, \pi_x, \tau; \mathfrak{M}) = \sum_{i=0}^{m_{\mathrm{stage}}(P_x, \pi_x)} \left| S_i(P_x, \pi_x, \tau; \mathfrak{M}) \right|. \tag{4.12}$$

This quantity counts stage-indexed semantic states.

The same abstract state occurring at two different reconstruction stages is counted twice because 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 4.4 (BC4 – Polynomial Semantic-State and Transition-Structure Control)**.** A uniformly encoded family $\mathcal{F} = \{P_x : x \in \{0,1\}^*\}$ satisfies BC4 for task τ, reconstruction model $\mathfrak{M}$, and admissible orderings π_x if there exist polynomials

$$p_{\mathrm{SemanticVertex}}, \qquad p_{\mathrm{SemanticEdge}} : \mathbb{N} \longrightarrow \mathbb{N},$$

independent of x, such that for every encoded instance P_x,

$$N_{\mathrm{sem,state}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_{\mathrm{SemanticVertex}}(|x|) \tag{4.13}$$

and

$$E_{\mathrm{sem}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_{\mathrm{SemanticEdge}}(|x|). \tag{4.14}$$

The first inequality controls the total number of stage-indexed reachable semantic-state occurrences.

The second controls the complete number of reachable semantic transition edges.
Since

$$S_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) = \max_i |S_i(P_x, \pi_x, \tau; \mathfrak{M})|, \tag{4.15}$$

one also has

$$S_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq N_{\mathrm{sem,state}}(P_x, \pi_x, \tau; \mathfrak{M}). \tag{4.16}$$

Therefore, BC4 also implies polynomial control of the maximum stage-wise semantic-state count.

The polynomial cardinality of the semantic-state space does not imply that each state can be generated, compared, or encoded in polynomial time unless these operations are included in the corresponding runtime bounds.

A short binary description of one semantic state does not imply that only polynomially many semantic states occur.

Similarly, 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 certification condition.

It does not by itself imply that the processing of one transition edge is inexpensive.

The complete elementary-operation cost of generating, identifying, merging, storing, and processing the semantic transitions remains

$$N_{\mathrm{Transition}}(P_x, \pi_x, \tau; \mathfrak{M}),$$

which must be controlled separately by the complete runtime model.

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.

4.6 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 4.5 (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 called CPR-tractable if there exist

- one declared decision task τ;
- one uniform CPR reconstruction model $\mathfrak{M}$;
- one uniform deterministic preprocessing algorithm $\mathcal{A}_{\mathrm{Preprocess}}$, containing an ordering-supply component $\mathcal{A}_\pi$, such that

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

- and six fixed polynomial functions

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

where

$$J = \{\mathrm{Build}, \mathrm{Reduce}, \mathrm{Transition}, \mathrm{Reconstruct}, \mathrm{Verify}, \mathrm{Output}\};$$

such that BC1–BC4 hold uniformly and, for every encoding x,

$$N_j(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_j(|x|) \tag{4.17}$$

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

$$L_{\mathcal{F}} = \{x \in \{0,1\}^* : P_x \text{ is a yes-instance for the declared task } \tau\}. \tag{4.18}$$

Definition 4.6 (The Language Class $\mathbf{P}_{\text{CPR}}$)**.** Define

$$\mathbf{P}_{\text{CPR}} = \Big\{L \subseteq \{0,1\}^* : L = L_{\mathcal{F}} \text{ for some uniformly encoded CPR-tractable decision family } \mathcal{F}\Big\}. \tag{4.19}$$

Thus, membership in $\mathbf{P}_{\text{CPR}}$ 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 the 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.

The class is not intended as a replacement for standard complexity classes, but as a certified subclass induced by the CPR framework.

4.7 Uniform Tractability

Theorem 4.1 (Uniform CPR Tractability)**.** *Let*

$$\mathcal{F} = \{P_x : x \in \{0,1\}^*\}$$

be a uniformly encoded family of finite decision-problem instances, and let $L_{\mathcal{F}}$ be the associated decision language.

Assume that 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 $\mathcal{A}_\pi$, and producing*

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

- *and six fixed polynomial functions*

$$p_j : \mathbb{N} \longrightarrow \mathbb{N}, \qquad j \in J;$$

such that BC1, BC2, BC3, and BC4 hold uniformly and, for every encoding x,

$$N_j(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_j(|x|) \tag{4.20}$$

for every $j \in J$.

Then

$$L_{\mathcal{F}} \in \mathbf{P}. \tag{4.21}$$

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:

$$x \in L_{\mathcal{F}} \quad \Longleftrightarrow \quad \mathrm{CPRDecision}(P_x, \pi_x, \tau; \mathfrak{M}) = 1. \tag{4.22}$$

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 3,

$$\begin{aligned} N_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) = {} & N_{\mathrm{Build}}(P_x, \pi_x, \tau; \mathfrak{M}) \\ & + N_{\mathrm{Reduce}}(P_x, \pi_x, \tau; \mathfrak{M}) \\ & + N_{\mathrm{Transition}}(P_x, \pi_x, \tau; \mathfrak{M}) \\ & + N_{\mathrm{Reconstruct}}(P_x, \pi_x, \tau; \mathfrak{M}) \\ & + N_{\mathrm{Verify}}(P_x, \pi_x, \tau; \mathfrak{M}) \\ & + N_{\mathrm{Output}}(P_x, \pi_x, \tau; \mathfrak{M}). \end{aligned} \tag{4.23}$$

Using the six assumed uniform polynomial bounds gives

$$\begin{aligned} N_{\mathrm{CPR}}(P_x, \pi_x, \tau; \mathfrak{M}) \leq {} & p_{\mathrm{Build}}(|x|) + p_{\mathrm{Reduce}}(|x|) \\ & + p_{\mathrm{Transition}}(|x|) + p_{\mathrm{Reconstruct}}(|x|) \\ & + p_{\mathrm{Verify}}(|x|) + p_{\mathrm{Output}}(|x|). \end{aligned} \tag{4.24}$$

The right-hand side is a finite sum of fixed polynomial functions and is therefore polynomially bounded in $|x|$.
Because the operation counts are interpreted in the fixed deterministic bit-cost computation model declared at the beginning of this chapter, the complete CPR procedure is executable in deterministic polynomial time.
Consequently, membership of x in $L_{\mathcal{F}}$ is decided by one deterministic polynomial-time procedure.
Hence,

$$L_{\mathcal{F}} \in \mathbf{P}.$$

□

Corollary 4.1 (Certified CPR Languages Are Polynomial-Time Languages)**.** *The framework-dependent language class satisfies*

$$\mathbf{P}_{\mathrm{CPR}} \subseteq \mathbf{P}. \tag{4.25}$$

Proof. Let

$$L \in \mathbf{P}_{\mathrm{CPR}}.$$

By Definition 4.6, the language L is associated with a uniformly encoded CPR-tractable decision family.
The hypotheses of Theorem 4.1 therefore hold.

Hence,

$$L \in \mathbf{P}.$$

□

The inclusion in Equation (4.25) is a consequence of the certification requirements built into the definition of $\mathbf{P}_{\mathrm{CPR}}$.
It is not a complexity-class collapse statement.

4.8 Conditional Complexity Consequence

Corollary 4.2 (Conditional NP-Complete Consequence)**.** *Let*

$$L^* \subseteq \{0,1\}^*$$

be an NP-complete language.
If

$$L^* \in \mathbf{P}_{\mathrm{CPR}}, \tag{4.26}$$

then

$$\mathbf{P} = \mathbf{NP}. \tag{4.27}$$

Proof. By Corollary 4.1,

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

Therefore, the hypothesis

$$L^* \in \mathbf{P}_{\mathrm{CPR}}$$

implies

$$L^* \in \mathbf{P}.$$

Since L^* is NP-complete, every language in $\mathbf{NP}$ is polynomial-time many-one reducible to L^*.
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 4.1 (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.

Therefore, the preceding corollary is a conditional statement only.

It does not establish

$$\mathbf{P} = \mathbf{NP}.$$

5 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 this chapter, let P denote a finite problem instance, let τ be a fixed declared computational task, let $\mathfrak{M}$ be a fixed reconstruction model, and let

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

be an admissible reconstruction ordering.

The dependence on τ and $\mathfrak{M}$ may be suppressed when no ambiguity can arise.

A central distinction is required throughout this chapter.

For every problem-class realization, the visibly defined structural boundary is first treated as a candidate interface

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

The cardinality-minimal task-sufficient interface defined in Chapter 2 is denoted by

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

Unless task sufficiency and cardinality minimality have been proved, one may conclude only

$$b_i^{\pi,\tau}(P) \leq \left|\widetilde{B}_i^{\pi,\tau}\right|,$$

and therefore

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

Equality requires an additional minimality proof.

5.1 Boolean Satisfiability

Let

$$\Phi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$$

be a Boolean formula over the variable set

$$V(\Phi) = \{x_1, \ldots, x_n\}.$$

Let

$$\pi = (x_{\pi(1)}, \ldots, x_{\pi(n)})$$

be an admissible variable ordering.

After reconstruction stage i, the processed and unresolved variable sets are

$$V_i^\pi = \{x_{\pi(1)}, \ldots, x_{\pi(i)}\}$$

and

$$\overline{V_i^\pi} = V(\Phi) \setminus V_i^\pi.$$

A clause $C_j \in \Phi$ is called unresolved at stage i if it contains at least one variable in $\overline{V_i^\pi}$, and resolved at stage i otherwise.

Define the structural SAT candidate interface by

$$\widetilde{B}_i^{\Phi,\pi} = \{x \in V_i^\pi : \exists\, C_j \in \Phi \text{ unresolved at stage } i \text{ with } x \text{ occurring in } C_j\}. \tag{5.1}$$

This set contains processed variables whose assigned values may still influence residual clauses.

It is a natural structural candidate interface.

It is not automatically the unique or cardinality-minimal task-sufficient interface.

Let

$$r \in \mathcal{R}_i^\pi(\Phi)$$

be a reachable prefix assignment.

For the exact satisfying-extension realization, define

$$E_i^{\Phi,\pi}(r) = \left\{ s \in \prod_{x \in \overline{V_i^\pi}} \{0,1\} : r \cup s \models \Phi \right\}. \tag{5.2}$$

Equality of satisfying-extension sets may be used as one particular reconstruction-safe prefix signature:

$$\mathrm{Sig}_{\mathrm{SAT},i}^\pi(r) = E_i^{\Phi,\pi}(r). \tag{5.3}$$

This signature is stronger than the mere current decision value

$$\mathbf{1}\left[E_i^{\Phi,\pi}(r) \neq \varnothing\right].$$

Therefore, the decision signature $\mathbf{1}[E_i^{\Phi,\pi}(r) \neq \varnothing]$ induces a coarser semantic quotient than the extension-set signature $\mathrm{Sig}_{\mathrm{SAT},i}^\pi(r)$, consistent with the task-dependence of semantic equivalence established in Chapter 3.

Because the signature records the complete residual satisfying-extension set, it preserves the decision answer, admissible next values, and successor residual classes.

Hence, it satisfies the reconstruction-safety conditions S1–S3 of Chapter 2.

If the candidate interface $\widetilde{B}_i^{\Phi,\pi}$ determines this signature, then it is task-sufficient and

$$b_i^{\pi,\mathrm{SAT}}(\Phi) \leq \left|\widetilde{B}_i^{\Phi,\pi}\right|. \tag{5.4}$$

Consequently,

$$\widehat{\omega}_{\mathrm{rec}}(\Phi, \pi, \mathrm{SAT}; \mathfrak{M}) \leq \max_i \left|\widetilde{B}_i^{\Phi,\pi}\right|. \tag{5.5}$$

An equality claim requires a proof that no smaller processed-variable set determines the same reconstruction-safe signature.

This realization does not claim that unrestricted SAT has bounded CPR-Width or admits polynomial CPR reconstruction.

5.2 Constraint Satisfaction Problems

Let

$$P_{\mathrm{CSP}} = (V, D, \mathcal{C})$$

be a finite constraint-satisfaction instance, where

$$V = \{x_1, \ldots, x_n\},$$

$$D = (D_{x_1}, \ldots, D_{x_n}),$$

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 by

$$\widetilde{B}_i^{P_{\mathrm{CSP}},\pi} = \left\{ x \in V_i^{\pi} : \begin{array}{l} \exists C \in \mathcal{C} \text{ such that } x \in \mathrm{scope}(C), \\ \mathrm{scope}(C) \cap \overline{V_i^{\pi}} \neq \varnothing \end{array} \right\}. \tag{5.6}$$

This structural boundary contains 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

$$\mathrm{Sig}_{\mathrm{CSP},i}^{\pi,\tau}(r) = \left\{ s \in \prod_{x \in \overline{V_i^{\pi}}} D_x : r \cup s \text{ satisfies every constraint in } \mathcal{C} \right\}. \tag{5.7}$$

The exact form of the CSP signature depends on the declared task τ. For decision tasks, it may preserve exactly the feasibility extensions shown above; for counting or optimization tasks, it must instead preserve the corresponding task value together with the transition information required by S1–S3.

If the candidate interface $\widetilde{B}_i^{P_{\mathrm{CSP}},\pi}$ determines the declared safe signature, then

$$b_i^{\pi,\tau}(P_{\mathrm{CSP}}) \leq \left| \widetilde{B}_i^{P_{\mathrm{CSP}},\pi} \right|. \tag{5.8}$$

Thus,

$$\widehat{\omega}_{\mathrm{rec}}(P_{\mathrm{CSP}}, \pi, \tau; \mathfrak{M}) \leq \max_i \left| \widetilde{B}_i^{P_{\mathrm{CSP}},\pi} \right|. \tag{5.9}$$

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

5.3 Graph Coloring

Let

$$G = (V, E)$$

be a finite graph and let

$$K = \{1, \dots, k\}$$

be the available color set.

For an ordering

$$\pi = (v_{\pi(1)}, \dots, v_{\pi(|V|)}),$$

define the processed and unresolved vertex sets by

$$V_i^{\pi} = \{v_{\pi(1)}, \dots, v_{\pi(i)}\}$$

and

$$\overline{V_i^{\pi}} = V \setminus V_i^{\pi}.$$

Define the structural graph-coloring candidate interface by

$$\widetilde{B}_i^{G,\pi} = \left\{ v \in V_i^{\pi} : \exists w \in \overline{V_i^{\pi}} \text{ with } \{v, w\} \in E \right\}. \tag{5.10}$$

This set is exactly the classical vertex-separation boundary of the ordering π.

However, the equality

$$b_i^{\pi,\tau}(G) = \left| \widetilde{B}_i^{G,\pi} \right|$$

does not follow solely from this structural observation.

Semantic redundancy may permit a smaller task-sufficient coordinate set.

The generally valid relation is

$$b_i^{\pi,\tau}(G) \leq \left| \widetilde{B}_i^{G,\pi} \right| \tag{5.11}$$

whenever the structural boundary determines the selected reconstruction-safe signature.

Consequently,

$$\widehat{\omega}_{\mathrm{rec}}(G, \pi, \tau; \mathfrak{M}) \leq \max_i \left| \widetilde{B}_i^{G,\pi} \right|. \tag{5.12}$$

The right-hand side is the ordering-specific vertex-separation value.

Therefore, classical vertex separation provides a structural upper bound for the candidate interface size.

The CPR-Width value may be strictly smaller if semantic quotienting identifies reconstruction-equivalent boundary states.

The left-hand side is 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.

Its role is to embed a classical structural boundary into a task-relative reconstruction model with semantic quotienting and complete runtime accounting.

5.3.1 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

$$N_{\mathrm{raw}} = 3^4 = 81. \tag{5.13}$$

Let

$$\Phi(C_4) = \{c : V \to \{1,2,3\} : c(u) \neq c(v) \text{ for every } \{u,v\} \in E\} \tag{5.14}$$

be the set of proper three-colorings.

The number of admissible colorings is

$$N_{\mathrm{adm}} = |\Phi(C_4)| = 18. \tag{5.15}$$

Fix the partial coloring

$$f(v_1) = 1, \qquad f(v_2) = 2. \tag{5.16}$$

The admissible completions are

$$(c(v_3), c(v_4)) \in \{(1,2),(1,3),(3,2)\}. \tag{5.17}$$

Hence,

$$N_{\mathrm{rec}}(f) = 3. \tag{5.18}$$

The three quantities describe different spaces:

$$N_{\mathrm{raw}} = 81, \qquad N_{\mathrm{adm}} = 18, \qquad N_{\mathrm{rec}}(f) = 3. \tag{5.19}$$

In particular,

$$N_{\mathrm{raw}} \neq N_{\mathrm{adm}} \neq N_{\mathrm{rec}}(f). \tag{5.20}$$

The three quantities represent different finite cardinality spaces appearing during structural restriction and reconstruction.

Here N_{raw} is the cardinality of the complete raw assignment space, N_{adm} is the cardinality of the admissible proper-coloring space, and $N_{\mathrm{rec}}(f)$ is the cardinality of the reconstruction fiber induced by the fixed partial coloring f.

These quantities are finite cardinalities.

They are 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.

These three cardinalities distinguish the raw assignment space, the admissible coloring space, and one reconstruction fiber.

They are not direct runtime comparisons.

The admissible-to-fiber factor is

$$K_{\mathrm{adm/fiber}} = \frac{18}{3} = 6, \tag{5.21}$$

and the raw-to-fiber factor is

$$K_{\mathrm{raw/fiber}} = \frac{81}{3} = 27. \tag{5.22}$$

5.4 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

$$\widetilde{B}_i^{\pi,\tau} = \text{structural candidate interface}$$

from

$$B_i^{\pi,\tau} = \text{cardinality-minimal task-sufficient interface.}$$

The following table summarizes the representative realizations introduced above and developed in the subsequent sections. Table 5.1 records the structural realizations and the corresponding instance-specific CPR-Width notations.

Table 5.1: Problem-class realizations of Controlled Partial Reconstruction Width.

Problem class	**Active interface**	**Semantic interpretation**	**Instance-specific width**
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.	$\omega_{\mathrm{rec}}(T, \pi)$
Full adder	Task-relevant local state sufficient for exact output reconstruction.	Future output behavior.	$\omega_{\mathrm{rec}}(A, \pi)$

A computational problem class can contain a partially reconstructible subclass whenever its instances admit an admissible reconstruction ordering with controlled CPR-Width and satisfy the complete CPR boundary and runtime conditions.

5.5 Common Reconstruction Chain

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

$$\text{Ordering} \longrightarrow \text{Structural Candidate Interface} \longrightarrow \text{Task-Sufficient Interface} \tag{5.23}$$

followed by

$$\text{Reachable Interface States} \longrightarrow \text{Semantic Quotient} \longrightarrow \text{Reconstruction.} \tag{5.24}$$

In symbolic form,

$$\pi \longrightarrow \widetilde{B}_i^{\pi,\tau} \longrightarrow B_i^{\pi,\tau} \longrightarrow U_i^{\mathrm{reach}} \longrightarrow S_i \longrightarrow \mathrm{Output}_\tau\,. \tag{5.25}$$

The candidate interface is problem-class dependent.

Task sufficiency, minimality, semantic quotienting, and complete runtime accounting are governed by the general CPR-Width theory.

5.6 Tensor Representations

Let

$$T_P : I(P) \times J(P) \times K(P) \longrightarrow \mathcal{A}$$

be a finite structural tensor associated with a problem instance P.

A reconstruction ordering may process tensor indices, fragments, or local structural relations.

Define a structural tensor candidate interface by

$$\widetilde{B}_i^{T_P,\pi} = \{\alpha \in V_i^{\pi} : \alpha \text{ still influences an unresolved tensor interaction}\}. \tag{5.26}$$

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, it may preserve

- 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

$$b_i^{\pi,\tau}(T_P) \leq \left|\widetilde{B}_i^{T_P,\pi}\right| \tag{5.27}$$

and

$$\widehat{\omega}_{\mathrm{rec}}(T_P, \pi, \tau; \mathfrak{M}) \leq \max_i \left|\widetilde{B}_i^{T_P,\pi}\right|. \tag{5.28}$$

Tensor representation supplies a structural carrier.

It does not by itself prove a small CPR-Width, efficient semantic quotienting, or polynomial runtime.

5.7 Full Adder

The full adder provides a finite output-preserving quotient.

Let

$$\mathbb{B} = \{0, 1\}$$

and define the input space

$$X_{\mathrm{FA}} = \mathbb{B}^3.$$

An input state has the form

$$x = (a, b, c_{\mathrm{in}}).$$

The full-adder output map is

$$F_{\mathrm{FA}} : \mathbb{B}^3 \longrightarrow \mathbb{B}^2.$$

Define the Hamming-weight projection

$$C_{\mathrm{FA}}(a, b, c_{\mathrm{in}}) = a + b + c_{\mathrm{in}}. \tag{5.29}$$

Thus,

$$C_{\mathrm{FA}} : \mathbb{B}^3 \longrightarrow \{0, 1, 2, 3\}.$$

Define

$$Q_{\mathrm{FA}}(h) = \left(h \bmod 2, \left\lfloor \frac{h}{2} \right\rfloor\right). \tag{5.30}$$

Then

$$F_{\mathrm{FA}} = Q_{\mathrm{FA}} \circ C_{\mathrm{FA}}. \tag{5.31}$$

The eight input states collapse into four equivalence classes induced by the Hamming-weight quotient:

$$8 \longrightarrow 4. \tag{5.32}$$

This is an exact finite output quotient.

The quotient demonstrates semantic compression, but it becomes a CPR realization only after an ordering, active interface definition, and reconstruction transition system are explicitly specified.

It should not automatically be identified with

$$S_{\mathrm{CPR}} = 4$$

unless the full adder is embedded into a fully declared staged CPR model.

For the isolated finite output task, the exact statement is

$$S_{\mathrm{out}}(P_{\mathrm{FA}}) = 4. \tag{5.33}$$

A later chapter may identify this value with a CPR semantic-state count after specifying the complete reconstruction-stage structure, task, ordering, and model.

5.8 Synthesis of Problem-Class Realizations

The problem classes considered in this chapter possess different structural boundaries.

For SAT, the candidate boundary is induced by residual clauses.

For CSP, it is induced by unresolved constraint scopes.

For graph coloring, it is induced by edges crossing the processed and unresolved vertex sets.

For tensor representations, it is induced by unresolved structural interactions.

For digital arithmetic, it is induced by the local state required for exact output behavior.

These structural boundaries provide candidate interfaces.

They become certified CPR interfaces only after task sufficiency has been proved.

They become exact minimum CPR interfaces only after cardinality minimality has also been proved.

Accordingly, every problem-class realization should establish, in this order,

1. an explicit component representation;
2. a declared task;
3. an admissible reconstruction ordering;
4. a structural candidate interface;
5. a reconstruction-safe signature satisfying S1–S3;
6. task sufficiency of the candidate interface;
7. cardinality minimality, if exact width equality is claimed;
8. semantic-state and transition bounds;
9. and all six complete runtime-cost bounds.

The following chapter develops the staged reconstruction model for digital arithmetic and derives the

corresponding CPR-Width and runtime conditions.

6 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 in the present manuscript is not to introduce a new addition algorithm. Their role is to demonstrate that the CPR-WIDTH framework can represent known local and linear reconstruction structures 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. Second, the n-bit ripple-adder family admits a uniform staged transducer model whose future behavior is controlled by one Boolean carry state. For the isolated full adder, the four Hamming-weight classes are treated as an output quotient. They are not identified automatically with a stage-wise CPR semantic state count. For the ripple-adder family, the declared task is exact evaluation of the addition map

$$\tau_{\mathrm{add}} = \text{exact computation of } (s_0, s_1, \ldots, s_{n-1}, c_n) \text{ from } (A, B, c_0). \tag{6.1}$$

All operation-count statements in this chapter use the fixed elementary-operation accounting model introduced in Chapter 3. This chapter develops the arithmetic reconstruction model and proves its structural and asymptotic properties, and records the exhaustive finite verification of the isolated full-adder quotient in Section 6.7.

6.1 Boolean Alphabet

Let

$$\mathbb{B} = \{0, 1\} \tag{6.2}$$

be the Boolean alphabet. A local full-adder input state is

$$x = (a, b, c_{\mathrm{in}}) \in \mathbb{B}^3, \tag{6.3}$$

where a and b are input bits and c_{in} is the incoming carry bit. The corresponding output state is

$$F_{\mathrm{FA}}(x) = (s, c_{\mathrm{out}}) \in \mathbb{B}^2, \tag{6.4}$$

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

Notational remark. This chapter uses three distinct state counts, introduced in later sections: S_{out} denotes the isolated full-adder output-quotient state count, S_{CPR} denotes the CPR semantic-state count, and S_{carry} denotes the carry automaton's state-alphabet size.

6.2 Full-Adder Map

The full-adder map is

$$F_{\mathrm{FA}} : \mathbb{B}^3 \longrightarrow \mathbb{B}^2. \tag{6.5}$$

It is defined by

$$s = a \oplus b \oplus c_{\mathrm{in}}, \tag{6.6}$$

and

$$c_{\mathrm{out}} = ab \vee ac_{\mathrm{in}} \vee bc_{\mathrm{in}}. \tag{6.7}$$

Equivalently,

$$a + b + c_{\mathrm{in}} = s + 2c_{\mathrm{out}}. \tag{6.8}$$

The complete truth table is shown in Table 6.1.

Table 6.1: 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

6.3 Hamming-Weight Projection

Define the Hamming-weight projection

$$C_{\text{FA}} : \mathbb{B}^3 \longrightarrow \{0, 1, 2, 3\} \tag{6.9}$$

by

$$C_{\text{FA}}(a, b, c_{\text{in}}) = a + b + c_{\text{in}}. \tag{6.10}$$

The eight Boolean input states are mapped to four Hamming-weight states:

$$8 \longrightarrow 4. \tag{6.11}$$

Let

$$Z_{\text{FA}} = \{0, 1, 2, 3\} \tag{6.12}$$

be the reduced Hamming-weight space. Define the canonical quotient reconstruction map

$$Q_{\text{FA}} : Z_{\text{FA}} \longrightarrow \mathbb{B}^2 \tag{6.13}$$

by

$$Q_{\text{FA}}(h) = \left(h \bmod 2, \left\lfloor \frac{h}{2} \right\rfloor\right). \tag{6.14}$$

The first component of $Q_{\text{FA}}(h)$ represents the sum bit s, while the second component represents the carry bit c_{out}. Then

$$F_{\text{FA}} = Q_{\text{FA}} \circ C_{\text{FA}}. \tag{6.15}$$

6.4 Exact Output Reconstruction

Theorem 6.1 (Exact Full-Adder Factorization)**.** *For every input state*

$$x = (a, b, c_{\text{in}}) \in \mathbb{B}^3,$$

the complete full-adder output is determined uniquely by the Hamming weight

$$h = C_{\text{FA}}(x).$$

Consequently,

$$F_{\text{FA}}(x) = Q_{\text{FA}}\left(C_{\text{FA}}(x)\right). \tag{6.16}$$

Proof. Let

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

Since

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

its parity determines the sum bit:

$$s = h \bmod 2. \tag{6.17}$$

Its integer quotient by two determines the outgoing carry:

$$c_{\text{out}} = \left\lfloor \frac{h}{2} \right\rfloor. \tag{6.18}$$

Therefore, the output pair (s, c_{out}) is uniquely determined by h. Hence,

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

□

The factorization preserves the declared output. It does not preserve the original input state uniquely. As shown in Section 6.5 below, Q_{FA} is in fact bijective between Z_{FA} and $\mathbb{B}^2$; this is used repeatedly in what follows.

6.5 Reconstruction Fibers

For every

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

define the Hamming-weight fiber by

$$\Omega_{\mathrm{FA}}(h) = C_{\mathrm{FA}}^{-1}(\{h\}) = \left\{x \in \mathbb{B}^3 : C_{\mathrm{FA}}(x) = h\right\}. \tag{6.19}$$

The four fibers are

$$\Omega_{\mathrm{FA}}(0) = \{(0,0,0)\}, \tag{6.20}$$

$$\Omega_{\mathrm{FA}}(1) = \{(1,0,0),(0,1,0),(0,0,1)\}, \tag{6.21}$$

$$\Omega_{\mathrm{FA}}(2) = \{(1,1,0),(1,0,1),(0,1,1)\}, \tag{6.22}$$

and

$$\Omega_{\mathrm{FA}}(3) = \{(1,1,1)\}. \tag{6.23}$$

Therefore,

$$(|\Omega_{\mathrm{FA}}(0)|, |\Omega_{\mathrm{FA}}(1)|, |\Omega_{\mathrm{FA}}(2)|, |\Omega_{\mathrm{FA}}(3)|) = (1,3,3,1). \tag{6.24}$$

The maximum fiber size is

$$d_{\mathrm{rec}}^{\max}(P_{\mathrm{FA}}) = 3. \tag{6.25}$$

The map Q_{FA} is bijective. Indeed,

$$0 \mapsto (0,0), \qquad 1 \mapsto (1,0), \qquad 2 \mapsto (0,1), \qquad 3 \mapsto (1,1).$$

Because the quotient reconstruction map is injective, two input states have identical full-adder outputs if and only if they belong to the same Hamming-weight fiber. Consequently,

$$C_{\mathrm{FA}}(x) = C_{\mathrm{FA}}(y) \quad \Longleftrightarrow \quad F_{\mathrm{FA}}(x) = F_{\mathrm{FA}}(y). \tag{6.26}$$

Thus, the Hamming-weight fibers are exactly the output-equivalence classes of the isolated full adder.

6.6 Output-Quotient State Count

For the isolated full-adder output task, define

$$x \equiv_{\mathrm{out}} y \quad \Longleftrightarrow \quad F_{\mathrm{FA}}(x) = F_{\mathrm{FA}}(y). \tag{6.27}$$

The corresponding output quotient is

$$\mathbb{B}^3/\!\equiv_{\mathrm{out}} = \{\Omega_{\mathrm{FA}}(0), \Omega_{\mathrm{FA}}(1), \Omega_{\mathrm{FA}}(2), \Omega_{\mathrm{FA}}(3)\}. \tag{6.28}$$

Define

$$S_{\mathrm{out}}(P_{\mathrm{FA}}) = \left|\mathbb{B}^3/\!\equiv_{\mathrm{out}}\right|. \tag{6.29}$$

Then

$$S_{\mathrm{out}}(P_{\mathrm{FA}}) = 4. \tag{6.30}$$

The associated binary output-state dimension is

$$D_{\mathrm{out}}(P_{\mathrm{FA}}) = \lceil \log_2 S_{\mathrm{out}}(P_{\mathrm{FA}}) \rceil = 2. \tag{6.31}$$

The quantities S_{out} and D_{out} refer to the isolated output quotient. They are not identified automatically with S_{CPR} and D_{CPR}. The isolated full-adder quotient is therefore an exact semantic output quotient and not yet a complete CPR reconstruction model. A CPR realization additionally requires an ordering, active interface definition, transition relation, and runtime model.

6.7 Finite Verification Record

The full-adder Hamming-weight factorization proved in Section 6.4 is confirmed here by exhaustive enumeration of the complete finite input space $\mathbb{B}^3$.

The complete finite input space and the corresponding outputs are shown in Table 6.2.

Table 6.2: Complete finite verification table for the full-adder Hamming-weight factorization.

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

For every row, Equation (6.8) holds.

The last three columns also show directly that

$$F_{\text{FA}}(x) = Q_{\text{FA}}\left(C_{\text{FA}}(x)\right) \tag{6.32}$$

for every one of the eight possible input states.

The row-wise verification provides an exhaustive verification of Theorem 6.1, together with the fiber cardinalities, output-quotient count, and injectivity of Q_{FA} established in Section 6.5 and Section 6.6.

The exhaustive enumeration verifies the following finite statements:

- the complete Boolean input space contains exactly eight states, and the Hamming-weight projection produces four reduced states, corresponding to four output-equivalence classes, $8 \to 4$;
- the Hamming-weight fibers of C_{FA} have cardinalities $(1,3,3,1)$ and partition $\mathbb{B}^3$;
- Q_{FA} is a bijection between Z_{FA} and the image $F_{\text{FA}}(\mathbb{B}^3) = \mathbb{B}^2$, so the output quotient contains exactly four classes, $S_{\text{out}}(P_{\text{FA}}) = 4$;
- and two input states share an output exactly when they share a Hamming weight.

This finite record has Evidence Level E1 (finite verification evidence, Section 6.22.8): it is checked exhaustively on the complete eight-state input space, but it does not by itself establish an asymptotic theorem, a complete operation-count advantage, or a measured wall-clock advantage. Nor does it imply corresponding structural or asymptotic properties for arbitrary circuit families.

Together with the structural proofs of the preceding sections, this finite verification completes the correctness record for the isolated full-adder example used throughout Chapter 6.

6.8 Ripple-Adder Function Family

For every $n \geq 1$, define the n-bit ripple-adder function

$$F_{\text{add},n} : \mathbb{B}^n \times \mathbb{B}^n \times \mathbb{B} \longrightarrow \mathbb{B}^{n+1}. \tag{6.33}$$

Its input is

$$(A, B, c_0),$$

where

$$A = (a_{n-1} \cdots a_1 a_0)_2 \tag{6.34}$$

and

$$B = (b_{n-1} \cdots b_1 b_0)_2. \tag{6.35}$$

The output is

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

For every position

$$i \in \{0, \ldots, n-1\},$$

define

$$h_i = a_i + b_i + c_i, \tag{6.36}$$

$$s_i = h_i \bmod 2, \tag{6.37}$$

and

$$c_{i+1} = \left\lfloor \frac{h_i}{2} \right\rfloor. \tag{6.38}$$

The complete arithmetic identity is

$$A + B + c_0 = \sum_{i=0}^{n-1} s_i 2^i + c_n 2^n. \tag{6.39}$$

The CPR model developed below is a uniform transducer model for the complete function $F_{\mathrm{add},n}$. A fixed encoded input (A, B, c_0) determines one deterministic path through this transducer.

6.9 Uniform Carry Transducer

Define the carry-state alphabet by

$$\mathcal{Q}_{\mathrm{carry}} = \{0, 1\}. \tag{6.40}$$

Define the local input alphabet by

$$\mathcal{X}_{\mathrm{local}} = \mathbb{B}^2. \tag{6.41}$$

A local input label is

$$\alpha_i = (a_i, b_i) \in \mathcal{X}_{\mathrm{local}}.$$

Define the carry transition map

$$\delta_{\mathrm{carry}} : \mathcal{Q}_{\mathrm{carry}} \times \mathcal{X}_{\mathrm{local}} \longrightarrow \mathcal{Q}_{\mathrm{carry}} \tag{6.42}$$

by

$$\delta_{\mathrm{carry}}(c, (a, b)) = \left\lfloor \frac{a + b + c}{2} \right\rfloor. \tag{6.43}$$

Define the local output map

$$\mu_{\mathrm{carry}} : \mathcal{Q}_{\mathrm{carry}} \times \mathcal{X}_{\mathrm{local}} \longrightarrow \mathbb{B} \tag{6.44}$$

by

$$\mu_{\mathrm{carry}}(c, (a, b)) = (a + b + c) \bmod 2. \tag{6.45}$$

Thus, every local step is

$$(c_i, (a_i, b_i)) \longmapsto (s_i, c_{i+1}). \tag{6.46}$$

The transducer has two carry states. For every carry state, four local input labels are possible. Hence, its complete local labeled transition table contains at most

$$2 \cdot 4 = 8 \tag{6.47}$$

labeled transitions. This is a property of the uniform family-level transducer. It is not the number of transitions executed during one fixed addition run.

6.10 CPR Component Representation

For the uniform n-bit transducer, define the reconstruction components

$$\gamma_1, \gamma_2, \ldots, \gamma_n, \tag{6.48}$$

where γ_j represents the carry state after processing bit positions

$$0, \ldots, j-1.$$

Formally,

$$\gamma_j \equiv c_j,$$

the carry value conventionally written c_j in the addition τ_{add} of Equation (6.1). Throughout this chapter, γ_j denotes the reconstruction component itself, while a value $\gamma_j(r)$ assigned to it under a prefix state r is the corresponding semantic state value; the signature and interface constructions below are stated in terms of these state values. The component set is

$$V(P_{\text{add},n}) = \{\gamma_1, \ldots, \gamma_n\}. \tag{6.49}$$

Every component has the Boolean domain

$$D_{\gamma_j} = \mathbb{B}. \tag{6.50}$$

The input bits a_i, b_i and the initial carry c_0 are external input labels of a transducer run. They are not reconstructed component coordinates. The reconstruction ordering is

$$\pi_{\text{add}} = (\gamma_1, \gamma_2, \ldots, \gamma_n). \tag{6.51}$$

The number of nonterminal reconstruction stages is

$$m_{\text{stage}}(P_{\text{add},n}, \pi_{\text{add}}) = n. \tag{6.52}$$

After stage j,

$$V_j^{\pi_{\text{add}}} = \{\gamma_1, \ldots, \gamma_j\}, \qquad 0 \le j \le n, \tag{6.53}$$

with the convention

$$V_0^{\pi_{\text{add}}} = \varnothing.$$

The unresolved suffix is

$$\overline{V_j^{\pi_{\text{add}}}} = \{\gamma_{j+1}, \ldots, \gamma_n\}. \tag{6.54}$$

6.11 Family-Level Reachability and Fixed Runs

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. Thus, the family-level reachable prefix space is denoted by

$$\mathcal{R}_j^{\pi_{\text{add}},\text{unif}}(P_{\text{add},n}). \tag{6.55}$$

Both carry values are family-reachable at every existing nonterminal position $1 \le j < n$ (for $n = 1$ this range is empty, and the statement holds vacuously):

$$\{0,1\} \subseteq \left\{\gamma_j(r) : r \in \mathcal{R}_j^{\pi_{\text{add}},\text{unif}}(P_{\text{add},n})\right\}. \tag{6.56}$$

For one fixed encoded input

$$(A, B, c_0),$$

the deterministic transition rule selects exactly one carry sequence

$$c_1, c_2, \ldots, c_n.$$

Therefore, 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 is used to characterize the uniform CPR interface and semantic alphabet. The latter is used to count the operations performed for one encoded input.

6.12 Carry Signature and Active Interface

For every stage

$$1 \le j < n,$$

define the family-level carry signature by

$$\text{Sig}_{\tau_{\text{add}},j}^{\pi_{\text{add}}}(r) = \gamma_j(r). \tag{6.57}$$

At stage zero, the externally supplied initial carry is the current transducer state. 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 by

$$\widetilde{B}_j^{\pi_{\mathrm{add}},\tau_{\mathrm{add}}} = \{\gamma_j\}. \tag{6.58}$$

At the initial and terminal stages, select

$$\widetilde{B}_0^{\pi_{\mathrm{add}},\tau_{\mathrm{add}}} = \widetilde{B}_n^{\pi_{\mathrm{add}},\tau_{\mathrm{add}}} = \varnothing. \tag{6.59}$$

Proposition 6.1 (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 and r' be two family-reachable prefix states at stage j satisfying

$$r|_{\{\gamma_j\}} = r'|_{\{\gamma_j\}}.$$

Then

$$\gamma_j(r) = \gamma_j(r').$$

By the definition of the carry signature,

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

and

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

Therefore,

$$\mathrm{Sig}^{\pi_{\mathrm{add}}}_{\tau_{\mathrm{add}},j}(r) = \mathrm{Sig}^{\pi_{\mathrm{add}}}_{\tau_{\mathrm{add}},j}(r').$$

Hence, the one-component interface is task-sufficient. □

6.13 Minimality of the Carry Interface

Proposition 6.2 (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,*

$$b_j^{\pi_{\mathrm{add}},\tau_{\mathrm{add}}}(P_{\mathrm{add},n}) = 1. \tag{6.60}$$

Proof. By family-level reachability, there exist reachable prefix states r_0 and r_1 satisfying

$$\gamma_j(r_0) = 0$$

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 residual continuation functions for the remaining computation. Since the declared task τ_{add} requires reconstruction of every output bit s_i, the difference in the next output bit is task-relevant. Therefore, the two reachable prefix states belong to different future-equivalence classes. Since the carry signature is exactly the carry value, $\mathrm{Sig}^{\pi_{\mathrm{add}}}_{\tau_{\mathrm{add}},j}(r_0) = \gamma_j(r_0) = 0 \neq 1 = \gamma_j(r_1) = \mathrm{Sig}^{\pi_{\mathrm{add}}}_{\tau_{\mathrm{add}},j}(r_1)$, so their signatures are different. The empty interface cannot distinguish them and is not task-sufficient. Since the singleton $\{\gamma_j\}$ is task-sufficient,

its cardinality is minimal. Hence,

$$b_j^{\pi_{\mathrm{add}},\tau_{\mathrm{add}}}(P_{\mathrm{add},n}) = 1.$$

□

It follows that

$$\widehat{\omega}_{\mathrm{rec}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = 1. \tag{6.61}$$

By the normalization convention,

$$\omega_{\mathrm{rec}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = 0. \tag{6.62}$$

Thus, normalized width zero still corresponds to one active Boolean carry component.

6.14 Reconstruction Safety of the Carry Signature

Proposition 6.3 (Carry Signature Is Reconstruction-Safe)**.** *The carry-signature family satisfies the reconstruction-safety conditions S1–S3 of Chapter 2.*

Proof. Let r and r' be two family-reachable prefix states at the same stage j with equal carry signatures. Then

$$\gamma_j(r) = \gamma_j(r').$$

For every common next input label

$$(a_j, b_j) \in \mathbb{B}^2,$$

the local arithmetic value is determined by

$$h_j = a_j + b_j + \gamma_j.$$

Because the carry values agree, the same input label produces the same sum bit and the same successor carry. Therefore, equal carry signatures induce identical residual transducer behavior. This establishes S1. The complete local input alphabet $\mathbb{B}^2$ is available from both equal carry states. Therefore, the admissible next-label sets coincide. This establishes S2. Finally, the same next input label applied to equal carry states produces equal successor carry values. Hence, the successor signatures coincide. This establishes S3. Therefore, the carry-signature family is reconstruction-safe. □

6.15 Inductive Evaluation of a Fixed Input

Theorem 6.2 (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

$$a_0, \qquad b_0, \qquad c_0$$

determine

$$h_0 = a_0 + b_0 + c_0.$$

Therefore, they determine uniquely

$$s_0 = h_0 \bmod 2$$

and

$$c_1 = \left\lfloor \frac{h_0}{2} \right\rfloor.$$

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

$$c_{i+1} = \left\lfloor \frac{h_i}{2} \right\rfloor.$$

Thus, every sum bit and the final carry are determined uniquely. □

6.16 Uniform Semantic-State Control

The uniform carry-state alphabet contains two states:

$$S^{\mathrm{unif}}_{\mathrm{carry}} = |\mathcal{Q}_{\mathrm{carry}}| = 2. \tag{6.63}$$

This is a family-level state-alphabet count. For a fixed encoded input, the deterministic evaluation follows one carry state at each stage. Therefore,

$$|S^{\mathrm{run}}_j(A, B, c_0)| = 1 \tag{6.64}$$

for every stage of one fixed run. The uniform family-level bound is

$$|S^{\mathrm{unif}}_j| \le 2. \tag{6.65}$$

For this construction, the CPR semantic-state space at each stage is represented by (a subset of) the uniform carry-state alphabet $\mathcal{Q}_{\mathrm{carry}}$; hence $S^{\mathrm{unif}}_{\mathrm{CPR}}$ is bounded by $S^{\mathrm{unif}}_{\mathrm{carry}}$, because every reachable semantic state is uniquely represented by the current carry value. Consequently,

$$S^{\mathrm{unif}}_{\mathrm{CPR}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) \le 2. \tag{6.66}$$

The corresponding uniform binary state dimension satisfies

$$D^{\mathrm{unif}}_{\mathrm{CPR}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) \le 1. \tag{6.67}$$

Definition 6.1 (Total Stage-Indexed Semantic-State Count)**.** Define

$$N^{\mathrm{unif}}_{\mathrm{sem,state}}(P, \pi, \tau; \mathfrak{M}) = \sum_{j=0}^{m_{\mathrm{stage}}(P,\pi)} |S^{\mathrm{unif}}_j(P, \pi, \tau; \mathfrak{M})|. \tag{6.68}$$

For the ripple-adder transducer,

$$N^{\mathrm{unif}}_{\mathrm{sem,state}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) \le 2(n+1). \tag{6.69}$$

Definition 6.2 (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

$$E^{\mathrm{unif}}_{\mathrm{sem}}(P, \pi, \tau; \mathfrak{M}) = \sum_{j=0}^{m_{\mathrm{stage}}(P,\pi)-1} \left|\mathcal{E}^{\mathrm{sem}}_j(P, \pi, \tau; \mathfrak{M})\right|. \tag{6.70}$$

The uniform ripple-adder transducer has at most eight labeled transition edges per stage. Therefore,

$$E^{\mathrm{unif}}_{\mathrm{sem}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) \le 8n. \tag{6.71}$$

This bound counts all possible local labeled transitions of the uniform transducer, not only the transitions reachable under one fixed input. For a fixed encoded input, exactly one transition edge is executed per bit position:

$$E^{\mathrm{run}}_{\mathrm{sem}}(A, B, c_0) = n. \tag{6.72}$$

6.17 Complete Runtime Bounds

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. The one-component carry interface is supplied explicitly and its sufficiency and minimality have already been proved. Thus, 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

$$N_{\mathrm{Build}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = O(n). \tag{6.73}$$

The previously established carry-interface rule is applied once per stage. Therefore,

$$N_{\mathrm{Reduce}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = O(n). \tag{6.74}$$

One deterministic transition is executed per bit position, and every transition uses a constant number of elementary Boolean and fixed-width arithmetic operations. Hence,

$$N_{\text{Transition}}(P_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{add}}; \mathfrak{M}_{\text{add}}) = O(n). \tag{6.75}$$

One sum bit is reconstructed at every bit position, followed by one final carry. Therefore,

$$N_{\text{Reconstruct}}(P_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{add}}; \mathfrak{M}_{\text{add}}) = O(n). \tag{6.76}$$

The identity

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

can be verified independently at every bit position. Thus,

$$N_{\text{Verify}}(P_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{add}}; \mathfrak{M}_{\text{add}}) = O(n). \tag{6.77}$$

The explicit output contains $n + 1$ bits:

$$L_{\text{Output}}(P_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{add}}; \mathfrak{M}_{\text{add}}) = n + 1. \tag{6.78}$$

Under the elementary bit-output model, outputting each bit requires constant cost. Under the elementary output model of Chapter 3, this gives

$$N_{\text{Output}}(P_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{add}}; \mathfrak{M}_{\text{add}}) = \Theta(n). \tag{6.79}$$

By the complete six-component decomposition,

$$\begin{aligned} N_{\text{CPR}}(P_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{add}}; \mathfrak{M}_{\text{add}}) = {} & N_{\text{Build}} + N_{\text{Reduce}} + N_{\text{Transition}} \\ & + N_{\text{Reconstruct}} + N_{\text{Verify}} + N_{\text{Output}}. \end{aligned} \tag{6.80}$$

Every component is $O(n)$, and the explicit output requires $\Omega(n)$ operations. Consequently,

$$N_{\text{CPR}}(P_{\text{add},n}, \pi_{\text{add}}, \tau_{\text{add}}; \mathfrak{M}_{\text{add}}) = \Theta(n). \tag{6.81}$$

This result reproduces the known linear structure of ripple-carry addition inside the complete CPR accounting framework. It is not presented as a new addition algorithm.

6.18 Boundary-Condition Record

The uniformly declared ripple-adder transducer satisfies the four CPR boundary conditions for the exact-output task. The runtime decomposition conditions are verified separately from the structural boundary conditions: BC1–BC4 govern the structural and semantic representation, while the six complete runtime-cost bounds of the next section govern the operation count.

BC1 – Explicit Uniform Structural Representation. The following objects are explicitly defined:

- 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.

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,

$$a_i + b_i + c_i = s_i + 2c_{i+1}. \tag{6.82}$$

The inductive evaluation theorem proves that every fixed input produces exactly one correct output

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

Conversely, the deterministic transition sequence constructs the complete declared output tuple for every encoded input. Therefore, the representation is complete with respect to τ_{add}. Therefore, BC3 holds.

BC4 – Polynomial Semantic-State and Transition Control. The uniform family-level semantic quantities satisfy

$$N_{\mathrm{sem,state}}^{\mathrm{unif}} \leq 2(n+1) \tag{6.83}$$

and

$$E_{\mathrm{sem}}^{\mathrm{unif}} \leq 8n. \tag{6.84}$$

For every fixed encoded input, only one state and one transition edge are used at each stage. Therefore, BC4 holds. The complete cost bounds are

$$\begin{aligned} N_{\mathrm{Build}} &= O(n), \\ N_{\mathrm{Reduce}} &= O(n), \\ N_{\mathrm{Transition}} &= O(n), \\ N_{\mathrm{Reconstruct}} &= O(n), \\ N_{\mathrm{Verify}} &= O(n), \\ N_{\mathrm{Output}} &= \Theta(n). \end{aligned} \tag{6.85}$$

Consequently,

$$N_{\mathrm{CPR}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = \Theta(n). \tag{6.86}$$

Thus, the uniform ripple-adder family satisfies the complete CPR polynomial-cost criterion for the declared exact-output task.

6.19 Structural Interpretation

The isolated full adder demonstrates the exact output-preserving factorization

$$\mathbb{B}^3 \xrightarrow{C_{\mathrm{FA}}} \{0,1,2,3\} \xrightarrow{Q_{\mathrm{FA}}} \mathbb{B}^2. \tag{6.87}$$

The ripple-adder family demonstrates uniform staged reconstruction through the two-state carry transducer

$$(c_i, (a_i, b_i)) \longmapsto (s_i, c_{i+1}). \tag{6.88}$$

At every nonterminal stage, the active task-sufficient component is the current carry component

$$\gamma_j.$$

Therefore,

$$\widehat{\omega}_{\mathrm{rec}} = 1 \tag{6.89}$$

and

$$\omega_{\mathrm{rec}} = 0. \tag{6.90}$$

The normalized width subtracts the mandatory one-component state carrier according to the convention introduced in Chapter 2. Therefore, normalized width zero does not mean the absence of an active interface; it means the active interface attains the minimum nonzero cardinality of one component. The family-level model contains two possible carry states. A fixed encoded input follows one deterministic path through that two-state model.

6.20 Conclusion

The full adder admits the exact output-preserving factorization

$$\mathbb{B}^3 \xrightarrow{C_{\mathrm{FA}}} \{0,1,2,3\} \xrightarrow{Q_{\mathrm{FA}}} \mathbb{B}^2. \tag{6.91}$$

Its Hamming-weight fibers have cardinalities

$$(1,3,3,1). \tag{6.92}$$

The isolated output quotient contains four classes:

$$S_{\mathrm{out}}(P_{\mathrm{FA}}) = 4. \tag{6.93}$$

For the uniform ripple-adder transducer, the active nonterminal interface is the current carry component. Therefore,

$$\widehat{\omega}_{\mathrm{rec}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = 1 \tag{6.94}$$

and

$$\omega_{\mathrm{rec}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = 0. \tag{6.95}$$

The uniform semantic representation satisfies

$$S^{\mathrm{unif}}_{\mathrm{CPR}} \le 2, \qquad N^{\mathrm{unif}}_{\mathrm{sem,state}} \le 2(n+1), \qquad E^{\mathrm{unif}}_{\mathrm{sem}} \le 8n. \tag{6.96}$$

For one fixed encoded input, one state and one transition are used at each stage. All six runtime components are linearly bounded, and the explicit output requires linear work. Consequently,

$$N_{\mathrm{CPR}}(P_{\mathrm{add},n}, \pi_{\mathrm{add}}, \tau_{\mathrm{add}}; \mathfrak{M}_{\mathrm{add}}) = \Theta(n). \tag{6.97}$$

Thus, the uniform ripple-adder family satisfies BC1–BC4 and the complete polynomial-cost requirements for the declared exact-output task. 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 without additional structural assumptions. Section 6.7 records the exhaustive finite verification of the isolated full-adder factorization and its output-equivalence fibers.

The preceding chapters established Controlled Partial Reconstruction Width as a reconstruction-oriented structural framework for finite combinatorial problems.

Throughout this chapter, let P be a finite problem instance, let τ be the declared computational task, let $\mathfrak{M}$ be the fixed CPR reconstruction model, and let

$$\pi \in \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M}) \tag{6.98}$$

be an admissible reconstruction ordering.

The central structural-runtime chain of the manuscript is

$$\boldsymbol{\omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) \longrightarrow S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) \longrightarrow N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M})}. \tag{6.99}$$

The first quantity denotes the normalized Controlled Partial Reconstruction Width.

The second denotes the maximum number of semantic quotient classes occurring at one reconstruction stage.

The third denotes the complete CPR operation count.

When no ambiguity can arise, the dependence on τ and $\mathfrak{M}$ may be suppressed for readability.

The chain in Equation (6.99) expresses possible structural dependencies.

It does not state that small reconstruction width alone implies polynomial-time tractability.

As established in Chapter 3, the complete CPR operation count is

$$\begin{aligned} N_{\mathrm{CPR}} = {} & N_{\mathrm{Build}} + N_{\mathrm{Reduce}} + N_{\mathrm{Transition}} \\ & + N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}. \end{aligned} \tag{6.100}$$

A small value of ω_{rec} controls only one structural contribution to the complete analysis.

A polynomial-time conclusion requires uniform polynomial control of every component of the declared CPR procedure.

For reference, define the runtime-index set

$$\mathcal{J} = \{\text{Build}, \text{Reduce}, \text{Transition}, \text{Reconstruct}, \text{Verify}, \text{Output}\}. \tag{6.101}$$

Let

$$\mathcal{F} = \{P_x : x \in \{0,1\}^*\} \tag{6.102}$$

be a uniformly encoded family of finite decision-problem instances.

Let

$$L_{\mathcal{F}} = \{x \in \{0,1\}^* : P_x \text{ is a yes-instance}\} \tag{6.103}$$

be the associated decision language.

A complete uniform CPR decision model requires

- one fixed decision task τ;
- one fixed uniform reconstruction model $\mathfrak{M}$;
- one deterministic preprocessing algorithm $A_{\text{Preprocess}}$, containing an ordering-supply component A_π;
- one uniform correctness specification;
- and six fixed polynomial functions $\{p_j\}_{j\in\mathcal{J}}$.

For every encoding x, let

$$\mathcal{D}_x = A_{\text{Preprocess}}(x) \tag{6.104}$$

denote the complete uniformly constructed CPR representation of P_x, including all structural data required before semantic-state processing begins.

Let

$$\pi_x = A_\pi(x) \tag{6.105}$$

be the ordering produced for P_x.

The operation counts of $A_{\text{Preprocess}}$ and A_π are included in

$$N_{\text{Build}}(P_x, \pi_x, \tau; \mathfrak{M}).$$

The complete uniform tractability requirement is

$$\left[\begin{array}{l} \exists \tau_0\, \exists \mathfrak{M}\, \exists A_{\text{Preprocess}}\, \exists \{p_j\}_{j\in\mathcal{J}}\, \forall x \in \{0,1\}^* : \\ \quad \mathcal{D}_x = A_{\text{Preprocess}}(x) \\ \quad \wedge\, \pi_x = A_\pi(x) \in \Pi_{\text{adm}}(P_x, \tau_0; \mathfrak{M}) \\ \quad \wedge \bigwedge_{k=1}^{4} \text{BC}_k(P_x, \pi_x, \tau_0; \mathfrak{M}) \\ \quad \wedge \bigwedge_{j\in\mathcal{J}} [N_j(P_x, \pi_x, \tau_0; \mathfrak{M}) \le p_j(|x|)] \end{array}\right] \Longrightarrow L_{\mathcal{F}} \in \mathbf{P}. \tag{6.106}$$

Here A_π is the ordering-supply component of $A_{\text{Preprocess}}$.

The order of quantifiers is essential.

The task, reconstruction model, preprocessing algorithm, ordering algorithm, correctness specification, and polynomial bounds must be fixed for the complete encoded family before an individual encoding is selected.

Under the complete definition of $\mathbf{P}_{\text{CPR}}$ given in Chapter 4, which includes BC1–BC4, uniform ordering supply, and polynomial bounds for all six runtime components, one has

$$\mathbf{P}_{\text{CPR}} \subseteq \mathbf{P}. \tag{6.107}$$

This inclusion is not a complexity-class collapse statement.

For an NP-complete language L^*, the implication

$$L^* \in \mathbf{P}_{\mathrm{CPR}} \Longrightarrow \mathbf{P} = \mathbf{NP} \tag{6.108}$$

remains strictly conditional.

No NP-complete language is proved in this work to belong to $\mathbf{P}_{\mathrm{CPR}}$.

Chapter 5 presents structural realization templates, controlled-width subclass theory, and finite illustrative calculations.

Chapter 6 establishes symbolic arithmetic reconstruction results, a uniform carry-transducer analysis, and an exhaustive finite verification record for the isolated full-adder Hamming-weight factorization.

These finite and structural results do not by themselves establish asymptotic tractability for unrestricted problem classes.

Count-level gain and measured wall-clock gain must also remain separate.

A favorable elementary-operation count is a statement within the declared accounting model.

A measured runtime claim requires a separate implementation, hardware description, software environment, benchmark protocol, and reproducible measurement procedure.

The remaining sections formulate the principal mathematical, algorithmic, and experimental research problems of CPR-WIDTH.

6.21 Structural and Algorithmic Open Problems

The first group of open problems concerns the internal mathematical structure of CPR reconstruction models.

The remainder of this chapter introduces the following record-level and comparative symbols, summarized here for reference:

$\mathcal{R}_{\mathrm{core}}$
the core structural and operation-count record for a single declared instance;
$\mathcal{R}_{\mathrm{measured}}$
the measured implementation record, extending $\mathcal{R}_{\mathrm{core}}$ with hardware- and wall-clock-dependent data;
$\mathcal{A}_{\mathrm{BC}}$
the boundary-condition audit record, at instance level or family level;
G_{count}
the count-level gain, comparing classical and CPR operation counts;
G_{time}
the measured wall-clock gain, comparing classical and CPR runtimes;
α_{width}
the unnormalized width ratio of a tested ordering relative to the minimum width;
κ_{bound}
the width-bound slack factor, comparing the Width–State Bound to the actual semantic-state count;
$\mathcal{O}$
the per-ordering experimental record used in ordering-ablation studies;
$\mathcal{W}$
the comparative record of CPR-Width against classical graph-width parameters.

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.

6.21.1 Optimal Reconstruction Orderings

Let

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

denote the set of admissible reconstruction orderings of the declared model.
Assume that

$$\Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M}) \neq \varnothing. \tag{6.109}$$

The minimum normalized CPR-Width is

$$\omega_{\mathrm{rec}}(P, \tau; \mathfrak{M}) = \min_{\pi \in \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M})} \omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}). \tag{6.110}$$

The corresponding minimum unnormalized width is

$$\widehat{\omega}_{\mathrm{rec}}(P, \tau; \mathfrak{M}) = \min_{\pi \in \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M})} \widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}). \tag{6.111}$$

The definition does not imply that a minimizing ordering can be found efficiently.
The exact ordering problem is to determine

$$\pi^* \in \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M}) \tag{6.112}$$

such that

$$\omega_{\mathrm{rec}}(P, \pi^*, \tau; \mathfrak{M}) = \omega_{\mathrm{rec}}(P, \tau; \mathfrak{M}). \tag{6.113}$$

A complete algorithmic analysis should distinguish the following questions.

1. Can the exact minimum width be computed in polynomial time for a specified uniformly encoded family?
2. Can one determine whether $\omega_{\mathrm{rec}} \leq k$ for a declared integer parameter k?
3. If exact minimization is computationally difficult, does there exist a polynomial-time approximation algorithm?
4. Are fixed-parameter algorithms possible with respect to $\widehat{\omega}_{\mathrm{rec}}$, $q(P)$, or another declared structural parameter?
5. Can useful orderings be constructed without first constructing the complete reachable prefix-state space or semantic quotient?
6. Can structural lower bounds on CPR-Width be obtained directly from the encoded instance?

6.21.2 Width Ratio

Let

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

be a declared admissible ordering.
Since

$$\widehat{\omega}_{\mathrm{rec}}(P, \tau; \mathfrak{M}) \in \mathbb{N}_0,$$

the denominator is always nonnegative. The ratio is therefore shifted by one in both numerator and denominator to avoid division by zero while preserving the exact optimality criterion.
Define the unnormalized width ratio by

$$\alpha_{\mathrm{width}}(P,\pi',\tau;\mathfrak{M}) = \frac{\widehat{\omega}_{\mathrm{rec}}(P,\pi',\tau;\mathfrak{M})+1}{\widehat{\omega}_{\mathrm{rec}}(P,\tau;\mathfrak{M})+1}. \tag{6.114}$$

The addition of one guarantees a positive denominator while preserving the exact optimality criterion.
The quantity α_{width} is a structural width ratio.
It is not an operation-count ratio and not a measured runtime overhead.

Proposition 6.4 (Width-Ratio Bound)**.** *For every admissible ordering π',*

$$\alpha_{\mathrm{width}}(P,\pi',\tau;\mathfrak{M}) \geq 1. \tag{6.115}$$

Moreover,

$$\alpha_{\mathrm{width}}(P,\pi',\tau;\mathfrak{M}) = 1 \tag{6.116}$$

if and only if

$$\widehat{\omega}_{\mathrm{rec}}(P,\pi',\tau;\mathfrak{M}) = \widehat{\omega}_{\mathrm{rec}}(P,\tau;\mathfrak{M}). \tag{6.117}$$

Proof. By the definition of the minimum unnormalized width,

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

Adding one to both sides gives

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

The denominator in Equation (6.114) is strictly positive.
Division therefore gives

$$\alpha_{\mathrm{width}}(P,\pi',\tau;\mathfrak{M}) \geq 1.$$

Equality holds exactly when the two unnormalized width values are equal.

□

6.21.3 Efficient Semantic-State Construction

At reconstruction stage i, the semantic quotient is

$$S_i(P,\pi,\tau;\mathfrak{M}) = U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})/\equiv_{P,\pi,i}^{\tau,\mathfrak{M}}. \tag{6.118}$$

Explicitly, $S_i(P,\pi,\tau;\mathfrak{M})$ is the set of equivalence classes of $U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})$ induced by $\equiv_{P,\pi,i}^{\tau,\mathfrak{M}}$; each element of $S_i(P,\pi,\tau;\mathfrak{M})$ is itself a semantic quotient class of reachable interface states, not a single interface state.
The primary semantic equivalence relation is

$$u \equiv_{P,\pi,i}^{\tau,\mathfrak{M}} v \quad\Longleftrightarrow\quad \widehat{\mathrm{Sig}}_{\tau,i}^{\pi}(u) = \widehat{\mathrm{Sig}}_{\tau,i}^{\pi}(v), \tag{6.119}$$

where

$$\widehat{\mathrm{Sig}}_{\tau,i}^{\pi} : U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M}) \longrightarrow \Sigma_{\tau,i}^{\pi} \tag{6.120}$$

is the induced reconstruction-safe interface-signature map.
The equivalence relation is therefore 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;
- it preserves conditions S1–S3;
- and the declared task requires that stronger information.

The principal algorithmic question is whether semantic quotient classes can be represented by a finite efficiently computable canonical encoding.

Let

$$\chi_i : U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M}) \longrightarrow \Gamma_i(P, \pi, \tau; \mathfrak{M}) \tag{6.121}$$

be a finite encoding map.

The symbol χ_i is used to avoid conflict with the stage-wise branching number λ_i introduced in Chapter 3 and with the admissible next-value sets $\Lambda_i^{\pi}(r)$.

The encoding is exact if

$$\chi_i(u) = \chi_i(v) \quad \Longleftrightarrow \quad u \equiv_{P,\pi,i}^{\tau,\mathfrak{M}} v. \tag{6.122}$$

Under this condition, the fibers of χ_i coincide exactly with the semantic quotient classes.

Consequently, there exists a canonical bijection

$$S_i(P, \pi, \tau; \mathfrak{M}) \cong \chi_i\left(U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M})\right). \tag{6.123}$$

The construction of polynomial-time exact encodings for nontrivial uniformly encoded families remains an open problem.

In addition, a tractability result requires that the encoding length of $\chi_i(u)$ itself is polynomially bounded in the original input size.

A one-sided condition must be interpreted carefully.

If

$$\chi_i(u) = \chi_i(v) \Longrightarrow u \equiv_{P,\pi,i}^{\tau,\mathfrak{M}} v, \tag{6.124}$$

then 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.

In that case,

$$\left|\chi_i\left(U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M})\right)\right| \geq |S_i(P, \pi, \tau; \mathfrak{M})|. \tag{6.125}$$

If the implication in Equation (6.124) fails, then the encoding may merge states whose reconstruction-safe future behavior differs.

Such a merge 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.

6.21.4 Width-Bound Slack

This bound applies for every finite problem instance P with per-component domain bound $q(P)$, active interface $B_i^{\pi,\tau}$, and finite reconstruction alphabet, all fixed as in Chapter 2.

The sharp operational Width–State Bound is

$$S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) \leq q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}. \tag{6.126}$$

Since

$$S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) \geq 1, \tag{6.127}$$

define the sharp width-bound slack factor by

$$\kappa_{\mathrm{bound}}(P, \pi, \tau; \mathfrak{M}) = \frac{q(P)^{\widehat{\omega}_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M})}}{S_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M})}. \tag{6.128}$$

Proposition 6.5 (Width-Bound Slack)**.** *For every finite problem instance, declared task, reconstruction model, and admissible ordering,*

$$\kappa_{\mathrm{bound}}(P, \pi, \tau; \mathfrak{M}) \geq 1. \tag{6.129}$$

Proof. By the Width–State Bound,

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

The denominator in Equation (6.128) is positive.

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.

6.21.5 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

$$G_P = \mathcal{G}(P) \tag{6.130}$$

be the explicitly declared graph representation associated with the problem instance P.

Depending on the problem class, G_P may be a primal graph, incidence graph, constraint graph, circuit graph, or another specified graph construction.

Let

$$\mathrm{tw}(G_P), \quad \mathrm{pw}(G_P), \quad \mathrm{bw}(G_P), \quad \mathrm{cw}(G_P) \tag{6.131}$$

denote treewidth, pathwidth, branchwidth, and clique-width.

Future research should investigate whether inequalities of the form

$$\omega_{\mathrm{rec}}(P, \tau; \mathfrak{M}) \leq f\left(\mathrm{tw}(G_P)\right) \tag{6.132}$$

or

$$\mathrm{tw}(G_P) \leq g\left(\omega_{\mathrm{rec}}(P, \tau; \mathfrak{M})\right) \tag{6.133}$$

hold for explicitly defined functions

$$f, g : \mathbb{N}_0 \longrightarrow \mathbb{N}_0. \tag{6.134}$$

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

1. the problem representation;
2. the declared task;
3. the reconstruction model;
4. the admissible-ordering family;
5. the active-interface rule;
6. the reconstruction-safe signature family;
7. and the graph or relational representation $\mathcal{G}(P)$.

The same underlying graph may induce different CPR-Width values for decision, counting, enumeration, optimization, or exact reconstruction tasks.

6.21.6 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.

Let

$$\mathcal{F}_{\mathrm{cand}} = \{P_x : x \in \{0,1\}^*\} \tag{6.135}$$

be a candidate uniformly encoded family, distinct from the family $\mathcal{F}$ fixed in Equation (6.102).

A complete CPR-tractability proof must provide one uniform algorithm producing

$$\pi_x \in \Pi_{\mathrm{adm}}(P_x, \tau; \mathfrak{M}) \tag{6.136}$$

and fixed polynomial bounds for

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

More precisely, there must exist fixed polynomials

$$p_j : \mathbb{N} \longrightarrow \mathbb{N}, \qquad j \in \mathcal{J}, \tag{6.137}$$

such that

$$N_j(P_x, \pi_x, \tau; \mathfrak{M}) \leq p_j(|x|) \tag{6.138}$$

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 Equation (6.106).

6.22 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.

6.22.1 Prototype Architecture

For a finite problem instance P, task τ, reconstruction model $\mathfrak{M}$, and ordering π, define the core structural and operation-count record by

$$\begin{aligned}\mathcal{R}_{\mathrm{core}}(P,\pi,\tau;\mathfrak{M}) = (&\widehat{\omega}_{\mathrm{rec}}, \omega_{\mathrm{rec}}, S_{\mathrm{CPR}}, N_{\mathrm{sem,state}}, E_{\mathrm{sem}}, m_{\mathrm{stage}},\\ &N_{\mathrm{Build}}, N_{\mathrm{Reduce}}, N_{\mathrm{Transition}}, N_{\mathrm{Reconstruct}}, N_{\mathrm{Verify}}, N_{\mathrm{Output}}).\end{aligned} \tag{6.139}$$

Define the total stage-indexed semantic-state count by

$$N_{\mathrm{sem,state}}(P,\pi,\tau;\mathfrak{M}) = \sum_{i=0}^{m_{\mathrm{stage}}(P,\pi)} |S_i(P,\pi,\tau;\mathfrak{M})|. \tag{6.140}$$

The quantity $N_{\mathrm{sem,state}}$ is a cumulative stage-wise count.

It must not be confused with

$$S_{\mathrm{CPR}},$$

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

Let

$$\mathcal{E}_i^{\mathrm{sem}}(P,\pi,\tau;\mathfrak{M})$$

denote the set of reachable labeled semantic quotient transitions from stage i to stage $i+1$.

Define the total semantic-transition count by

$$E_{\mathrm{sem}}(P,\pi,\tau;\mathfrak{M}) = \sum_{i=0}^{m_{\mathrm{stage}}(P,\pi)-1} \left|\mathcal{E}_i^{\mathrm{sem}}(P,\pi,\tau;\mathfrak{M})\right|. \tag{6.141}$$

If $m_{\mathrm{stage}}(P,\pi) = 0$, the sum is interpreted as the empty sum and therefore has value zero.

The quantities $N_{\mathrm{sem,state}}$ and E_{sem} are general model-relative counts.

The superscript "unif" used in Chapter 6 denotes their family-level specialization for the uniform ripple-adder transducer.

A complete measured implementation record should additionally include

$$\mathcal{R}_{\mathrm{measured}} = (T_{\mathrm{CPR}}, M_{\mathrm{peak}}, L_{\mathrm{Output}}, I_{\mathrm{CPR}}, H, S_{\mathrm{env}}, \mathcal{P}_{\mathrm{measure}}), \tag{6.142}$$

where

- T_{CPR} is the measured wall-clock time;
- M_{peak} is the peak memory use;

- L_{Output} is the encoded output length;
- I_{CPR} identifies the implementation;
- H identifies the hardware;
- S_{env} identifies the software environment;
- $\mathcal{P}_{\mathrm{measure}}$ identifies the measurement protocol.

All quantities must refer to the same representation, task, model, ordering, correctness requirement, and output requirement.

A prototype that reports only the reduced-state count or only the transition count does not test the complete CPR runtime framework.

6.22.2 Reference Execution Pipeline

A reproducible implementation should respect the non-circular construction order established in Chapters 2 and 3.

The reference execution pipeline is:

1. Parse and validate the encoded problem instance.
2. Construct the complete initial CPR representation.
3. Construct or read an admissible reconstruction ordering.
4. Generate the processed-prefix and unresolved-suffix sequence.
5. Generate the reachable prefix-state spaces under the declared reachability and transition rules.
6. Compute, retrieve, or certify the reconstruction-safe prefix signatures.
7. Determine the family of task-sufficient interfaces at every stage.
8. Select a cardinality-minimal task-sufficient interface whenever the exact CPR-Width is required.
9. Construct the raw interface-state spaces.
10. Project reachable prefix states onto the selected interfaces.
11. Construct the induced interface-signature maps.
12. Form the semantic quotients induced by the reconstruction-safe signatures.
13. Construct and process the semantic quotient transitions.
14. Perform task-relevant reconstruction.
15. Verify soundness and completeness for the declared task.
16. Materialize the required output.
17. Record all structural quantities, operation counts, memory use, and measured runtime data.

The semantic quotient in Step 12 is not generally a future-extension-set quotient.

It is the quotient induced by the reconstruction-safe signature.

For every stage i, define the stage record

$$\begin{aligned}\mathcal{A}_i(P,\pi,\tau;\mathfrak{M}) = (&b_i^{\pi,\tau}(P), |U_i(P,\pi,\tau;\mathfrak{M})|,\\ &|U_i^{\mathrm{reach}}(P,\pi,\tau;\mathfrak{M})|, |S_i(P,\pi,\tau;\mathfrak{M})|).\end{aligned} \tag{6.143}$$

The complete stage sequence is

$$\mathcal{A}(P,\pi,\tau;\mathfrak{M}) = \big(\mathcal{A}_0,\mathcal{A}_1,\ldots,\mathcal{A}_{m_{\mathrm{stage}}(P,\pi)}\big). \tag{6.144}$$

This record separates

- interface width;
- raw interface-state growth;
- reachability filtering;
- semantic quotienting;
- transition cost;
- reconstruction cost;
- verification cost;
- and output cost.

6.22.3 Comparison with Classical Baselines

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}.
Let

$$T_{\mathrm{Classic}}(P, \tau; I_{\mathrm{Classic}})$$

denote the measured wall-clock time of a declared classical implementation.
Let

$$T_{\mathrm{CPR}}(P, \pi, \tau; I_{\mathrm{CPR}})$$

denote the measured wall-clock time of the declared CPR implementation.
Assume

$$N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M}) > 0. \tag{6.145}$$

The count-level gain is

$$G_{\mathrm{count}}(P, \pi, \tau; \mathfrak{M}, A_{\mathrm{Classic}}) = \frac{N_{\mathrm{Classic}}(P, \tau; A_{\mathrm{Classic}})}{N_{\mathrm{CPR}}(P, \pi, \tau; \mathfrak{M})}. \tag{6.146}$$

Assume also

$$T_{\mathrm{CPR}}(P, \pi, \tau; I_{\mathrm{CPR}}) > 0. \tag{6.147}$$

The measured wall-clock gain is

$$G_{\mathrm{time}}(P, \pi, \tau; I_{\mathrm{CPR}}, I_{\mathrm{Classic}}) = \frac{T_{\mathrm{Classic}}(P, \tau; I_{\mathrm{Classic}})}{T_{\mathrm{CPR}}(P, \pi, \tau; I_{\mathrm{CPR}})}. \tag{6.148}$$

The two gains remain distinct:

$$G_{\mathrm{count}} > 1 \not\Longrightarrow G_{\mathrm{time}} > 1. \tag{6.149}$$

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.

6.22.4 Boundary-Condition Audit

The CPR audit must distinguish finite instance records from uniform family-level certification.

Instance-Level Record

For one finite benchmark instance P, define the instance-level audit record

$$\mathcal{A}_{\mathrm{BC}}^{\mathrm{inst}}(P) = (\sigma_1^{\mathrm{inst}}(P), \sigma_2^{\mathrm{inst}}(P), \sigma_3^{\mathrm{inst}}(P), \sigma_4^{\mathrm{inst}}(P)). \tag{6.150}$$

Each component takes one of the values

$$\{\text{verified}, \text{failed}, \text{undetermined}\}. \tag{6.151}$$

The instance-level interpretation is:

BC1 evidence.
Whether the required structural objects are explicitly available for the instance.
BC2 evidence.
Whether the finite construction process and its measured operation count are completely recorded for the instance.
BC3 evidence.
Whether soundness and completeness are verified for the declared finite task and benchmark.
BC4 evidence.
Whether the complete stage-indexed semantic-state and transition record is available for the instance.

An instance-level record does not prove polynomial family behavior.

Family-Level Certification

For a uniformly encoded family $\mathcal{F}$, define

$$\mathcal{A}_{\mathrm{BC}}^{\mathrm{fam}}(\mathcal{F}) = (\sigma_1^{\mathrm{fam}}(\mathcal{F}), \sigma_2^{\mathrm{fam}}(\mathcal{F}), \sigma_3^{\mathrm{fam}}(\mathcal{F}), \sigma_4^{\mathrm{fam}}(\mathcal{F})). \tag{6.152}$$

Define the audit-status alphabet

$$\Sigma_{\mathrm{audit}} = \{\text{proved}, \text{disproved}, \text{open}\}. \tag{6.153}$$

Each certification component satisfies

$$\sigma_k^{\mathrm{fam}}(\mathcal{F}) \in \Sigma_{\mathrm{audit}}, \qquad k \in \{1, 2, 3, 4\}.$$

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.

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 cannot be certified as CPR-tractable by this framework:

$$\exists k \in \{1, 2, 3, 4\} : \sigma_k^{\mathrm{fam}}(\mathcal{F}) \neq \text{proved} \Longrightarrow \mathcal{F} \text{ cannot be certified as CPR-tractable by this framework.} \tag{6.154}$$

6.22.5 Benchmark Families

A future benchmark corpus should contain favorable, neutral, and unfavorable instances.

The purpose is not to select only examples with visibly small candidate interfaces.

The benchmark design should 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

1. Boolean satisfiability instances with controlled clause-interaction structure;
2. finite CSP instances with varying domain size, constraint arity, and primal-graph structure;
3. graph-coloring instances with varying density and separator structure;
4. arithmetic systems including full adders, ripple adders, and larger finite circuit blocks;
5. tensor-structured systems with explicitly defined reconstruction components and active-index interfaces;
6. routing and scheduling systems with bounded and unbounded interaction boundaries;
7. coding and decoding systems with controlled trellis structure;
8. negative-control instances designed to produce large active interfaces or expensive semantic signatures.

For each benchmark family, the following objects must be documented:

- instance-generation procedure;
- input encoding;
- task;
- reconstruction model;
- admissibility rules;
- output requirements;
- baseline algorithms;
- and correctness criteria.

6.22.6 Ordering Ablation Studies

Because CPR-Width is ordering-dependent, experiments should compare multiple admissible orderings while keeping every other model component fixed.

Let

$$\Pi_{\mathrm{test}}(P, \tau; \mathfrak{M}) = \{\pi_1, \ldots, \pi_r\} \subseteq \Pi_{\mathrm{adm}}(P, \tau; \mathfrak{M}) \tag{6.155}$$

be the finite set of tested orderings.

For each ordering π_j, record

$$\begin{aligned} \mathcal{O}(P, \pi_j, \tau; \mathfrak{M}) = (&\widehat{\omega}_{\mathrm{rec}}(P, \pi_j, \tau; \mathfrak{M}), \\ &\omega_{\mathrm{rec}}(P, \pi_j, \tau; \mathfrak{M}), \\ &S_{\mathrm{CPR}}(P, \pi_j, \tau; \mathfrak{M}), \\ &N_{\mathrm{CPR}}(P, \pi_j, \tau; \mathfrak{M}), \\ &T_{\mathrm{CPR}}(P, \pi_j, \tau; I_{\mathrm{CPR}})). \end{aligned} \tag{6.156}$$

Only the ordering may vary.

The following objects must remain fixed:

- problem representation;
- task;
- reconstruction model;
- signature family;
- implementation;
- hardware;

- software environment;
- output requirement;
- and measurement protocol.

A smaller width need not imply a smaller measured runtime:

$$\omega_{\mathrm{rec}}(P,\pi_a,\tau;\mathfrak{M}) < \omega_{\mathrm{rec}}(P,\pi_b,\tau;\mathfrak{M}) \not\Longrightarrow T_{\mathrm{CPR}}(P,\pi_a,\tau;I_{\mathrm{CPR}}) < T_{\mathrm{CPR}}(P,\pi_b,\tau;I_{\mathrm{CPR}}). \tag{6.157}$$

6.22.7 Experimental Comparison with Classical Width Parameters

For graph-based or hypergraph-based benchmarks, the experimental record should include all available classical width parameters of the declared representation $G_P = \mathcal{G}(P)$.
Define

$$\begin{aligned}\mathcal{W}(P,\pi,\tau;\mathfrak{M},\mathcal{G}) = (&\omega_{\mathrm{rec}}(P,\pi,\tau;\mathfrak{M}),\\ &\mathrm{tw}(G_P),\mathrm{pw}(G_P),\mathrm{bw}(G_P),\mathrm{cw}(G_P)).\end{aligned} \tag{6.158}$$

The purpose of the record is not to infer a theorem from empirical correlation.
It is intended to identify

- candidate upper bounds;
- candidate lower bounds;
- separation examples;
- ordering effects;
- and representation dependence.

6.22.8 Evidence Classification and Admissible Conclusions

CPR-WIDTH results should be classified by the specific evidence categories supporting them.

The categories are not a single linear hierarchy.

The evidence categories are mutually non-exclusive.

A mathematical result may simultaneously belong to several categories, depending on the type of justification provided.

The following classification is used.

E0 — Definition Evidence.
A symbol, object, relation, operator, or structural quantity is introduced formally.
E1 — Finite Verification Evidence.
A claim is checked exhaustively or directly on one explicitly stated finite example.
E2 — Structural Pattern Evidence.
A recurring structural pattern is identified across several models or examples.
E3 — Operation-Count Evidence.
A finite or symbolic elementary-operation comparison is established relative to a declared baseline and accounting convention.
E4 — Asymptotic Theorem Evidence.
A theorem is proved uniformly for an encoded family under explicit assumptions and boundary conditions.
E5 — Measured Runtime Evidence.
A wall-clock claim is supported by an implementation, hardware description, software environment, benchmark protocol, and reproducible measurements.

The categories satisfy the non-implications

$$\mathrm{E1} \not\Longrightarrow \mathrm{E4}, \tag{6.159}$$

$$\mathrm{E3} \not\Longrightarrow \mathrm{E5}, \tag{6.160}$$

$$\mathrm{E4} \not\Longrightarrow \mathrm{E5}, \tag{6.161}$$

and

$$\mathrm{E5} \not\Longrightarrow \mathrm{E4}. \tag{6.162}$$

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.

Measured performance on a benchmark corpus does not establish a uniform asymptotic theorem.

Every result should therefore state its complete evidence profile.

For example, a result may simultaneously possess E1, E3, and E5 support while having no E4 theorem.

6.22.9 Future Research Sequence

A disciplined CPR-WIDTH research program may proceed through the following stages.

1. Complete the exact mathematical characterization of CPR-Width for selected finite models.
2. Implement exact reachable-prefix, signature, interface, and semantic-quotient construction.
3. Record BC1–BC4 evidence at the instance level.
4. Prove BC2 and BC4 at the uniform family level whenever an asymptotic tractability claim is intended.
5. Publish complete component-wise operation-count records.
6. Compare multiple admissible reconstruction orderings while keeping all other model and implementation conditions fixed.
7. Compare CPR-WIDTH with declared classical baselines.
8. Extend finite analysis to uniformly encoded families.
9. Derive formal relations among interface width, semantic-state growth, transition structure, and complete operation count.
10. Investigate exact and approximate relations to classical width parameters.
11. Determine controlled or constant CPR-Width bounds for additional problem-class subclasses.
12. Conduct 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.

It also prevents structural state reduction from being interpreted as a complete runtime proof.

6.23 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

$$\boldsymbol{\omega}_{\mathbf{rec}}(\boldsymbol{P}, \boldsymbol{\pi}, \boldsymbol{\tau}; \mathfrak{M}) \longrightarrow \boldsymbol{S}_{\mathbf{CPR}}(\boldsymbol{P}, \boldsymbol{\pi}, \boldsymbol{\tau}; \mathfrak{M}) \longrightarrow \boldsymbol{N}_{\mathbf{CPR}}(\boldsymbol{P}, \boldsymbol{\pi}, \boldsymbol{\tau}; \mathfrak{M}). \tag{6.163}$$

The first arrow expresses the influence of the active reconstruction interface on the raw and semantic state spaces.

The second arrow expresses the contribution of semantic-state processing to the complete CPR operation count.

Neither arrow is a universal tractability theorem.

The principal unresolved questions concern

- efficient construction of favorable orderings;
- exact polynomial-time semantic encodings;
- efficient construction of minimum task-sufficient interfaces;
- 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

$$\mathrm{BC}_1 \wedge \mathrm{BC}_2 \wedge \mathrm{BC}_3 \wedge \mathrm{BC}_4.$$

It is the full uniform quantified requirement stated in Equation (6.106).

A polynomial-time conclusion is permitted only after one fixed decision task, one fixed uniform reconstruction model, one deterministic preprocessing algorithm, one deterministic ordering algorithm, one uniform correctness specification, and six fixed polynomial component bounds have been established for every encoded instance.

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, all four boundary conditions, and polynomial control of all six cost components.

Measured acceleration requires a separate reproducible implementation and benchmark protocol.

Future work will focus on exact complexity classifications of the CPR-Width minimization problem, fixed-parameter algorithms with respect to CPR-Width and related structural parameters, approximation guarantees when exact minimization is computationally difficult, and formal relationships between CPR-Width and established graph-width parameters such as treewidth, pathwidth, and clique-width.

Thus, CPR-WIDTH remains a formal framework for controlled partial reconstruction, semantic-state compression, and width-based runtime analysis.

CPR-WIDTH therefore provides a reconstruction-oriented structural framework whose ultimate algorithmic significance depends on future mathematical characterization, efficient implementation, and verification on uniformly encoded problem families.

Appendix A
Core Symbol Register

This appendix records the main symbols used throughout the manuscript. The purpose is to provide a compact reference for the CPR-WIDTH notation.

Table A.1: Core Symbol Register for CPR-WIDTH.

Symbol	Meaning
P	Finite computational problem or finite problem instance.
$V(P)$	Finite set of elementary components of P.
$v_1, \dots, v_n$	Components of the finite problem representation.
π	Admissible reconstruction ordering.
$v_{\pi(i)}$	Component processed at reconstruction stage i.
V_i^{π}	Processed prefix after stage i.
$\overline{V}_i^{\pi}$	Unresolved suffix after stage i.
$\ell(P)$	Encoding length or input length of P.
B_i^{π}	Active reconstruction interface at stage i.
$B(P,\pi)$	Complete interface sequence induced by π.
$\omega_{\mathrm{rec}}(P,\pi)$	Controlled Partial Reconstruction Width induced by π.
$\omega_{\mathrm{rec}}(P)$	Minimum Controlled Partial Reconstruction Width over admissible orderings.
$\Pi_{\mathrm{adm}}(P)$	Set of admissible reconstruction orderings of P.
D_v	Finite local domain of component v.
$q(P)$	Maximum local domain size of P.
$\mathcal{U}_i(P,\pi)$	Raw interface-state space at stage i.
$\mathcal{U}_i^{\mathrm{reach}}(P,\pi)$	Reachable interface-state space at stage i.
$\mathcal{E}_i(u)$	Admissible future-continuation set of a reachable state u; task-independent.
$\Theta_{P,i}^{\tau}$	Task-evaluation map; extracts the task-relevant information from a continuation set.
$\nu_{P,i}^{\tau}$	Signature-normalization map; assigns a canonical representative to a task-evaluation value.
$\widehat{\sigma}_{P,i}^{\tau}(u)$	Normalized future signature of u: $\widehat{\sigma}_{P,i}^{\tau}(u) = \nu_{P,i}^{\tau}(\Theta_{P,i}^{\tau}(\mathcal{E}_i(u)))$.
$\equiv_{P,i}^{\tau}$	Task-relative signature equivalence: $u \equiv_{P,i}^{\tau} v \iff \widehat{\sigma}_{P,i}^{\tau}(u) = \widehat{\sigma}_{P,i}^{\tau}(v)$.
$\mathcal{S}_i(P,\pi,\tau)$	Semantic quotient state space at stage i.
$S_{\mathrm{CPR}}(P,\pi,\tau)$	Maximum semantic-state count of the reconstruction process.
N_{CPR}	Total CPR operation count.
N_{Build}	Operations required to build the initial representation.
N_{Reduce}	Operations required for reduction and semantic quotienting.
$N_{\mathrm{Transition}}$	Operations required for stage-to-stage transitions.
$N_{\mathrm{Reconstruct}}$	Operations required to reconstruct admissible outputs.
N_{Verify}	Operations required to verify reconstructed outputs.
N_{Output}	Operations required to produce final outputs.
G_{count}	Elementary-operation-count gain relative to a declared classical baseline.
G_{time}	Time gain relative to a declared classical runtime baseline.
$\mathcal{F}$	Uniformly encoded family of finite problem instances.
$L_{\mathcal{F}}$	Decision language associated with the family $\mathcal{F}$.

Continued on the next page

Symbol	Meaning
$\mathsf{P}_{\mathrm{CPR}}$	Class of languages certified by the CPR-WIDTH audit conditions.
P	Classical polynomial-time decision class.
NP	Classical nondeterministic polynomial-time decision class.

Appendix B
Core Definition Register

This appendix summarizes the main definitions used in the manuscript.

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

Definition / Object	Formula	Meaning
Finite component set	$V(P) = \{v_1, \dots, v_n\}$	Finite set of elementary components of the problem instance P.
Admissible ordering	$\pi = (v_{\pi(1)}, \dots, v_{\pi(n)})$	Order in which the finite components are processed.
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.
Active reconstruction interface	$B_i^{\pi,\tau} \in \arg\min_{B \in \mathcal{B}_i^{\pi,\tau}(P)} \lvert B\rvert$, where $\mathcal{B}_i^{\pi,\tau}(P) = \{B \subseteq V_i^{\pi} : \forall r, r' \in R_i^{\pi}(P),\ r\lvert_B = r'\lvert_B \Rightarrow \mathrm{Sig}_{\tau,i}^{\pi}(r) = \mathrm{Sig}_{\tau,i}^{\pi}(r')\}$ (Def. 2.5, Def. 2.6)	A cardinality-minimal task-sufficient reconstruction interface: the smallest set of processed components on which the reconstruction-safe signature $\mathrm{Sig}_{\tau,i}^{\pi}$ still factors.
Interface sequence	$B(P, \pi, \tau) = (B_0^{\pi,\tau}, \dots, B_n^{\pi,\tau})$	Complete sequence of selected active interfaces induced by the ordering π and task τ.
Controlled Partial Reconstruction Width	$b_i^{\pi,\tau}(P) = \min\{\lvert B\rvert : B \in \mathcal{B}_i^{\pi,\tau}(P)\}$ (Def. 2.6); $\widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = \max_{0 \le i \le n} b_i^{\pi,\tau}(P)$ (Def. 2.7); $\omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) = \max\{0, \widehat{\omega}_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M}) - 1\}$ (Def. 2.8)	Ordering-dependent normalized maximum size of the selected cardinality-minimal task-sufficient interface, taken over all reconstruction stages.
Minimum CPR-Width	$\omega_{\mathrm{rec}}(P, \tau; \mathfrak{M}) = \min_{\pi \in \Pi_{\mathrm{adm}}(P,\tau;\mathfrak{M})} \omega_{\mathrm{rec}}(P, \pi, \tau; \mathfrak{M})$ (Def. 2.9); shorthand $\omega_{\mathrm{rec}}(P)$ when $\tau, \mathfrak{M}$ are fixed	Smallest CPR-Width attainable over all admissible reconstruction orderings, for the declared task and model.
Raw interface-state space	$\mathcal{U}_i(P, \pi) = \prod_{v \in B_i^{\pi}} D_v$	All possible assignments to the active interface at stage i.
Maximum local domain size	$q(P) = \max_{v \in V(P)} \lvert D_v\rvert$	Largest local domain size among the components of P.
Reachable interface states	$\mathcal{U}_i^{\mathrm{reach}}(P, \pi) \subseteq \mathcal{U}_i(P, \pi)$	Raw states that can occur under the declared reconstruction rules.

Continued on the next page

Definition / Object	**Formula**	**Meaning**
Future-continuation set	$\mathcal{E}_i(u)$	Admissible future-continuation set of a reachable state u; task-independent (Def. 2.10).
Task-evaluation map	$\Theta^{\tau}_{P,i}(\mathcal{E}_i(u))$	Extracts the task-relevant information from the continuation set of u (Def. 2.11).
Signature normalization	$\nu^{\tau}_{P,i}$	Assigns a canonical representative to a task-evaluation value (Def. 2.12).
Normalized future signature	$\widehat{\sigma}^{\tau}_{P,i}(u) = \nu^{\tau}_{P,i}(\Theta^{\tau}_{P,i}(\mathcal{E}_i(u)))$	Canonical task-relevant future description of u (Def. 2.13).
Task-relative signature equivalence	$u \equiv^{\tau}_{P,i} v \iff \widehat{\sigma}^{\tau}_{P,i}(u) = \widehat{\sigma}^{\tau}_{P,i}(v)$	Two reachable states are equivalent iff their normalized task signatures coincide. Equality of the full continuation sets $\mathcal{E}_i(u) = \mathcal{E}_i(v)$ is required only for the enumeration task (Prop. 2.4); decision needs only existence, counting only cardinality, optimization only the declared certificate (Def. 2.14).
Semantic quotient	$\mathcal{S}_i(P,\pi,\tau) = \mathcal{U}_i^{\mathrm{reach}}(P,\pi)/\equiv^{\tau}_{P,i}$	State space obtained by merging reachable states with identical task-relevant futures.
Maximum semantic-state count	$S_{\mathrm{CPR}}(P,\pi,\tau) = \max_{0\le i\le n} \lvert\mathcal{S}_i(P,\pi,\tau)\rvert$	Maximum number of semantic quotient states over all reconstruction stages.
Binary semantic-state dimension	$D_{\mathrm{CPR}}(P,\pi) = \lceil \log_2 \max\{1, S_{\mathrm{CPR}}(P,\pi)\} \rceil$	Number of bits required to identify one semantic state.
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 operation-count model of the CPR process.
Post-transition cost	$R_{\mathrm{post}} = N_{\mathrm{Reconstruct}} + N_{\mathrm{Verify}} + N_{\mathrm{Output}}$	Combined reconstruction, verification, and output cost.
Count-level gain	$G_{\mathrm{count}}(P,\pi) = \dfrac{N_{\mathrm{Classic}}(P)}{N_{\mathrm{CPR}}(P,\pi)}$	Operation-count comparison relative to a declared classical baseline.
Measured wall-clock gain	$G_{\mathrm{time}}(P,\pi) = \dfrac{T_{\mathrm{Classic}}(P)}{T_{\mathrm{CPR}}(P,\pi)}$	Measured runtime comparison relative to a declared implementation baseline.
Certified CPR class	$L_{\mathcal{F}} \in \mathsf{P}_{\mathrm{CPR}}$	The language is certified by a complete CPR audit model with polynomial operation count.
CPR tractability requirement	$\mathrm{BC1} \wedge \mathrm{BC2} \wedge \mathrm{BC3} \wedge \mathrm{BC4} \wedge N_{\mathrm{CPR}}(P_x,\pi_x) \in \mathrm{poly}(\ell(P_x))$	Complete condition required before a polynomial-time CPR conclusion is permitted.

Appendix C
Quotient Preservation

Let

$$u, v \in U_i^{\mathrm{reach}}(P, \pi, \tau; \mathfrak{M})$$

and assume that

$$u \equiv_{P,\pi,i}^{\tau,\mathfrak{M}} v. \tag{C.1}$$

By definition,

$$\widehat{\mathrm{Sig}}_{\tau,i}^{\pi}(u) = \widehat{\mathrm{Sig}}_{\tau,i}^{\pi}(v). \tag{C.2}$$

Choose reachable prefix representatives

$$r, r' \in \mathcal{R}_i^{\pi}(P)$$

such that

$$\rho_i^{\pi,\tau}(r) = u, \qquad \rho_i^{\pi,\tau}(r') = v.$$

The induced-signature factorization gives

$$\mathrm{Sig}_{\tau,i}^{\pi}(r) = \mathrm{Sig}_{\tau,i}^{\pi}(r'). \tag{C.3}$$

Condition S1 implies

$$\mathrm{Ans}_{\tau,i}^{\pi}(r) = \mathrm{Ans}_{\tau,i}^{\pi}(r'). \tag{C.4}$$

Condition S2 implies

$$\Lambda_i^{\pi}(r) = \Lambda_i^{\pi}(r'). \tag{C.5}$$

For every common admissible next value

$$a \in \Lambda_i^{\pi}(r) = \Lambda_i^{\pi}(r'),$$

condition S3 gives

$$\mathrm{Sig}_{\tau,i+1}^{\pi}(\delta_i^{\pi}(r, a)) = \mathrm{Sig}_{\tau,i+1}^{\pi}(\delta_i^{\pi}(r', a)). \tag{C.6}$$

Therefore, the projected successors belong to the same semantic class at stage $i + 1$.

Hence, quotienting preserves

- the declared task-relative future value;
- the admissible next-value set;
- the semantic class of every admissible successor;
- and the well-definedness of the quotient transition.

Literal equality of complete continuation sets is not required.

Appendix D
Audit Checklist

This appendix provides a compact checklist for applying the CPR-WIDTH audit model. The checklist records which declarations and boundary conditions must be present before a reconstruction process can be certified inside the CPR-WIDTH framework.

Table D.1: CPR-WIDTH audit checklist.

Audit item	**Required declaration or check**
Instance declaration	Declare the finite problem instance P, the component set $V(P)$, the local domains D_v, the encoding length $\ell(P)$, the declared task τ, and the intended output type.
Task declaration	Specify whether the task is decision, counting, optimization, enumeration, or another explicitly stated reconstruction objective.
Reconstruction ordering	Declare an admissible reconstruction ordering π. The ordering must be part of the audit because the width value depends on it.
Processed and unresolved parts	Declare the processed prefixes V_i^π and unresolved suffixes $\overline{V}_i^\pi$ for the reconstruction stages.
Active interfaces	Declare the active interfaces B_i^π and the full interface sequence $B(P,\pi)$.
Width declaration	Record the induced width $\omega_{\mathrm{rec}}(P,\pi) = \max\Big\{0, \max_{0\le i\le n} \lvert B_i^\pi\rvert - 1\Big\}$. If a bounded-width claim is made, the intended bound must be stated explicitly.
Raw state spaces	Declare the raw interface-state spaces $\mathcal{U}_i(P,\pi)$.
Reachable state spaces	Declare the reachable interface-state spaces $\mathcal{U}_i^{\mathrm{reach}}(P,\pi)$. Declare the future-continuation sets $\mathcal{E}_i(u)$ (task-independent, Def. 2.10), the task-evaluation map $\Theta_{P,i}^\tau$, and the normalized future signature $\widehat{\sigma}_{P,i}^\tau(u)$.
Semantic equivalence	Declare the equivalence relation $u \equiv_{P,i}^\tau v \iff \widehat{\sigma}_{P,i}^\tau(u) = \widehat{\sigma}_{P,i}^\tau(v)$. Literal equality of the continuation sets $\mathcal{E}_i(u) = \mathcal{E}_i(v)$ is required only for the enumeration task (Prop. 2.4); decision needs only existence, counting only cardinality, optimization only the declared certificate.
Semantic quotient	Declare the semantic quotient spaces $\mathcal{S}_i(P,\pi,\tau)$.
Semantic-state count	Record $S_{\mathrm{CPR}}(P,\pi,\tau) = \max_{0\le i\le n} \lvert\mathcal{S}_i(P,\pi,\tau)\rvert$.
Operation-count components	Declare N_{Build}, N_{Reduce}, $N_{\mathrm{Transition}}$, $N_{\mathrm{Reconstruct}}$, N_{Verify}, and N_{Output}.
Complete operation count	Record the complete cost model $N_{\mathrm{CPR}} = N_{\mathrm{Build}}+N_{\mathrm{Reduce}}+N_{\mathrm{Transition}}+N_{\mathrm{Reconstruct}}+N_{\mathrm{Verify}}+N_{\mathrm{Output}}$.
Polynomial bound check	For every component $j \in \{\mathrm{Build}, \mathrm{Reduce}, \mathrm{Transition}, \mathrm{Reconstruct}, \mathrm{Verify}, \mathrm{Output}\}$ there must exist a polynomial p_j such that $N_j(P,\pi,\tau) \le p_j(\ell(P))$.
Soundness check	Verify that every reconstructed output is valid for the declared task: $y \in \mathrm{Output}_{\mathrm{CPR}}(P,\pi,\tau) \implies \mathrm{Valid}_\tau(P,y)$.

Continued on the next page

Audit item	**Required declaration or check**
Completeness check	Verify that no required task-relevant solution is lost. For enumeration, one may require $\mathrm{Sol}_\tau(P) \subseteq \mathrm{Output}_{\mathrm{CPR}}(P, \pi, \tau)$. For decision tasks, one may require $\mathrm{Sol}_\tau(P) \neq \varnothing \implies \mathrm{CPRDecision}(P, \pi, \tau) = 1$.
Certification status	A positive answer to all required declarations and checks is needed for CPR-WIDTH certification.
Non-certification status	If any required item is unanswered, undetermined, or refuted, the audit does not certify the reconstruction process. This is an audit status, not a classical complexity conclusion.

The checklist is a control instrument. It does not prove tractability by itself. A polynomial-time conclusion requires all declared structural, semantic, correctness, and operation-count requirements to be satisfied for the relevant uniformly encoded family.

Bibliography

[1] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[2] Hans L. Bodlaender. "A Partial k-Arboretum of Graphs with Bounded Treewidth". In: *Theoretical Computer Science* 209.1–2 (1998), pp. 1–45.

[3] Stephen A. Cook. "The Complexity of Theorem-Proving Procedures". In: *Proceedings of the Third Annual ACM Symposium on Theory of Computing* (1971), pp. 151–158.

[4] Marek Cygan et al. *Parameterized Algorithms*. Springer, 2015.

[5] Adnan Darwiche and Pierre Marquis. "A Knowledge Compilation Map". In: *Journal of Artificial Intelligence Research* 17 (2002), pp. 229–264.

[6] Rina Dechter. *Constraint Processing*. Morgan Kaufmann, 2003.

[7] Reinhard Diestel. *Graph Theory*. 5th ed. Springer, 2017.

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

[9] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[10] David Money Harris and Sarah L. Harris. *Digital Design and Computer Architecture*. 2nd ed. Morgan Kaufmann, 2012.

[11] Richard M. Karp. "Reducibility Among Combinatorial Problems". In: *Complexity of Computer Computations* (1972), pp. 85–103.

[12] Tamara G. Kolda and Brett W. Bader. "Tensor Decompositions and Applications". In: *SIAM Review* 51.3 (2009), pp. 455–500.

[13] Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

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

[15] Francesca Rossi, Peter van Beek, and Toby Walsh, eds. *Handbook of Constraint Programming*. Elsevier, 2006.

[16] Michael Sipser. *Introduction to the Theory of Computation*. 3rd ed. Cengage Learning, 2012.