What Survives Reconstruction

Admissibility, Irreversibility, and the Architecture of Externalized Structure

Flyxion

July 2026

Abstract

In March 2026, the complete source of Anthropic's Claude Code CLI was exposed through a misconfigured build artifact, revealing that a system widely understood as a single neural inference engine was in fact a sprawling assemblage of deterministic parsers, permission systems, orchestration loops, caches, and logs surrounding a language model. A public debate followed: did the presence of conventional, non-neural machinery inside the system undermine the claim that the model exhibits genuine understanding of the code it manipulates?

This monograph argues that the debate, as conducted, asked the wrong question. The interesting fact about the leak is not that symbolic and statistical computation coexist — this was never in serious doubt, and a straightforward appeal to the autonomy of syntax from semantics disposes of the naive inference from "there is a parser" to "there is no understanding." The interesting fact is that some of what leaked cannot be explained by the ordinary economics of caching at all.

We distinguish two reasons a computational system materializes structure rather than continuously re-deriving it. The first, reconstruction-cost materialization, is the familiar engineering tradeoff: a structure is kept because recomputing it is more expensive than storing it, and this tradeoff is contingent on available hardware, algorithms, and workload. The second, irreversibility materialization, is not a tradeoff at all: a structure is kept because, absent that structure, a prior state of the system's history becomes permanently unreachable from its current state. No improvement in computational speed dissolves this second kind of necessity, because it is not a claim about cost but a claim about the topology of a system's reachable history.

We formalize this distinction using the admissibility and repair machinery developed for the HYDRA architecture, in which explicit distinction projections — inspectable intermediate representations rather than inferred latent structure — are continuously checked against incoming evidence and revised under failure. This gives the distinction a falsifiable form: components of a system whose absence would render certain histories unreachable should appear in the system's architecture regardless of hardware trends, while components that merely save recomputation should track the cost curve and could in principle disappear

as computation gets cheaper.

We then use the Claude Code leak as a case study, sorting its exposed components against this criterion, and return with a sharpened version of the transformer semantics question that motivated the original public debate: not whether a Logical-Form-like object exists inside a language model's latent space, but whether the architecture surrounding the model tracks the reversible/irreversible boundary of its own operation, and what that tracking would look like if it were present.

Contents

Part I

The Question

Chapter 1

Introduction

On March 31, 2026, a build artifact belonging to Anthropic's Claude Code CLI was published to the npm registry with a source map attached that should not have shipped. Within hours, the complete TypeScript source of the tool — on the order of half a million lines — had been reconstructed, mirrored, and picked apart by thousands of developers. Anthropic characterized the event as a packaging error rather than a security breach; whatever its cause, its effect was to turn a production coding agent into a public artifact that could be read line by line.

What readers found was not a single neural network. It was a heterogeneous system: a custom terminal renderer, dozens of purpose-built tools, a permission and approval layer, a background memory-consolidation process, context-compaction logic, multi-agent orchestration, and — the detail that set off the argument this monograph responds to — deterministic, hand-written parsing infrastructure sitting alongside the language model rather than inside it.

1.1 The Inference and the Rebuttal

A line of public commentary treated this discovery as damaging.[1] The argument, stripped to its premise, ran roughly as follows: if a system's competence at a task is genuinely produced by a neural network's understanding of that task, then the network's forward pass should be doing the relevant work end to end. Finding conventional, non-learned machinery performing a core part of the task — here, parsing source code into a structured representation

[1]The argument this section reconstructs appears to originate with a LinkedIn post by Vincent Lextrait, Behind Claude Code's Curtain: Generative Grammars Doing Heavy Lifting (2026), subsequently amplified by Gary Marcus among others. The original post's exact wording could not be independently verified for this monograph, since LinkedIn does not permit automated retrieval of its content; the reconstruction here follows the summary and framing given in the rebuttal cited below, which quotes and responds to it directly.

— was taken as evidence that the appearance of competence was manufactured elsewhere, that the model was, in the language the argument reached for, a wizard behind a curtain.

The rebuttal to this argument is not difficult to state and does not require any special theoretical apparatus.[2] It requires only noticing that the premise is false. Nothing about genuine competence at a task implies that a system exhibiting that competence must perform every subcomputation involved in the task using the same mechanism. Database systems are not held to fail at querying because they use B-trees rather than learned indexes. Satisfiability solvers are not held to fail at reasoning because they use clause learning rather than gradient descent. The premise smuggled into the original argument — that real understanding is recognizable by its refusal to decompose into specialized subsystems — has no support in the history of computing, and arguably runs backward: decomposition into specialized subsystems is the normal signature of a mature engineering solution to a hard problem, not evidence against the sophistication of the whole.

A more careful version of the rebuttal draws on the distinction, central to generative linguistics since the middle of the twentieth century, between syntax and semantics as separately characterizable computational problems. Chomsky's demonstration that a sentence can be syntactically well-formed while semantically anomalous — colorless green ideas sleep furiously is the canonical example — establishes that grammaticality is not parasitic on meaning. Structural computation can and does proceed by its own rules, producing representations over which semantic interpretation is subsequently defined. On this view, finding a parser inside a language model's surrounding architecture is not merely unsurprising; it is close to what one would predict. Syntax is exactly the kind of computation that admits an autonomous, structurally exact treatment independent of whatever is happening at the level of meaning.

1.2 Why This Is Not Yet the Interesting Question

The rebuttal is correct, and this monograph does not dispute it. But correct is not the same as complete. Having established that the presence of a parser says nothing decisive about whether the surrounding system possesses genuine semantic competence, the natural next question is not whether such competence exists — a question that remains open and that no amount of architectural inspection will settle by itself — but a different, more tractable one: given that real systems visibly do decompose into learned and non-learned components, is

[2]The rebuttal sketched in this section follows, and in places directly draws its structure from, Mirtunjaya Goswami, Claude Code, Chomsky, and the Illusion of the Wizard Behind the Curtain (LinkedIn, July 6, 2026), `https://www.linkedin.com/pulse/claude-code-chomsky-illusion-wizard-behind-curtain-mirtunjaya-goswami-qhr5c/`. The extension of that rebuttal into the reconstruction-cost and irreversibility-materialization framework developed from Chapter 3 onward is this monograph's own contribution and is not present in Goswami's essay.

there a principled account of which computations end up externalized, or is the boundary simply wherever engineering convenience happened to draw it on a given afternoon?

The dominant answer available in the existing discussion is economic. A computation gets pulled out of the learned component and given a dedicated, exact implementation when doing so is cheap relative to the alternative: when the subproblem admits a polynomial-time algorithm with completeness guarantees, and when learning that algorithm implicitly, imperfectly, and expensively inside a neural network buys nothing that the exact version does not already provide for less. This is a real and useful principle. It explains, at the coarsest level, why a parser would be worth building. It does not explain very much else. It does not explain why some structures — logs, revision histories, provenance records, the retained trace of a system's own prior states — persist even when nothing about them is expensive to recompute in the ordinary sense, because in the relevant sense they cannot be recomputed at all: the information needed to reconstruct them was discarded the moment the system moved past the state that produced it.

This monograph is organized around the claim that these are two different phenomena wearing the same appearance — "a structure got kept instead of rebuilt" — and that collapsing them into a single economic story obscures the more interesting of the two. We will call the familiar one reconstruction-cost materialization and the neglected one irreversibility materialization, and the distinction between them, developed formally in Chapter 4 through Chapter 6, is the theoretical spine of everything that follows.

1.3 Where HYDRA Enters

The distinction is not merely descriptive. It can be given a formal treatment using machinery already developed, under a different motivation, for the HYDRA architecture: a multi-module system built around the notion that a state's admissibility is not an intrinsic property of the state but a relational one, defined with respect to a trajectory and a family of permitted continuations. HYDRA maintains explicit distinction projections — partial, inspectable models of an environment — and repairs them under failure rather than silently absorbing failure into an opaque representation. This gives the reconstruction/irreversibility distinction a home: a projection is worth retaining, on HYDRA's account, exactly when its loss would make some class of prior states unreachable from the system's current position, which is a claim about the topology of the system's history rather than a claim about the price of a CPU cycle.

Because HYDRA's projections are explicit objects rather than latent vectors that must be inferred after the fact through probing and causal tracing, the architecture offers something the transformer literature has been chasing for years without settling: a candidate

answer to what an intermediate structural representation — a Logical-Form analogue, in the vocabulary of Chapter 2 — would actually look like if a system built one on purpose rather than approximating its behavior implicitly.

1.4 Plan of the Argument

Chapter 2 restates the syntax/semantics rebuttal in full, because the rest of the monograph depends on having it available precisely rather than gesturally, and because the Logical Form literature it draws on — the syntactic explanation of scope ambiguity, the multiple derivations available for a single surface string — turns out to matter again later, once we ask what an explicit intermediate representation inside an artificial system would need to do to earn the comparison. Chapter 3 lays out the reconstruction-cost account of externalization in its strongest form and shows where it runs out of grip: cases where structure persists that the cost account cannot explain without simply redescribing the outcome as evidence of its own premise. Chapter 4 introduces the reconstruction/ irreversibility distinction proper. Chapter 5 gives it formal content in terms of trajectories, reachability, and admissibility. Chapter 6 extracts a criterion that can be checked against a real system's architecture rather than only applied retrospectively. Chapter 7 through Chapter 9 return to HYDRA and to the Claude Code leak, using the distinction to sort the leak's contents and to re-pose, in a sharpened form, the question of what — if anything — mediates syntax and semantics inside a language model. The concluding part, Chapter 11 through Chapter 14, situates the argument within a broader claim about continuation as a computational resource in its own right, and states plainly what would falsify the theory being proposed.

Chapter 2

The Category Mistake

This chapter restates, in a form precise enough to be used later, the argument that disposes of the naive inference from "there is a parser" to "there is no understanding." Readers already convinced of this point may proceed to Chapter 3; what follows is stated in full because the vocabulary it establishes — autonomy of syntax, Logical Form, the interface architecture of a grammar — is reused without re-derivation in Chapter 7 and Chapter 9.

2.1 Architectural Decomposition Is Not Ontological Decomposition

The mistake at the center of the original argument is a conflation of two different kinds of claim. An architectural claim describes how a system's computation is organized: which subproblems are solved by which mechanisms, and how those mechanisms are composed. An ontological claim describes what kind of thing a system's overall competence amounts to: whether it constitutes genuine understanding, mere simulation, or something not well captured by either label. Observing that a system's architecture contains a deterministic component is a fact about the first kind of claim. It bears on the second kind of claim only if one has already assumed, without argument, that genuine competence is identifiable by the absence of architectural decomposition — that is, only if one has already assumed the conclusion.

Definition 2.1 (Architectural decomposition). A system S exhibits architectural decomposition with respect to a task T if T can be partitioned into subtasks $T_1, \ldots, T_n$ such that each T_i is solved by a distinguishable mechanism M_i, where the M_i need not share an implementation substrate.

Remark 2.2. Definition 2.1 is deliberately weak: it says nothing about whether the M_i are learned or hand-written, symbolic or statistical, exact or approximate. Compilers, database engines, satisfiability solvers, and — as it turns out — coding agents built around large

language models all exhibit architectural decomposition in this sense. The definition is stated only to make explicit that decomposition is a structural property with no ontological content attached to it by default.

2.2 The Autonomy of Syntax

The strongest available precedent for treating architectural decomposition as expected rather than damning comes from generative linguistics. Chomsky's mature position on the relation between syntax and semantics is that syntactic computation possesses explanatory autonomy: the well-formedness of a structural derivation is not determined by, and does not reduce to, facts about meaning. The standard demonstration is a pair of sentences with opposite diagnostic status:

> Colorless green ideas sleep furiously.

This sentence is syntactically well-formed — it obeys the combinatorial rules of English grammar — while being semantically anomalous under ordinary interpretation. Its counterpart,

> Furiously sleep ideas green colorless.

is neither grammatical nor interpretable. The contrast shows that grammaticality and semantic coherence are independent axes: a string can fail on one while succeeding on the other. If syntax were merely a surface reflection of semantic well-formedness, this dissociation would not be possible.

A second example, also due to Chomsky, demonstrates the converse dependency: that semantic ambiguity is frequently explained by multiple available syntactic derivations rather than existing prior to and independent of syntactic structure.

> I almost had a book stolen.

This sentence supports at least three readings: that the speaker nearly became the victim of a theft; that the speaker nearly arranged for someone else to steal a book; and that the speaker nearly succeeded in stealing a book themselves. Many speakers do not spontaneously recognize the third reading until it is pointed out. The explanation is not that the sentence is vague at the level of meaning and merely happens to admit several paraphrases; it is that the string is compatible with more than one underlying syntactic derivation, each of which supports a distinct semantic composition. Meaning here is downstream evidence for hidden structure, not a substitute for computing that structure.

Proposition 2.3 (Autonomy of syntax, informal). Syntactic well-formedness and semantic interpretability are dissociable: neither is uniformly derivable from the other, and semantic ambiguity is in general explained by multiplicity at the level of syntactic derivation rather than existing as an independent primitive.

2.3 Logical Form as Interface

The dissociation in Proposition 2.3 motivates a specific architectural claim about the grammar itself. Rather than treating semantic interpretation as operating directly on surface strings, or even directly on surface syntactic structure, the relevant tradition locates interpretation at a further level of representation — Logical Form (LF) — derived from syntax by additional structure-building operations (quantifier raising, among others) and serving as the interface at which compositional semantics is defined. Quantifier scope ambiguity is the clearest illustration.

> Every programmer fixed a bug.

This sentence supports two readings: one on which a single bug was fixed by every programmer, and one on which each programmer fixed a possibly distinct bug. The ambiguity is not lexical — no word in the sentence is ambiguous — and it is not obviously present in the surface syntactic structure, which is unique. It arises because the surface structure is compatible with two distinct Logical Forms, differing in the relative scope of the universal and existential quantifiers, and semantic composition is sensitive to which LF is selected. The architecture this motivates is a pipeline:

$$\text{Syntax} \longrightarrow \text{Logical Form} \longrightarrow \text{Semantic Interpretation}$$

rather than the flatter picture in which surface strings map directly to meanings. This is the architecture, and not merely the vocabulary, that will matter later: an intermediate representation, distinct from both the raw string and the final interpretation, whose job is to make structure that is only implicit at the surface explicit enough for composition to be well-defined.

2.4 The Rebuttal, Stated Precisely

The pipeline above supplies the missing premise the original argument needed and did not have. If even human linguistic competence — the least controversial case of genuine semantic understanding available — routes through an intermediate structural stage that is itself

computed by autonomous, non-semantic machinery, then finding an analogous intermediate structural stage inside an artificial system is not evidence against that system's semantic competence. It is, if anything, evidence of architectural maturity: the system has factored a subproblem that admits exact, autonomous treatment into a dedicated stage, exactly as the grammar does.

Proposition 2.4 (Rebuttal). The presence of a deterministic parsing stage within a system's architecture is not, by itself, evidence for or against the presence of genuine semantic competence elsewhere in that architecture. It is evidence only that the system's overall computation is sufficiently decomposed to (a) admit Definition 2.1 with respect to the parsing subtask, and (b) benefit from doing so.

Proposition 2.4 is what disposes of the original argument. It is not, however, a positive theory of anything. It tells us what a parser's presence fails to show; it does not tell us what a system's decomposition pattern, taken as a whole, does show. That is the question the remaining chapters take up, and it requires leaving the syntax/semantics vocabulary behind, because — as Chapter 3 argues — not every instance of externalized structure in a real system is playing the role a parser plays.

Chapter 3

Why Structure Externalizes

Chapter 2 established that the presence of a non-learned component inside an otherwise learned system is not, by itself, informative about that system's semantic competence. This chapter asks a different question: given that real systems do decompose in this way, is there a principled account of which computations get externalized, or is the boundary simply drawn wherever engineering convenience happened to fall?

3.1 The Economic Account

The most immediately available answer is economic, and it is not wrong so much as incomplete. Some computations admit exact algorithms with strong complexity guarantees; parsing a context-free or mildly context-sensitive grammar is a canonical example, solvable in polynomial time by any of several well-understood algorithms. When a subproblem has this property, and when a system also has access to a statistical component capable of approximating the same subproblem less reliably and at greater expense, replacing the approximation with the exact algorithm is simply good engineering. Nothing about this requires any special theory beyond ordinary cost-benefit reasoning.

Definition 3.1 (Reconstruction-cost materialization). A structure X, derivable from available inputs by some process P, is said to be materialized under reconstruction-cost grounds if a system retains X (or a dedicated exact procedure for producing it) because the cost of running P each time X is needed exceeds the cost of storing or directly computing X.

Parsers fall under Definition 3.1. So do database indexes, memoized function results, compiled bytecode caches, and a great deal of the ordinary machinery of software engineering. The defining feature of this category is that the underlying information needed to reconstruct X is never actually lost; the system merely prefers not to pay the cost of reconstructing it. This has an immediate and testable consequence: materialization on

reconstruction-cost grounds is contingent on the cost structure that motivates it. Faster hardware, better algorithms, or a change in workload characteristics can in principle dissolve the advantage and make it rational to stop materializing X — to fold the parser back into a general-purpose learned approximation, for instance, if the approximation became cheap and reliable enough. Nothing about the structure itself resists this; only the price does.

3.2 Where the Account Runs Out

The economic account explains parsers, indexes, and caches comfortably. It does not explain, without strain, a second class of structures commonly found in the same systems: logs, revision histories, provenance chains, audit trails, version control, and — in the specific case under study here — the retained record of a coding agent's prior tool calls, permission decisions, and abandoned reasoning branches.

One can attempt to fold these into the economic account by arguing that reconstructing a log from scratch would also be expensive, and so the log is retained for the same reason a parser's output is retained. This move is available, but it purchases generality at the cost of emptiness. Under a sufficiently loose reading, any retained structure can be redescribed as having been "expensive to reconstruct," because reconstruction of a specific past state from an arbitrary later state is, in the relevant cases, not merely expensive but impossible — there is no procedure, however costly, that recovers the discarded information, because the information is genuinely gone. Describing an impossible reconstruction as an expensive one erases exactly the distinction that matters.

Example 3.2. Consider two structures maintained by a coding agent over the course of a session: (a) a cached parse tree for a source file that has not changed since it was last parsed, and (b) a log entry recording that the agent attempted an edit, the edit was rejected by a permission check, and the agent then chose a different approach. In case (a), discarding the cache costs the system a future re-parse; the parse tree can be exactly regenerated from the file on disk at any time, because the file on disk has not changed. In case (b), discarding the log entry costs the system something categorically different: there is no file on disk, no static input, from which "the agent tried X, was denied, and pivoted to Y" can be regenerated once the in-memory state has moved past that point. The information was never a deterministic function of anything that persists independently of the log.

Example 3.2 is the seed of the distinction developed formally in the next chapter. The two cases feel superficially similar — in both, a system is keeping something around rather than recomputing it on demand — but they differ in a way that the economic account, taken

alone, cannot register: in case (a), the information persists elsewhere (on disk) whether or not the cache is kept, so the cache is a convenience; in case (b), the log is the only place the information persists at all, so the log is not a convenience but a condition of the information's continued existence in the system.

3.3 A Broader Pattern

The pattern in Example 3.2 generalizes. Across many domains, one finds pairs of externalized structures that look economically similar but are doing different work:

- Parsers retain syntactic structure that could, in principle, be recomputed from source text at any time; search indexes retain retrieval structure recomputable from the underlying corpus; both are reconstruction-cost cases.
- Logs retain a system's own history, which cannot be recomputed from any static input because the history is not a function of a static input; version control retains a project's prior states, which are similarly not recoverable from the current state alone; both are candidates for a different kind of materialization.

The temptation, on noticing this pattern, is to reach immediately for a slogan: "whenever reconstruction becomes more expensive than retention, structure becomes explicit." This slogan is true of the first bullet and false, or at best vacuously true, of the second, and collapsing the two under one heading obscures exactly the distinction this monograph is built around. The remainder of Part II develops that distinction on its own terms.

Part II

The Sharper Distinction

Chapter 4

Two Kinds of Persistence

Example 3.2 distinguished, informally, a cached parse tree from a log entry. This chapter makes the distinction precise enough to carry theoretical weight, and argues that only one of the two kinds of persistence it identifies is properly explained by an economic argument. The first pass at this distinction, in terms of whether a prior state can be reconstructed from a later one, turns out to be slightly the wrong primitive. The correct primitive is whether a prior state remains distinguishable from other states that could equally well have produced the same successor. The chapter develops both notions and shows why the second subsumes and corrects the first.

4.1 Reconstruction-Cost Materialization, Restated

Definition 3.1 already gave the economic case its due: a structure is retained because recomputing it is more expensive than storing it. We restate it here with one addition that will matter throughout the rest of the monograph: the notion of a generating input.

Definition 4.1 (Generating input). Given a structure X produced by a process P, a generating input for X is a state s, independent of the system's own execution history, from which P can produce X (or something observationally equivalent to X) without reference to any intermediate state of the system that produced the original instance of X.

Remark 4.2. For a cached parse tree, the source file on disk is a generating input: P (the parser) can be re-run against it at any time, and the result does not depend on which prior parses happened to occur or in what order. For a search index, the underlying corpus is a generating input in the same sense.

Proposition 4.3 (Reconstruction-cost materialization requires a generating input). A structure X is a candidate for reconstruction-cost materialization (Definition 3.1) only if X has

a generating input in the sense of Definition 4.1. If no such input exists, the decision to retain X cannot be explained on cost grounds, because there is no alternative — however expensive — of recomputing X to be weighed against retention.

Proposition 4.3 is close to definitional, but its consequence is not: it hands us a test. Given a structure a system retains, ask whether a generating input exists for it. If yes, the retention decision is, at least potentially, an ordinary cost tradeoff, and Chapter 3's economic account applies without remainder. If no, the retention decision requires a different account, developed in the rest of this chapter.

4.2 Transition Dynamics and the Collapse of Predecessors

To state that account precisely we need a minimal model of how a system moves through states at all.

Definition 4.4 (Transition system). A transition system is a triple (X, U, F) where X is a set of states, U is a set of admissible inputs (edits, tool calls, inferences, corrections — whatever the system's own dynamics accept as driving a state change), and $F : X \times U \to X$ is a transition function, $x_{t+1} = F(x_t, u_t)$.

Definition 4.5 (Fiber). For a state $y \in X$, the fiber of y under F, restricted to a set $P \subseteq X \times U$ of admissible predecessor pairs, is

$$\mathsf{Fib}_P(y) = \{(x, u) \in P : F(x, u) = y\}.$$

F is non-injective over P at y if $|\mathsf{Fib}_P(y)| > 1$.

Non-injectivity is the ordinary case, not the exception. Editing a file, revising a belief, compacting a context window, applying a correction, merging two branches — each of these is, from the point of view of the resulting state alone, compatible with more than one admissible predecessor. Two different edit sequences can produce the same diff; two different chains of evidence can produce the same revised belief; two different sequences of corrections can produce the same repaired projection. The successor state, taken by itself, does not encode which member of its fiber actually occurred.

Proposition 4.6 (Fiber collapse). If F is non-injective over P at y, then no function of y alone determines which $(x, u) \in \mathsf{Fib}_P(y)$ produced y.

Proof. Immediate from the definition of a function: if g were such a function, $g(y)$ would have to equal every element of $\mathsf{Fib}_P(y)$ simultaneously, which is impossible when $|\mathsf{Fib}_P(y)| > 1$. □

Proposition 4.6 is close to a restatement of the pigeonhole principle, and it is stated here not because it is deep but because it is the premise everything else in the chapter depends on, and because making it explicit forces the right question into view: not "can y be used to reconstruct its predecessor," which Proposition 4.6 already answers in the negative whenever the fiber is nontrivial, but what has to be added to y for the predecessor's identity to survive at all.

4.3 History Channels and Distinguishability

The natural repair for fiber collapse is to augment the state with a history channel that tracks, alongside x_t, enough information to separate the fiber when it matters.

Definition 4.7 (History-augmented transition system). A history-augmented transition system extends (X, U, F) with a history space H and an update function $G : H \times X \times U \to H$, giving the joint dynamics

$$(x_{t+1}, h_{t+1}) = \big(F(x_t, u_t),\ G(h_t, x_t, u_t)\big).$$

The question of whether this augmentation actually helps — whether retaining h_t does anything beyond taking up space — is exactly the question of whether G separates the fiber that F collapses.

Definition 4.8 (Distinguishability preservation). Fix a fiber $\mathsf{Fib}_P(y)$ and a value h_t of the history channel prior to the transition. The augmented dynamics preserve distinguishability over $\mathsf{Fib}_P(y)$ at h_t if the map

$$(x, u) \longmapsto G(h_t, x, u), \qquad (x, u) \in \mathsf{Fib}_P(y),$$

is injective: distinct admissible predecessors in the fiber are sent to distinct values of h_{t+1}.

Theorem 4.9 (Historical Distinction Theorem). Let (X, U, F) be a transition system augmented with history space H and update function G as in Definition 4.7, and let $y \in X$ with fiber $\mathsf{Fib}_P(y)$ over admissible predecessors P. Then the identity of the predecessor $(x_t, u_t) \in \mathsf{Fib}_P(y)$ is recoverable from (y, h_{t+1}) alone if and only if the augmented dynamics preserve distinguishability over $\mathsf{Fib}_P(y)$ at h_t in the sense of Definition 4.8.

Proof. ($\Leftarrow$) If $(x, u) \mapsto G(h_t, x, u)$ is injective on $\mathsf{Fib}_P(y)$, then the observed value h_{t+1} determines a unique $(x, u) \in \mathsf{Fib}_P(y)$ with $G(h_t, x, u) = h_{t+1}$, since no other member of the fiber maps to the same h_{t+1}. This (x, u) is recoverable by inverting G restricted to the fiber. ($\Rightarrow$) If the map is not injective, there exist distinct $(x, u) \neq (x', u')$ in $\mathsf{Fib}_P(y)$

with $G(h_t, x, u) = G(h_t, x', u') = h_{t+1}$. Both are consistent with the observed pair (y, h_{t+1}), so no procedure operating on (y, h_{t+1}) alone can determine which occurred, by the same pigeonhole argument as Proposition 4.6. □

Theorem 4.9 is the theorem the informal discussion in Chapter 3 was circling without quite reaching. It replaces the vague claim that "logs help" with an exact condition: a history channel helps with respect to a given fiber exactly when it is injective on that fiber, and not otherwise. This has an immediate and useful corollary against a tempting shortcut.

Corollary 4.10 (Retention is not automatically adequate). A system may retain a nonempty history channel $h_{t+1} = G(h_t, x_t, u_t)$ and still fail to preserve distinguishability over a fiber of interest, if G is not injective on that fiber. The mere presence of a trace does not entail that the trace is adequate to the collapse it is meant to record.

A log that records only "a file was edited" does not preserve distinguishability over the fiber of possible edits; a log that records the diff does. Both are, in the loose sense, "a retained history channel." Corollary 4.10 is what makes the difference between them a theoretical distinction rather than a matter of degree.

4.4 Reconstructive Irreversibility

We can now replace the reconstruction-based notion used informally earlier with the corrected, distinguishability-based one, and give the phenomenon a name that does not invite confusion with thermodynamic irreversibility. Nothing in this chapter concerns entropy, phase-space volume, or coarse-graining; the collapse at issue is a property of a transition function's injectivity, not of a physical process's statistical mechanics. We will use reconstructive irreversibility throughout to keep this separation explicit.

Definition 4.11 (Reconstructive irreversibility). A transition $x_{t+1} = F(x_t, u_t)$ is reconstructively irreversible over a fiber $\mathsf{Fib}_P(x_{t+1})$ if F is non-injective over $\mathsf{Fib}_P(x_{t+1})$ (Definition 4.5) and no generating input in the sense of Definition 4.1 exists from which the predecessor's identity could otherwise be recovered.

Definition 4.12 (Irreversibility materialization). A history channel h is materialized on reconstructive-irreversibility grounds over a fiber $\mathsf{Fib}_P(y)$ if the system retains h because, absent that retention, the transition into y would be reconstructively irreversible (Definition 4.11) over that fiber, and h preserves distinguishability (Definition 4.8) over the fiber.

Example 3.2's log entry satisfies Definition 4.12: the fact that the agent attempted an edit, was denied, and pivoted, is not recoverable from any static input once the in-memory

state advances past that point, because the fact is a fact about a transition the system's own dynamics performed and collapsed, not a fact about an external artifact the transition merely read. Retaining the log entry is not a matter of avoiding an expensive recomputation; it is a matter of keeping the predecessor distinguishable at all.

4.5 The Distinction Is Not Gradable in the Same Dimension

It is tempting to treat reconstruction-cost and reconstructive- irreversibility materialization as two ends of a single spectrum — "cheap to rebuild" shading continuously into "expensive to rebuild" shading into "impossible to rebuild." This is the move Chapter 3 warned against, and Theorem 4.9 shows precisely why it fails. Cost is a scalar quantity that can in principle be driven arbitrarily close to zero by external improvements — faster hardware, better algorithms — without changing anything about what is being computed. Distinguishability preservation is not a scalar that improved hardware can shrink; it is a fact about whether a map is injective on a fiber, and injectivity is not the kind of property that a faster processor can grant to a function that lacks it.

Proposition 4.13 (Non-reducibility). There is no monotonic transformation of cost, however extreme, that converts a reconstruction-cost materialization into an irreversibility materialization (Definition 4.12) or vice versa. The two are distinguished by the existence of a generating input (Definition 4.1) and by the injectivity of the retained history channel on the relevant fiber (Definition 4.8), neither of which is a cost.

Sketch. Suppose X is materialized on reconstruction-cost grounds, so by Proposition 4.3 a generating input s exists for X. No change to the cost of running P on s removes s's status as a generating input. Conversely, suppose a history channel h is materialized on reconstructive-irreversibility grounds over some fiber, so by Definition 4.11 no generating input exists and by Definition 4.12 h's value lies in its being injective on that fiber (Theorem 4.9). Cost reduction does not manufacture a generating input where none existed, nor does it change whether G is injective on the fiber; injectivity is a structural property of G and the fiber, fixed independently of how fast G can be evaluated. □

4.6 Two Tests

The chapter closes with two tests, sharpened from the informal versions given earlier, to be applied to the Claude Code leak's contents in Chapter 8.

- The generating-input test. Ask whether a state or artifact exists, independent of the system's own execution history, from which the structure could be regenerated

without reference to the intervening trajectory. If yes, Definition 4.1 is satisfied and the structure is at most a reconstruction-cost case.

- The fiber-injectivity test. If no generating input exists, identify the fiber of admissible predecessors the transition in question collapses, and ask whether the retained structure is injective over that fiber in the sense of Definition 4.8. If yes, the structure is doing real reconstructive-irreversibility work per Theorem 4.9. If a structure is retained around a non-injective transition but is not itself injective on the relevant fiber, Corollary 4.10 applies: the system has paid a retention cost without buying back the distinction it presumably wanted.

These tests are stated at a level of generality that applies equally to a compiler, a database, a coding agent, or a cognitive architecture. The next chapter develops the admissibility apparatus proper, situating Theorem 4.9 within the trajectory-relative notion of admissibility that HYDRA uses, and asking what it takes for a system to allocate history channels in a way that tracks reconstructive-irreversible transitions specifically, rather than either everywhere or nowhere.

Chapter 5

Formalizing Irreversibility

Chapter 4 established when a history channel is doing real work: exactly when it is injective on the fiber of predecessors a transition collapses (Theorem 4.9). What it did not establish is which fibers a system ought to care about distinguishing in the first place. Not every collapsed distinction matters. A system that tried to preserve distinguishability over every non-injective transition it ever executed would drown in its own history channel long before the retention bought it anything useful. This chapter supplies the missing criterion, using the notion — central to the HYDRA architecture — that admissibility is a relational property of a state with respect to a trajectory and a family of permitted continuations, rather than an intrinsic property of the state alone.

5.1 Admissibility as a Relational Property

Definition 5.1 (Continuation family). Given a transition system (X, U, F) as in Definition 4.4, a continuation family assigns to each state $x \in X$ a set $C(x) \subseteq X^{\mathbb{N}}$ of trajectories beginning at x that are considered permitted — consistent with whatever constraints (physical, logical, normative, or task-specific) the system is operating under.

Definition 5.2 (Trajectory-relative admissibility). A state x_i occurring within a trajectory $\tau = (x_0, x_1, \dots)$ is admissible relative to τ and a continuation family C if the suffix of τ beginning at x_i is a member of $C(x_i)$: that is, if the way the trajectory actually continues after x_i is one of the continuations x_i permits.

Remark 5.3. Definition 5.2 makes admissibility relational in the precise sense HYDRA's architecture requires: it is not a predicate on x_i alone, since the same state can be admissible relative to one continuation and inadmissible relative to another. This is why admissibility cannot be checked by inspecting a state in isolation — it requires access to the trajectory

the state is embedded in, or at minimum to enough of the trajectory's future to evaluate membership in $C(x_i)$.

This relational character is precisely what connects admissibility to the machinery of Chapter 4. If admissibility depended only on the current state, a system could always re-derive it from x_i and would never need a history channel to evaluate it. Because admissibility depends on trajectory membership, and because Proposition 4.6 tells us trajectories are generically collapsed by F, a system's ability to evaluate its own admissibility — now or later — can itself be destroyed by the same collapse that destroys predecessor identity.

5.2 Admissibility Estimates and Repair

A terminological note is necessary before proceeding. In the HYDRA literature, "projection" has a specific, prior meaning: the Minimal Projection Theorem constructs, for a fixed admissibility structure A on a trajectory space $\mathcal{X}$, a unique canonical quotient map $\pi^* : \mathcal{X} \to \mathcal{M}^*$ collapsing exactly the states with identical admissible futures ($x_1 \sim x_2 \iff A(x_1) = A(x_2)$), through which every other admissibility-preserving map factors. That object is static, canonical, and not something a system revises — it is fixed once A is fixed, and "repairing" it would only make sense as revising one's beliefs about what A is, not as an operation on π^* itself.

The object this chapter needs is different in kind: not the canonical map π^*, but a system's own time-indexed, resource-bounded estimate of it, built and corrected under evidence the system receives incrementally rather than derived in closed form from a fully specified A. We write this object $\hat{\pi}_t$ throughout, to keep it visibly distinct from π^*, and return to the precise relationship between the two — $\hat{\pi}_t$ is best understood as a sequence converging toward, but not generally reaching, π^* — in Chapter 7.

HYDRA does not evaluate admissibility against the raw state x_t directly. It maintains this intermediate estimate.

Definition 5.4 (Admissibility estimate). An admissibility estimate $\hat{\pi}$ is a partial model of the environment maintained by a system: a structured, inspectable hypothesis about which states are admissible, distinct from x_t itself and revisable independently of it. The space of admissibility estimates is written $\hat{\Pi}$.

Definition 5.5 (Fit relation and anomaly). A fit relation $\models \subseteq \hat{\Pi} \times E$ relates admissibility estimates to evidence, where E is a space of observations the system can receive. A piece of evidence e is anomalous with respect to $\hat{\pi}$ if $\hat{\pi} \not\models e$: the estimate fails to accommodate the observation.

Definition 5.6 (Repair operator). A repair operator is a function $R : \hat{\Pi} \times E \to \hat{\Pi}$ such that, for any $\hat{\pi}$ and any e anomalous with respect to $\hat{\pi}$, $R(\hat{\pi}, e) \vDash e$: the repaired estimate accommodates the evidence that falsified its predecessor.

A repair event is a transition in exactly the sense of Definition 4.4, with $\hat{\Pi}$ playing the role of X, E playing the role of U, and R playing the role of F:

$$\hat{\pi}_{t+1} = R(\hat{\pi}_t, e_t).$$

Proposition 5.7 (Repair transitions generically collapse fibers). For a repair operator R satisfying Definition 5.6, it is generically the case — true for most fit relations $\vDash$ of practical interest, and false only when $\vDash$ is fine enough to make every estimate uniquely falsifiable in exactly one way — that R is non-injective: distinct pairs $(\hat{\pi}, e) \neq (\hat{\pi}', e')$ can satisfy $R(\hat{\pi}, e) = R(\hat{\pi}', e')$.

Justification. Repair operators are typically required to produce an estimate that accommodates the new evidence and is otherwise minimal or canonical in some further sense (simplest correction, least-disruptive revision, maximum-likelihood update). Whenever more than one prior estimate $\hat{\pi}, \hat{\pi}'$, subjected to more than one piece of anomalous evidence e, e', converges under this minimality requirement on the same corrected estimate, R is non-injective at that estimate by Definition 4.5. This is the ordinary case: many different specific errors, once corrected under a "fix the anomaly with minimal disruption" policy, converge on the same repaired state, because the correction is determined more by the minimality criterion than by the specific defect that triggered it. □

Proposition 5.7 is the bridge Chapter 4's apparatus needed. Repair is not a special kind of transition exempt from fiber collapse; it is, if anything, an especially prone case, because minimality criteria actively erase the specific character of whatever they correct. Theorem 4.9 therefore applies to repair directly: whether the identity of a failed estimate survives its own repair depends on whether the system's retained history channel is injective over the fiber of estimates that repair could have corrected to the same result.

5.3 Admissibility-Relevant Fibers

Corollary 4.10 already warned that retaining some history channel is not the same as retaining an adequate one. This section supplies the criterion for adequacy that Chapter 4 deliberately left open: adequate with respect to what. The answer, given Definition 5.2, is: with respect to whichever fibers affect the evaluation of admissibility for some future state.

Definition 5.8 (Admissibility-relevant fiber). A fiber $\mathsf{Fib}_P(y)$ is admissibility-relevant if there exists some later state x_j, reachable by the system's dynamics from y, and some continuation family C, such that the admissibility of x_j relative to C (Definition 5.2) differs depending on which member of $\mathsf{Fib}_P(y)$ actually occurred.

Example 5.9. Suppose a coding agent's context-compaction routine collapses two distinct reasoning traces — one that correctly ruled out an unsafe refactor, and one that merely ran out of budget before considering it — into the same summarized context. If a later step re-proposes the same refactor, whether that proposal is admissible (has it already been correctly ruled out, or does it merely appear unconsidered) depends on which predecessor occurred. The compaction fiber is admissibility-relevant. By contrast, if two different low-level tool invocations happen to produce byte-identical log lines, and nothing downstream ever depends on which specific invocation produced the line, the corresponding fiber is not admissibility-relevant, even though it is, in the sense of Definition 4.5, collapsed.

Theorem 5.10 (Selective Retention). A system that wishes to preserve exactly the admissibility judgments its continuation families can support, at minimum retention cost, must allocate history channels that are injective (Definition 4.8) precisely over admissibility-relevant fibers (Definition 5.8), and need not allocate injective history channels over fibers that are not admissibility-relevant.

Sketch. Necessity: if a fiber is admissibility-relevant, there is some future state x_j and continuation family C whose admissibility verdict depends on which predecessor in the fiber occurred. If the system's history channel is not injective over that fiber, then by Theorem 4.9 the predecessor's identity is not recoverable from (y, h_{t+1}), so the admissibility verdict for x_j cannot be determined either, since by hypothesis it depends on that identity. Injectivity over the fiber is therefore required to support the admissibility judgment. Sufficiency of restricting injective retention to admissibility-relevant fibers, and the minimum- cost claim, follow from the fact that (by Definition 5.8) no future admissibility judgment depends on distinguishing predecessors within a non-admissibility- relevant fiber; retaining injectivity there would preserve a distinction with no consumer, at nonzero retention cost, and could be dropped without any admissibility judgment becoming unsupportable. □

Theorem 5.10 is the theorem that gives the earlier, informal prediction — that trace allocation should track historical non-recoverability rather than mere computational expense — its sharpest form. It does not merely say retention should track irreversibility in general; it says retention should track the specific subset of reconstructively irreversible fibers whose collapse would otherwise make some future admissibility judgment undecidable, and it licenses discarding distinguishability everywhere else, including at other reconstructively irreversible fibers that happen not to feed into any later admissibility check.

5.4 What This Rules Out

Theorem 5.10 has a corollary worth stating on its own, because it is the form in which the theorem will be tested against the Claude Code leak in Chapter 8.

Corollary 5.11 (Two failure modes). A system's retention architecture can fail Theorem 5.10 in exactly two ways: it can under-retain, leaving some admissibility-relevant fiber without an injective history channel, so that some future admissibility judgment becomes undecidable; or it can over-retain, maintaining injective history channels over fibers that are not admissibility-relevant, paying a retention cost that no future admissibility judgment consumes.

Under-retention is the failure mode the original "wizard behind the curtain" debate never considered, because it is invisible from the outside until the specific future state that needed the distinction actually arrives and cannot be evaluated. Over-retention is the failure mode naive economic accounts (Chapter 3) are well equipped to notice and penalize, since it shows up directly as wasted storage or compute. The interesting architectural question — taken up properly once HYDRA's own projection machinery is on the table in Chapter 7 — is whether a system's actual pattern of retention tracks admissibility-relevance specifically, or merely approximates it by over-retaining broadly (safe but wasteful) or under-retaining broadly (cheap but occasionally undecidable), neither of which requires the system to have represented admissibility-relevance as such at all.

Chapter 6

A Falsifiability Criterion

Theorem 5.10 states what a system's retention architecture ought to track, given the definitions Chapter 5 supplies. Before using it to read HYDRA's own architecture (Chapter 7) or the Claude Code leak (Chapter 8), it is worth being honest about what kind of claim it is, because the honest answer determines what would actually count as evidence against it.

6.1 What Would It Mean to Falsify a Theorem Proved From Definitions?

Theorem 5.10 was proved, in Chapter 5, directly from Definition 5.8: a fiber counts as admissibility-relevant exactly when some future admissibility judgment depends on distinguishing its members, and the theorem then observes that injective retention is required exactly there and nowhere else. This is close to definitional in the same sense that the HYDRA architecture's own Recursive Admissibility Stabilization result is close to definitional given its notion of an A-preserving transition: the inductive step follows immediately once the definitions are fixed, and no substantive empirical claim about the world has yet been made. A theorem of this kind cannot be falsified by any observation, because no observation could conflict with a claim that is true by construction. Its value, as with HYDRA's result, is organizational: it tells you exactly what property to look for, not whether that property holds of anything real.

The empirical content of this monograph's argument, if it has any, has to live elsewhere. Two questions carry that content, and this chapter addresses both.

1. Can admissibility-relevance (Definition 5.8) be determined for a real fiber in a real system without already knowing what that system retains — or does the notion collapse into the same vacuity Chapter 3 diagnosed in the naive reconstruction-cost

account, where any retained structure can be redescribed after the fact as having been worth retaining?

2. If admissibility-relevance can be determined independently, does a real system's actual pattern of retention track it — and if it does not, what observable consequence follows?

6.2 An Independence Criterion

The vacuity risk is real and specific. Chapter 3 showed that "expensive to reconstruct" can be applied to anything a system happens to retain, after the fact, with no independent check. Definition 5.8 is at risk of the same move: nothing so far stops someone from declaring a fiber admissibility-relevant precisely because, and only because, the system under study happens to retain a distinguishing trace there. What is needed is a way to decide admissibility-relevance that does not consult the system's retention architecture at all.

Definition 6.1 (Externally testable admissibility-relevance). A fiber $\mathsf{Fib}_P(y)$ is externally admissibility-relevant at a later state x_j, relative to a continuation family C, if replaying the system's dynamics forward from each distinct predecessor $(x,u),(x',u') \in \mathsf{Fib}_P(y)$ in turn — holding every subsequent input fixed — yields different admissibility verdicts for x_j under C (Definition 5.2), regardless of whether the system's own architecture actually retains anything that would let it compute this difference.

Definition 6.1 is an interventionist test: it asks what would have happened under two counterfactual histories, holding everything downstream fixed, and checks whether the two replays diverge in their eventual admissibility verdict. It can be carried out — in principle, and in practice for systems simple enough to simulate or log exhaustively — by an external observer with access to the system's specification and dynamics, without any reference to what the system itself chose to keep. This is what makes it usable as an independence check.

Proposition 6.2 (Non-vacuity). Admissibility-relevance, tested via Definition 6.1, is decidable (in the sense of being determinable by simulation or replay against the system's specified dynamics) without reference to the system's retention architecture. Consequently, the prediction of Theorem 5.10 — that retention should track admissibility-relevance — is not vacuous in the way Chapter 3 showed the naive reconstruction- cost account to be: relevance and retention are established by two independent procedures, and it is possible, prior to inspecting a system's retention architecture, to have already fixed which of its fibers are relevant.

This is the qualification that matters: Proposition 6.2 does not claim any particular system's retention tracks relevance. It claims only that the question is well-posed — that there is a fact of the matter, checkable independently, against which a system's actual retention pattern can be compared and found to match or not to match.

6.3 The Audit Procedure

Put together, Definition 6.1 and Theorem 5.10 license a concrete procedure, applied to HYDRA's own architecture in Chapter 7 and to the Claude Code leak in Chapter 8.

1. Enumerate candidate fibers. Inspect the system's transition dynamics for non-injective transitions (Definition 4.5) — edits, corrections, compactions, merges, overwrites, and any other operation that is many-to-one on admissible predecessors.

2. Test relevance externally. For each candidate fiber, apply Definition 6.1: replay from distinct predecessors with subsequent inputs held fixed, and check whether any downstream admissibility verdict diverges.

3. Audit retention. Independently, inspect what the system actually retains at that fiber, and check whether the retained structure is injective on the fiber (Definition 4.8).

4. Compare. Cross-tabulate the two independent findings against Corollary 5.11: relevant and retained is compliant; relevant and not retained is under-retention; not relevant and retained is over-retention; not relevant and not retained is compliant by omission.

6.4 Two Failure Signatures

The comparison in step 4 produces a prediction with observable consequences, and it is worth stating what those consequences should look like, because "under-retention" and "over- retention" are not directly visible from the audit alone — they are claims about what a system's behavior should do, which still needs to be checked against what it does do.

Definition 6.3 (Undecidability failure). A system exhibits an undecidability failure at x_j if a downstream admissibility judgment for x_j is, by Definition 6.1, genuinely relevance-dependent on some fiber, the system's retention does not preserve distinguishability over that fiber (Definition 4.8), and the system nonetheless produces a verdict for x_j — necessarily either by guessing, by falling back to a default, or by silently picking one member of the fiber as if it were the only one.

Definition 6.3 is stated carefully because the naive expectation — that under-retention shows up as a visible crash or error — is usually wrong. Systems under-provisioned in this sense do not typically halt; they resolve the ambiguity silently, often by defaulting to whichever fiber member is most recent, most common, or otherwise privileged by the implementation, and proceed as if no ambiguity existed. This makes the failure mode empirically subtler than a crash, but not empirically inaccessible.

Definition 6.4 (Default-resolution test). Where an undecidability failure is suspected, construct a controlled pair of runs that differ only in which member of the relevant fiber occurred, with everything else held fixed, and check whether the system's eventual verdict for x_j tracks the true predecessor across repeated trials at a rate exceeding chance. If the verdict is statistically independent of which predecessor actually occurred, the system is resolving the ambiguity by default rather than by genuine distinction, confirming an undecidability failure by Definition 6.3 even though no explicit error was raised.

Over-retention is comparatively easy to confirm: it shows up directly as retained structure, chargeable against storage or compute, that no downstream admissibility judgment — checked exhaustively via Definition 6.1 against every state reachable from the fiber in question — ever consults.

6.5 Two Falsification Conditions

With Definition 6.3 and Definition 6.4 in hand, the empirical claim behind Theorem 5.10 — that a well-engineered system's retention pattern should approximate the relevance/retention match Corollary 5.11 describes, and that departures from the match should correlate with observable consequences — can be stated in a form that a specific system can fail.

- Falsification by silent success. If a system is found, by Definition 6.1, to under-retain at a genuinely admissibility-relevant fiber, and the default-resolution test (Definition 6.4) shows its verdicts at the affected downstream states nonetheless track the true predecessor at better than chance — meaning some mechanism other than the audited retention channel is carrying the distinction — the audit procedure has misattributed the system's competence, and the specific claim that this retention channel is what supports this admissibility judgment is false. This does not refute Theorem 5.10 as a definitional claim, but it refutes the specific reading of a system as an instance of the theorem's prescription, and would require locating the channel that is actually doing the work.

- Falsification by uncorrelated retention. If a system's overall retention pattern, audited across many fibers, shows no tendency for admissibility-relevant fibers (per

Definition 6.1) to receive injective treatment more often than admissibility-irrelevant fibers do — if relevance and retention are statistically independent across the audited fibers — then the broader claim that engineering pressure shapes retention architecture toward admissibility-relevance, rather than toward something else entirely (raw reconstruction cost, implementation convenience, historical accident), is falsified for that system. This is the condition Chapter 8 will actually test.

6.6 Scope Limits

Two limitations of the audit procedure are worth stating plainly rather than discovering by way of an overreached claim later.

First, exhaustive enumeration of fibers (step 1 of the audit procedure) is generally infeasible for any system complex enough to be interesting; real audits will sample candidate fibers rather than enumerate them, and a sampled audit can only support probabilistic conclusions about the overall correlation claim, not certainty about any specific unsampled fiber.

Second, Definition 6.1's counterfactual replay assumes the system's dynamics are specified precisely enough to replay at all. For a system with genuinely stochastic or environment-coupled transitions, "holding subsequent inputs fixed" requires care about what counts as the same subsequent input across two runs that started from different predecessors, and the test degrades gracefully into a statistical rather than a deterministic comparison. This is not a defect specific to this framework; it is the ordinary difficulty of counterfactual evaluation in any non-deterministic system, and the default-resolution test (Definition 6.4) is already built to accommodate it by using repeated trials rather than a single comparison.

Neither limitation undermines the two falsification conditions above; both mean that a real audit produces a graded, evidence-weighted verdict rather than a single decisive proof. Chapter 7 applies the procedure to HYDRA's own projection machinery, where the architecture is specified precisely enough for a fairly direct audit. Chapter 8 applies it to the Claude Code leak, where the audit is necessarily sampled and partial, and the chapter says so explicitly rather than overstating what a leaked codebase, inspected after the fact, can actually establish.

Part III

HYDRA and the Case Study

Chapter 7

HYDRA's Projections as Explicit Intermediate Representation

Chapter 5 left one question open by design: whether iterating the repair operator R actually drives the estimate sequence $\hat{\pi}_0, \hat{\pi}_1, \hat{\pi}_2, \ldots$ toward the canonical quotient π^* that HYDRA's Minimal Projection Theorem guarantees exists, or merely toward some sequence that keeps accommodating whatever evidence arrives without ever becoming correct. This chapter answers the question, and the answer is more interesting than the affirmative most treatments would reach for: consistency, stability, and convergence are three different properties, the repair operator as defined so far only guarantees the weakest of them, and the gap between stability and convergence has an exact, constructible counterexample rather than remaining a vague worry.

7.1 Repair as Hypothesis Pruning

The cleanest way to state the distinction is to stop treating $\hat{\pi}_t$ as a free-floating "estimate" and instead say explicitly what it is an estimate of.

Definition 7.1 (Hypothesis space). Let $\mathcal{A}$ be the space of admissibility structures a system considers possible on its trajectory space $\mathcal{X}$ — the candidates among which the true structure $A^* \in \mathcal{A}$ lies. Each $A \in \mathcal{A}$ induces, by the Minimal Projection Theorem, a canonical quotient $\pi_A : \mathcal{X} \to \mathcal{M}_A$; write $\pi^* := \pi_{A^*}$ for the canonical quotient induced by the true structure.

Definition 7.2 (Estimate as hypothesis set). A system's admissibility estimate at time t is the set $H_t \subseteq \mathcal{A}$ of admissibility structures still compatible with everything accommodated so far. The induced estimate quotient is $\hat{\pi}_t := \pi_{H_t}$, defined by $x_1 \sim_{H_t} x_2$ iff $\pi_A(x_1) = \pi_A(x_2)$

for every $A \in H_t$: two states are treated as equivalent by the estimate only when every remaining candidate structure agrees they are equivalent.

Definition 7.2 makes the estimate conservative by construction: it only asserts an identification when no live hypothesis would contest it. This is the right default for an object that is supposed to support admissibility judgments (Definition 5.2) rather than merely summarize evidence, since a false identification — treating two states as equivalent when the true structure distinguishes them — is a different and worse kind of error than a missing identification.

Definition 7.3 (Repair as pruning). A repair operator R acting on hypothesis sets satisfies:

1. Fit. Every $A \in R(H_t, e_t)$ is compatible with e_t (the evidence that triggered repair).

2. Contraction. $R(H_t, e_t) \subseteq H_t$: repair only removes candidates, never reintroduces a structure previously excluded. This is the single axiom subsuming what earlier discussion treated as two separate conditions — distinguishability monotonicity and non-reintroduction of eliminated equivalence classes are the same requirement stated on the estimate side and the hypothesis side of the same object.

3. Soundness. If the evidence stream $e_0, e_1, \ldots$ is genuinely generated by A^*, then $A^* \in H_t$ for all t: repair never prunes the true structure, because true evidence never contradicts it.

Fit is the condition Definition 5.6 already stated. Contraction and Soundness are the additions this chapter makes explicit, and together they replace the earlier, weaker requirement — that a repaired estimate merely accommodate the latest anomaly — with the stronger requirement that it remain compatible with the entire evidence history. This is exactly the gap identified against Definition 5.6 as originally stated: nothing there prevented a repair from silently reintroducing an inconsistency with e_{t-1} while fixing its inconsistency with e_t. Definition 7.3 closes that gap by axiom rather than leaving it to be assumed.

7.2 Consistency, Stability, and Convergence

With H_t as the primitive object, the three notions can be stated without ambiguity.

Definition 7.4 (Consistency). The estimate sequence is consistent if $H_t \neq \emptyset$ for all t. By Soundness (Definition 7.3), consistency is automatic whenever the evidence is genuinely generated by some fixed $A^* \in \mathcal{A}$: A^* itself always witnesses non-emptiness.

Proposition 7.5 (Stability under a descending chain condition). If $\mathcal{A}$ satisfies the descending chain condition (every strictly decreasing chain of subsets of $\mathcal{A}$ is finite), then the sequence $H_0 \supseteq H_1 \supseteq H_2 \supseteq \cdots$ guaranteed by Contraction (Definition 7.3) stabilizes: there exists T such that $H_t = H_T$ for all $t \geq T$. Write $H_\infty := H_T$ for this eventual value.

Proof. A strictly decreasing chain $H_0 \supsetneq H_1 \supsetneq H_2 \supsetneq \cdots$ would violate the descending chain condition unless it terminates; since $H_t \supseteq H_{t+1}$ always by Contraction, the sequence either strictly decreases (finitely often, by hypothesis) or stabilizes, and once $H_t = H_{t+1}$, Contraction forces $H_{t'} = H_t$ for all $t' \geq t$. □

Proposition 7.5 is doing exactly the work anticipated earlier: stability is not a further axiom on R, it is a free consequence of Contraction plus a well-foundedness assumption on the hypothesis space. Systems with infinite, non-well-founded hypothesis spaces need not stabilize at all; this is a genuine restriction, not a technicality, and is returned to in Section 7.5.

Definition 7.6 (Convergence). The estimate sequence converges (to π^*) if H_∞ induces the same quotient as A^*: $\pi_{H_\infty} = \pi^*$.

Proposition 7.7 (The limit estimate refines the truth). π_{H_∞} is always at least as fine as π^*: whenever $\pi_{H_\infty}(x_1) = \pi_{H_\infty}(x_2)$, also $\pi^*(x_1) = \pi^*(x_2)$. Equivalently, π_{H_∞} never falsely identifies two states that π^* distinguishes.

Proof. By Definition 7.2, $\pi_{H_\infty}(x_1) = \pi_{H_\infty}(x_2)$ requires $\pi_A(x_1) = \pi_A(x_2)$ for every $A \in H_\infty$. By Soundness, $A^* \in H_\infty$, so in particular $\pi_{A^*}(x_1) = \pi_{A^*}(x_2)$, i.e., $\pi^*(x_1) = \pi^*(x_2)$. □

Proposition 7.7 is the precise sense in which the estimate is conservative rather than merely cautious: it can fail to converge only by retaining too many distinctions, never by collapsing ones the truth requires. This gives an exact characterization of when convergence holds and, more usefully, an exact description of what a failure to converge looks like.

Theorem 7.8 (Convergence characterization). The estimate sequence converges to π^* (Definition 7.6) if and only if every $A \in H_\infty$ induces the same quotient as A^* — equivalently, if and only if the evidence stream eventually separates A^*, in the sense of inducing a different quotient, from every rival hypothesis in $\mathcal{A}$ that disagrees with A^* on the classification of at least one pair of states.

Proof. By Proposition 7.7, π_{H_∞} never identifies a pair π^* distinguishes, so $\pi_{H_\infty} = \pi^*$ can fail only by π_{H_∞} withholding an identification π^* makes. By Definition 7.2, π_{H_∞} withholds the identification of x_1, x_2 exactly when some $A' \in H_\infty$ dissents, i.e., $\pi_{A'}(x_1) \neq \pi_{A'}(x_2)$ while $\pi^*(x_1) = \pi^*(x_2)$ — which is precisely a pair on which A' disagrees with A^*. So

$\pi_{H_\infty} = \pi^*$ holds iff no surviving $A' \in H_\infty$ disagrees with A^* on any pair, i.e., iff every surviving hypothesis induces the identical quotient to A^*. □

Example 7.9 (Stable, consistent, and non-convergent). Let $\mathcal{X} = \{x_1, x_2, x_3\}$ and let $\mathcal{A} = \{A^*, A'\}$ where A^* places x_1 and x_2 in the same admissibility class, A' places them in distinct classes, and both structures agree on every other classification, including everything about x_3. Suppose the evidence stream $e_0, e_1, \ldots$ consists entirely of observations bearing on x_3 and on the shared classifications, never on whether x_1 and x_2 are admissibility-equivalent. Then $A^* \in H_t$ for all t by Soundness, and A' is never excluded either, since no evidence has ever borne on the one point where A^* and A' disagree; so $H_t = \{A^*, A'\}$ for all t, the sequence is trivially stable from $t = 0$, and it is consistent throughout. But $\pi^*(x_1) = \pi^*(x_2)$, while $\pi_{H_t}(x_1) \neq \pi_{H_t}(x_2)$ for every t, since A''s dissent blocks the identification by Definition 7.2. The sequence is stable and consistent for all t, and never converges: it permanently withholds an identification the truth supports, because the one piece of evidence that would resolve A^* against A' never arrives.

Example 7.9 is the constructible version of "a projection can be stable without being correct": stability is a property of the evidence stream drying up on the disagreements that remain, not a property of the disagreements having been resolved. Convergence requires the stronger condition that the evidence stream be, in the relevant sense, exhaustive with respect to every rival hypothesis — a condition Theorem 7.8 states exactly and Example 7.9 shows is not automatic.

7.3 Repair Logs as Admissibility-Relevant History Channels

The hypothesis-pruning formulation also answers a question Chapter 5 left implicit: what, concretely, should a system retain around a repair event, beyond the repaired estimate itself? The natural candidate is now visible directly from Definition 7.3: the record of which hypotheses were pruned, and by which evidence.

Definition 7.10 (Repair log). A repair log at time t is a record of the pruning step $H_{t-1} \setminus H_t$ together with the evidence e_{t-1} responsible for it.

By Theorem 5.10, this log is admissibility-relevant — worth retaining as an injective history channel in the sense of Definition 4.8 — exactly when some later comparison between surviving hypotheses in $H_{t'}$, $t' > t$, would come out differently depending on which specific structures were pruned at step t rather than merely how many. This is not automatic: two different pruning steps that happen to leave H_t in the same state are, by Proposition 4.6, indistinguishable from H_t alone, and Theorem 5.10 says the log is worth keeping only

insofar as some downstream admissibility judgment actually depends on knowing which. A repair log kept unconditionally, for every pruning step regardless of downstream relevance, is exactly the over-retention failure mode of Corollary 5.11.

7.4 What Makes This an Explicit Intermediate Representation

Chapter 2 left open what it would take for an artificial system to possess something functionally comparable to Logical Form: an intermediate representation, distinct from both raw input and final interpretation, that structure-building operations construct and later stages consume. The apparatus of this chapter is offered as a candidate answer, and its distinguishing property is not that it exists — interpretability researchers have long since found structure of various kinds inside transformer latent spaces — but that it is inspectable in the specific sense the audit procedure of Chapter 6 requires.

Given a system built on Definition 7.2, the audit procedure's steps are not merely available in principle; they are answerable directly from the architecture's own bookkeeping. Step 1 (enumerate candidate fibers) corresponds to enumerating repair events; step 3 (audit retention) corresponds to checking which repair logs (Definition 7.10) were kept; and Definition 6.1's counterfactual replay, which for an opaque system requires simulating the entire architecture twice, here reduces to comparing H_t across the two hypothetical evidence orderings directly, since H_t is already an explicit set rather than a distributed pattern that must be reconstructed by probing. A latent vector must be interpreted before it can be compared; a hypothesis set can simply be compared.

This is a narrower claim than "HYDRA understands admissibility and transformers do not," and it should be read as exactly that narrow. What Section 7.4 establishes is that if a system is built on Definition 7.2, the question of whether its retention tracks admissibility-relevance is decidable by inspection rather than by inference. Whether any given implementation of HYDRA is in fact built this way, and whether anything comparable can be recovered from a transformer's internals well enough to run the same audit, are separate empirical questions, taken up for the leaked architecture specifically in Chapter 8 and for the general transformer semantics question in Chapter 9.

7.5 Scope: What This Chapter Assumes About HYDRA

Two assumptions this chapter's results depend on are worth stating plainly, because HYDRA's own presentation is honest about exactly where its formal machinery is established versus conjectural, and this chapter's results should be filed the same way.

First, Proposition 7.5 requires $\mathcal{A}$ to satisfy a descending chain condition. HYDRA's

own account of admissibility structures does not establish this in general; it is a substantive assumption about the hypothesis space a given implementation searches over, true for finite or well-ordered hypothesis spaces and not guaranteed otherwise. Where it fails, an estimate sequence may continue revising indefinitely without ever stabilizing, and neither Definition 7.6 nor Theorem 7.8 directly applies.

Second, this chapter's entire apparatus presupposes that a system's estimate is well-modeled as a subset H_t of some fixed hypothesis space $\mathcal{A}$, known in advance. Real systems may instead construct or extend $\mathcal{A}$ itself as evidence arrives — a form of ontological revision, rather than mere pruning of a fixed space, that Definition 7.3 does not cover. Whether HYDRA's actual repair machinery operates over a fixed or an expanding hypothesis space is left open here and is relevant to how much of Theorem 7.8 survives contact with a real implementation — a question Chapter 8 does not resolve either, since a leaked codebase's implementation choices do not, by themselves, settle a question about the mathematical structure of its hypothesis space.

Chapter 8

Reading the Leak Through the Distinction

This chapter applies the audit procedure of Chapter 6 to the Claude Code leak described in Chapter 1. It does not revisit whether the presence of a parser inside the leaked codebase says anything about semantic competence; Chapter 2 settled that. The question here is narrower and more useful: does the leaked architecture's pattern of externalized structure actually sort into reconstruction-cost and irreversibility materialization the way Chapter 4 and Chapter 5 predict, and where it retains history channels around collapsed fibers, does that retention track admissibility-relevance (Definition 5.8) rather than something else?

8.1 What Kind of Audit This Is

Chapter 6 was explicit that the audit procedure degrades gracefully but not silently when full replay access is unavailable. This chapter's version of the audit is a specification-level audit: it works from architectural details made public through post-leak technical analysis — blog write-ups, security research, and independent reconstructions of the leaked TypeScript source — rather than from direct execution of the codebase against controlled counterfactual histories. This means step 2 of the audit procedure, the counterfactual replay of Definition 6.1, is applied here in a weakened form: rather than literally running the system twice from divergent predecessors, admissibility-relevance is inferred from documented behavior — what the architecture's own hooks, checkpoints, and permission logic are described as doing — which is a real but strictly weaker form of evidence than a controlled replay. Every classification below should be read with that qualification attached, and Section 8.8 returns to it directly.

Structure	Generating input?	Admissibility-relevant (documented)?	Retention observed?	Classification
Parse/search infrastructure (file indexing, grep-style tools)	Yes — source tree on disk	No	Cached, not logged	Reconstruction-cost
Prompt cache (cache-break vectors)	Yes — prior context tokens	No	Cached with explicit invalidation conditions	Reconstruction-cost
Companion PRNG state (deterministic per-user generator)	Yes — user identifier plus fixed salt	No	Recomputed, not stored	Reconstruction-cost (trivial)
Bash command risk classification (numbered security checks)	N/A — stateless function of command content	No fiber collapse to track	None needed	Not a retention case
Compaction chain (non-destructive boundary markers on the transcript)	Yes, while source transcript retained — see Section 8.4	Yes — later steps must be able to tell what was summarized versus discarded	Retained; boundaries patched at read time; transcript retention duration undocumented	Time-dependent: reconstruction-cost while transcript survives, reclassifies on pruning
Pre-risky-operation checkpoint (version-control commit before a flagged mutation)	Yes, but only because explicitly created for this purpose	Yes — rollback admissibility depends on the pre-operation state	Retained, deliberately, keyed to risk classification	Irreversibility
Cross-session trust state	No	Contested — see Section 8.5	Deliberately not retained	See Section 8.5
Background daemon decision log (unconfirmed feature)	No	Plausible, unconfirmed	Reportedly append-only	Irreversibility (low confidence)

Table 8.1: Specification-level audit of documented Claude Code components against Definition 3.1 and Definition 4.12.

8.2 The Audit Table

8.3 A Confirming Case

One row of Table 8.1 is worth dwelling on as a clean confirmation, and it is worth being honest that it is the only one that stays clean under scrutiny. Section 8.4 returns to the compaction chain, which looked equally clean on first pass and turns out not to be once its generating input is examined over time rather than at a single instant.

The pre-risky-operation checkpoint confirms the selective part of Theorem 5.10. A version-control commit is not retained before every operation; it is documented as triggered specifically before operations flagged as risky. This is close to a direct implementation of the fiber-injectivity test from Chapter 4: the architecture is not paying retention cost uniformly across all transitions, which Corollary 5.11 would flag as over-retention if it were, nor is it declining to retain around destructive edits generally, which would be under-retention; it is retaining precisely at the transitions where a later admissibility judgment — is rollback to the pre-operation state still available and consistent — would otherwise become undecidable in the sense of Definition 6.3. Unlike the compaction chain, the checkpoint's status does not depend on anything decaying later: a git commit, once made, does not become a worse generating input over time in the way Section 8.4 shows a compacted transcript can.

8.4 An Ambiguous Case: Context Compaction and the Decay of Generating Inputs

The compaction chain was classified, above and in Table 8.1, as a clean instance of irreversibility materialization: boundaries recorded explicitly, original material preserved append-only rather than destructively edited. That classification is correct for a system in the state the documentation describes — but it is correct only conditionally, and making the condition explicit exposes a gap in Definition 4.1 itself that the earlier chapters did not need to confront and this case forces into the open.

Definition 4.1 asks whether a state s exists, independent of the system's own execution history, from which a structure X can be reproduced. Nothing in that definition says when s must exist. Every use of the definition so far has been able to get away with this omission because the cases considered — source files on disk, corpora underlying a search index — are generating inputs that, for practical purposes, do not expire. Transcripts are different. They are exactly the kind of state that real systems prune: documented session-lifecycle hooks include cleanup tasks that remove stale logs and temporary files, and no system retains raw conversational history indefinitely as storage and cost considerations

accumulate. This forces a question Definition 4.1 was never asked before: a generating input for what, measured at which time?

Definition 8.1 (Time-relative generating input). A state s is a generating input for X at time t (Definition 4.1) if s remains accessible to the system, or to an auditor applying Chapter 6's procedure, at t. Write $\mathsf{GenInput}(X, t) = 1$ if such an s is accessible at t, and 0 otherwise.

Proposition 8.2 (Classification can decay). A structure X can satisfy $\mathsf{GenInput}(X, t_1) = 1$ and $\mathsf{GenInput}(X, t_2) = 0$ for $t_2 > t_1$, without any change to X itself, to the process that produced it, or to anything executed at t_2: it suffices that the generating input s, accessible at t_1, becomes inaccessible by t_2 through an unrelated retention decision — a cleanup routine, an archival policy, a storage limit — that has nothing to do with X or with the transition that produced it.

Proof. Immediate from Definition 8.1: the predicate $\mathsf{GenInput}(X, t)$ depends only on whether s is accessible at t, which can change due to any process that removes s, regardless of whether that process has any relationship to X. □

Applied to the compaction chain, Proposition 8.2 says something the earlier confirming treatment elided. So long as the pre-compaction transcript is retained, the compacted summary is, in the relevant sense, reconstruction-cost material relative to that transcript: a higher-fidelity re-derivation remains possible, and the transcript itself is the object actually doing the irreversibility- grounds work, exactly as the earlier classification assumed. But if a session's raw transcript is later pruned — by exactly the kind of cleanup task documented elsewhere in the same architecture — every summary derived from it silently changes classification at that moment, from a reconstruction-cost derivative of a retained transcript to the only remaining trace of a now-permanently-collapsed fiber. And because compaction is lossy by design — it exists specifically to discard detail under context pressure — the summary that remains after that moment is not injective over the fiber it would now need to preserve (Definition 4.8): it was never built to be, because at the time it was built, it did not need to be.

This is a genuinely uncomfortable finding for the classification scheme this monograph has been applying, not because it shows the theory wrong, but because it shows the theory's own machinery identifying a failure mode invisible to a single-timepoint audit. The available reporting does not establish whether the compaction system tracks, or is designed to care about, whether a summary's source transcript has since been pruned; there is no documented mechanism analogous to the pre-risky-operation checkpoint's explicit triggering condition that would suggest the architecture treats this transition — from reconstruction-cost to reconstructive irreversibility — as one requiring any response. If no such mechanism

exists, the architecture is, at the moment of pruning, silently converting a population of compliant structures into under-retained ones without any audit trail marking that it happened, which is precisely the kind of failure Definition 6.3 describes and precisely the kind step 3 of the audit procedure (Chapter 6) is least equipped to catch, since it audits retention as it currently stands rather than tracking how retention adequacy changes as generating inputs elsewhere expire.

The same pattern recurs twice more. Two other rows in Table 8.1 exhibit the identical structure once looked at this way, and neither is worked through in full here, since the mechanism is now established. A trust score derived from a history of approvals and denials is reconstruction-cost material relative to that history while the history survives; if the history is later discarded and only the aggregate score retained, the same silent reclassification occurs, and the score — a single scalar — is certainly not injective over the fiber of specific approval histories that could have produced it. Repair logs (Definition 7.10) themselves are not exempt: Chapter 7 treated a repair log as the history channel that preserves distinguishability over a pruning event, but a repair log is itself a structure subject to its own later transitions — archival, compression, summarization under storage pressure — and Proposition 5.7's argument that repair transitions generically collapse fibers applies recursively to the log's own lifecycle exactly as it applies to the estimate the log was built to explain. A compressed repair log is a history channel that has itself become subject to reconstructive irreversibility, and nothing this monograph has established guarantees that the compression preserves distinguishability over whichever fibers the original log was retained to cover.

A second, sharper tension, independent of pruning. The decay argument above concerns what happens to a summary's classification over time, as its generating input expires. A different and harder problem is already present at the moment the summary is first created, before any transcript has been pruned at all. A single compaction event does not act on one fiber; it acts across many simultaneously — every distinct fact, decision, and ruled-out alternative in the pre-compaction transcript is a candidate fiber in its own right, and the summarizer's output is injective over some of these and non-injective over the rest, by construction, in the same breath. This is not a defect to be engineered away: a summary that was injective over every fiber in the original transcript would simply be the transcript, and compaction would have accomplished nothing.

Definition 8.3 (Selective compression). Let $F_1, \ldots, F_n$ be the candidate fibers present in a structure X prior to a compressive transition G. Write $S_G \subseteq \{1, \ldots, n\}$ for the subset of fibers over which G is injective (Definition 4.8). G is admissibility-aligned if S_G contains

every admissibility-relevant fiber (Definition 5.8) among $F_1, \ldots, F_n$; admissibility-misaligned if some admissibility-relevant fiber falls outside S_G.

Definition 8.3 is the sharper version of the question this case actually poses. It is not "does the compacted summary preserve distinguishability," which has the trivial answer partially, obviously, that is what compression means. It is whether the specific partiality lines up with what future admissibility judgments will need — whether S_G happens to cover the admissibility-relevant fibers specifically, or covers some other selection entirely, chosen by whatever criterion the summarizer actually optimizes for. Theorem 5.10, applied at this resolution, is a design target for the summarizer, not merely for the surrounding architecture: a compaction mechanism that is admissibility-aligned in the sense of Definition 8.3 is doing exactly what Theorem 5.10 would prescribe at the level of individual facts, and one that is misaligned is producing exactly the under-retention failure of Corollary 5.11, fact by fact, regardless of how sophisticated or well-reviewed its overall summarization quality appears to be by any other standard.

Nothing in the available reporting settles whether Claude Code's compaction is admissibility-aligned in this sense, and it is worth being explicit about why this is a harder question than anything else audited in this chapter. The parser, the checkpoint, and the trust score are all governed by code whose retention behavior is, at least in principle, inspectable directly: a reviewer can read the triggering condition. A compaction step performed by the model itself has no comparable artifact to inspect — "what did the summarizer decide was worth keeping, and why" is not a policy written down anywhere; it is a judgment made by the same kind of computation Chapter 9 already flagged as functionally adequate in ways that are not always architecturally transparent (Definitions 9.1 and 9.2). This means the compaction case is not merely unresolved by this chapter's specification-level audit; it may be a case where resolving it requires exactly the protocol Chapter 9 proposed and did not carry out — constructing disambiguating contexts, compacting them, and checking whether the admissibility-relevant fact survives the compaction — applied to the summarizer itself rather than to the base model's ordinary forward pass. The audit procedure does not fail silently here; it identifies, correctly, that the question it was built to answer has, in this one case, been handed to a component this monograph has already argued cannot be audited the same way everything else in this chapter was.

None of this identifies a case satisfying the strongest disconfirming pattern Section 8.7 describes — a structure with no generating input, a non-injective retained channel, and behavior that nonetheless tracks the true predecessor, indicating some other mechanism is quietly carrying the distinction. Finding that case would require exactly the replay-level access Section 8.8 already flagged this chapter as lacking: the ability to test, against the running system rather than its documentation, whether a post-pruning admissibility

judgment that should be undecidable by this chapter's own analysis is in fact answered correctly, and if so, by what. What this section establishes is narrower and, on its own terms, still a real complication: that the classification scheme has a time axis this monograph did not originally give it, that it also has a within-instant resolution axis this monograph did not originally give it either, that at least one leaked component sits exactly on both boundaries at once, and that the available reporting cannot say whether the architecture on the other side of either boundary behaves the way Theorem 5.10 would recommend or simply degrades without anyone having designed for the degradation.

8.5 The Session-Trust Case

One row resists easy classification, and it is worth working through rather than forcing into the table, because it exposes a real gap in how Definition 5.8 was stated. Documented behavior indicates that permission and trust decisions are not restored across session boundaries: trust is re-established each session rather than inherited from the last one. Read naively, this looks like under-retention of an admissibility-relevant fiber — whether a directory or repository has been trusted plainly affects the admissibility of later tool executions — and Theorem 5.10 would seem to predict a failure.

The naive reading is wrong, and the reason it is wrong matters more than the specific case. Definition 5.8 defines relevance relative to a continuation family C (Definition 5.1), and C is not fixed by the bare fact that trust affects later behavior; it is fixed by what the system is actually supposed to treat as a permitted continuation. If the governing policy is that trust must be re-established every session — that a continuation in which stale, unre-verified trust is carried forward is not itself admissible, regardless of what the trust state used to be — then the fiber in question is not admissibility-relevant at all under the correctly specified C: no permitted continuation ever depends on distinguishing which prior session's trust decision occurred, because no permitted continuation is allowed to use a prior session's trust decision in the first place. Discarding the state is then not a retention failure but the direct implementation of the admissibility structure itself.

This is a useful corrective to Chapter 6's own apparatus: Definition 6.1's counterfactual test is only as good as the continuation family it is run against, and getting C wrong — treating "trust used to be granted" as automatically part of a permitted continuation, when the actual policy denies this — would misclassify a deliberate safety boundary as an engineering oversight. The audit procedure does not resolve this automatically; it requires knowing, independently, what the system's designers intended C to permit. Where that is documented, as it is here, the audit can proceed correctly. Where it is not, this case is a warning about how easily the same observation can be read two ways.

8.6 A Case That Cuts the Other Direction

Chapter 6 predicted, via Definition 6.3 and Definition 6.4, that under-retention at a genuinely admissibility-relevant fiber should show up not as a crash but as a silent default resolution, exploitable if the default happens to be predictable. A documented pre-patch vulnerability in the trust pipeline matches this prediction closely enough to be worth stating as a confirming instance rather than only a hypothetical one.

The relevant fiber is whether a request was authorized by a trust decision the user genuinely made, or occurred before the user's trust prompt was ever shown — triggered instead by configuration supplied through a malicious repository. Under the documented vulnerability, the system could be induced to act on network configuration before the trust-verification step had executed, meaning the two members of this fiber — genuinely-authorized request and request smuggled in ahead of authorization — were, at the point of execution, indistinguishable to the system. This is Definition 6.3 exactly: a downstream action (making the request) proceeds by defaulting, in effect, to treating the fiber as though it had only one member. The default-resolution test (Definition 6.4) is precisely what independent security researchers ran, informally, when they demonstrated the exploit: constructing the two histories, holding the eventual action fixed, and showing the system's behavior did not track which one had actually occurred. Unlike the checkpoint case, and unlike the compaction chain so long as its transcript survives, this is not evidence that the architecture's retention tracks admissibility-relevance; it is evidence of exactly the failure mode the theory says under-retention at a relevant fiber should produce, at a point in the pipeline where, prior to the patch documented alongside the leak's disclosure, it evidently did not.

8.7 What Would Have Falsified the Prediction Here

Chapter 6 named two falsification conditions. Neither is decisively triggered by the material available for this audit, and it is worth saying plainly what would have triggered them, so the absence of falsification here reads as a real (if weak) test rather than an inability to fail.

Falsification by uncorrelated retention would have required finding that documented retention effort — logging, checkpointing, non-destructive history — shows no tendency to concentrate at documented non-injective transitions relative to injective ones: for instance, if the architecture retained exhaustive logs of read-only, side-effect-free operations (a case with no fiber to collapse in the first place) at the same density as it retains state around destructive edits and compactions. The available reporting shows the opposite pattern — retention concentrated at compaction boundaries, pre-mutation checkpoints, and permission-relevant events, sparse elsewhere — which is consistent with, though it does not

prove, the correlation Theorem 5.10 predicts.

Falsification by silent success would have required finding a documented case where a genuinely admissibility-relevant fiber was under-retained and yet downstream judgments tracked the true predecessor anyway, indicating some other mechanism was quietly doing the work the audited channel was credited with. No such case appears in the available reporting, though Section 8.8's caveats mean this is better read as "not found" than as "ruled out."

8.8 Limits of This Audit

The caveats from Section 8.1 are worth restating in their specific form here rather than left general. This chapter's classifications rest on public technical reporting produced by external parties reverse-engineering a leaked, minified-then- reconstructed codebase, not on direct inspection of Anthropic's source or design documentation. Several rows of Table 8.1 are marked at reduced confidence for exactly this reason, and the background daemon row in particular should be read as illustrative rather than established, given that its existence is itself disputed in the reporting this chapter draws on.

More importantly, the audit as conducted here cannot distinguish a system whose retention architecture was designed around admissibility-relevance from one whose retention architecture merely happens, for unrelated reasons, to correlate with it — for instance, if compaction boundaries are retained mainly because they are a natural byproduct of how the summarization step is implemented, and their admissibility-relevance is a fortunate side effect rather than a design target. Settling that question would require access this chapter does not have: internal design rationale, or a genuine replay-based audit of the kind Definition 6.1 describes in its strong form. What this chapter establishes is narrower and still worth having: that the distinction between reconstruction-cost and irreversibility materialization sorts a real, independently documented architecture into two populated and behaviorally distinct categories, that the sorting lines up with where the architecture is reported to concentrate retention effort, and that at least one documented failure — Section 8.6 — matches the specific failure signature the theory predicts for under-retention at a relevant fiber, rather than some other signature the theory did not anticipate.

Chapter 9

What the Leak Does Not Settle About Transformer Semantics

Chapter 2 left open the question that motivated this entire monograph: whether a transformer's internal computation contains anything functionally comparable to Logical Form, or whether its apparently LF-like behavior — tracking scope, binding, coreference, and the rest — is produced by some other means entirely. Having built the reconstruction-cost and irreversibility apparatus, applied it to HYDRA's own projection machinery, and audited a real leaked architecture against it, we are in a position to say something sharper about this question than Chapter 2 could: not an answer, but a precise specification of what would count as one, together with an important distinction the original debate ran together without noticing.

9.1 Two Bars, Not One

The Logical Form debate, as inherited from the original public argument and its rebuttals, implicitly asks a single question: does the transformer have an LF-like object. Chapter 7's apparatus makes visible that this is actually two questions, and that conflating them is most of what has kept the debate unresolved.

Definition 9.1 (Functional adequacy). A system is functionally adequate with respect to an admissibility-relevant fiber (Definition 5.8) if its downstream behavior passes the audit procedure of Chapter 6: its outputs track the true predecessor at better than chance under the default-resolution test (Definition 6.4), whatever internal mechanism produces that tracking.

Definition 9.2 (Architectural transparency). A system is architecturally transparent with respect to a fiber if, beyond being functionally adequate, it maintains an explicit, inspectable

object — in the sense of Definition 7.2, a hypothesis set that can be read off directly rather than inferred by probing — whose state can be compared against the fiber without requiring intervention on the system's internals.

HYDRA's admissibility estimate, as formalized in Chapter 7, is a design that targets both bars at once: the hypothesis-set construction is meant to be functionally adequate (it tracks π^* under the conditions of Theorem 7.8) and, by construction, architecturally transparent (the audit in Chapter 6 can be run against H_t directly). Nothing in the original Chomsky-derived debate distinguished these two properties, and the transformer case is exactly where they can come apart. A transformer could in principle be functionally adequate over a given fiber — its behavior genuinely tracks the distinction, passing every version of the audit procedure — while remaining architecturally opaque, with no component anyone has identified that plays the role H_t plays for HYDRA. Conversely, a system could be architecturally transparent by design and still fail functional adequacy if its explicit machinery is wrong. The two are independent, and settling one does not settle the other.

This reframes Chapter 2's open question. "Does the transformer have an LF-analog" was really asking about architectural transparency while gesturing at evidence — correct handling of scope, coreference, and the rest — that only bears on functional adequacy. A transformer could pass every behavioral test that motivated the original question while the question, read literally, remains open, because behavioral success is evidence for Definition 9.1 and silent on Definition 9.2.

9.2 A Transformer Audit Protocol

The audit procedure of Chapter 6 does not require architectural transparency to be run; it requires only the ability to construct controlled counterfactual histories and observe downstream behavior, which is available for a transformer through ordinary prompt intervention, without needing to interpret its internals at all.

1. Identify a candidate fiber. Choose a syntactic construction whose surface form is compatible with more than one underlying derivation — exactly the cases Chapter 2 used to motivate Logical Form in the first place.

2. Construct divergent predecessors. Build two contexts that supply different disambiguating evidence for which derivation is intended, while keeping the ambiguous string itself, and everything asked about it downstream, identical.

3. Hold the probe fixed. Ask the same downstream question — a paraphrase, an entailment judgment, a translation requiring the distinction to be resolved — against both contexts.

4. Apply the default-resolution test. Across many such pairs, check whether the answers track the supplied disambiguating evidence at better than chance (Definition 6.4). If they do, the transformer is functionally adequate over that fiber. If they do not — if the answer is the same regardless of which context was supplied, or tracks some surface feature uncorrelated with the intended derivation — it is not.

5. Only then, and only if adequacy holds, ask about transparency. Attempt to localize a component of the computation — a direction in the residual stream, a subset of attention heads, an intervention point discoverable by activation patching — whose value is injective over the fiber in the sense of Definition 4.8, independent of the final output. Success here is evidence for architectural transparency; failure to find such a component is not evidence against it, only evidence that it has not been found, which is a weaker claim and should be reported as one.

This protocol is offered as a methodological proposal, not as a result this monograph has carried out. Its content, in the classification this monograph has tried to be honest about throughout, is a framework definition organizing future empirical work, not an established finding.

9.3 Mapping the Candidate Hypotheses onto Audit Outcomes

Prior discussion of this question distinguished several ways a transformer's LF-like behavior might arise. Restated against Definition 9.1 and Definition 9.2, each corresponds to a distinct, in-principle-distinguishable audit outcome.

- Genuine internal construction. The transformer builds something functionally isomorphic to an LF representation and consults it. Predicted outcome: functional adequacy holds robustly, including under adversarial variants of step 2 designed to break surface correlations; and step 5 succeeds in localizing an injective component, since a genuinely constructed intermediate representation should leave some causal trace an intervention can find.

- Distributed approximation without an explicit object. The transformer's behavior tracks the distinction, but no localized component carries it; the relevant information is smeared across the representation in a way no single intervention point isolates. Predicted outcome: functional adequacy holds robustly, as in the first case, but step 5 fails to localize a component even under sustained search — functional adequacy without architectural transparency, exactly the combination Section 9.1 said the original debate had no vocabulary for.

- Statistical shortcut mimicking the distinction. The transformer's behavior tracks the distinction only insofar as the distinction correlates, in ordinary usage, with some surface feature the model has learned to exploit — word order frequency, lexical co-occurrence, or similar. Predicted outcome: functional adequacy holds under step 4's default form, but fails specifically under the adversarial variant of step 2 that preserves the intended derivation while breaking the surface correlation the shortcut depends on. This is the outcome that would most directly vindicate the original skeptical argument Chapter 2 was responding to, not because a parser was found, but because behavior would fail exactly where genuine derivation-sensitivity would be required and shortcut- sensitivity would not suffice.

The three outcomes are empirically distinguishable from one another using the same protocol, which is the protocol's main virtue: it does not require deciding among them in advance, only running steps 1 through 5 and reading off which pattern of successes and failures resulted.

9.4 Worked Candidates

Chapter 2's own examples are ready-made fibers for step 1. I almost had a book stolen supplies three derivations distinguishable by which participant is agent, patient, or would-be-agent of the theft; contexts can be constructed that disambiguate toward each in turn, with a downstream probe ("who was going to do the stealing") that requires the distinction to be resolved rather than merely gestured at. Every programmer fixed a bug supplies a scope ambiguity; contexts can be constructed establishing either a single shared bug or one bug per programmer, with a downstream probe ("how many distinct bugs were fixed") that has a definite, derivation-dependent answer. Both are exactly the kind of case Definition 6.1 needs: an unambiguous downstream verdict whose correctness depends on which underlying structure was intended, constructible without needing access to anything beyond ordinary prompting.

9.5 What the Leak Does Not Bear On

It is worth being explicit that Chapter 8's audit and this chapter's protocol concern entirely different objects. The leak exposed the scaffolding around the model — parsers, permission systems, compaction machinery, checkpointing — and Chapter 8 audited that scaffolding's retention pattern against Theorem 5.10. Nothing in that audit touches the transformer's own internal computation, which was never part of what leaked; the model weights and forward-pass computation are not source code, and no npm sourcemap exposes them. A

reader who came away from Chapter 8 thinking the leak had somehow settled this chapter's question has conflated the two objects this section is explicitly separating. The leak is evidence about architectural decomposition in the surrounding system, exactly as Chapter 2 argued from the outset; it was never evidence, one way or the other, about what happens inside the model itself.

9.6 What This Chapter Establishes

Not an answer. What it establishes is that the question motivating the whole debate — filtered through Definition 9.1 and Definition 9.2 — decomposes into two independent, separately testable questions where the original discussion had only one blurred question and no protocol for testing it. It further establishes, via Section 9.2's mapping, that the candidate hypotheses debated informally in the original exchange are not merely different intuitions about the same unresolvable mystery; they make different predictions about a concrete and executable procedure, and the procedure does not require settling questions about consciousness, understanding, or genuine semantics that the informal debate kept circling back to. It requires only running an intervention and reading off whether behavior tracks derivation or tracks shortcut, and then, separately, whether anyone can point to where.

Part IV

Synthesis

Chapter 10

Hamiltonian, Lagrangian, and Continuation Descriptions

The apparatus built across Part II and Part III — transition systems, fiber collapse, reconstructive irreversibility, admissibility as a relational property — was developed without reference to physics. This chapter places it against the two classical frameworks for describing how systems move through state spaces at all, not as imported vocabulary but as a way of saying precisely what this monograph's framework is and is not doing. The point of the comparison is not that continuation geometry resembles Hamiltonian or Lagrangian mechanics; it is that the three frameworks answer three different questions, and the difference is exact enough to state as propositions rather than analogies.

10.1 Hamiltonian Systems and State Sufficiency

A Hamiltonian system evolves a state $(q, p) \in M$ — position and momentum coordinates on a phase space M — under Hamilton's equations, generating for each t a flow map $\Phi_t : M \to M$.

Proposition 10.1 (Flow maps are diffeomorphisms). For a Hamiltonian vector field smooth enough to guarantee existence and uniqueness of solutions, the flow map Φ_t is, for each t in its domain of definition, a diffeomorphism of M: smooth, bijective, with smooth inverse Φ_{-t} (standard ODE existence and uniqueness theory; see Arnold, 1989).

This is the entire content needed to place Hamiltonian systems within the framework of Chapter 4.

Corollary 10.2 (Hamiltonian dynamics never collapse fibers). Treating $(M, \varnothing, \Phi_t)$ as a transition system in the sense of Definition 4.4 (with a trivial input space, since Hamiltonian

evolution is autonomous), every fiber $\mathsf{Fib}_P(y)$ under Φ_t is a singleton for every $y \in M$ and every admissible predecessor set P. Reconstructive irreversibility (Definition 4.11) therefore never occurs in a Hamiltonian system: $\Phi_{-t}(y)$ is always a generating input (Definition 4.1) for the predecessor state.

Proof. Immediate from bijectivity (Proposition 10.1): a bijective map has fibers of size exactly one, so Proposition 4.6's hypothesis is never satisfied, and Φ_{-t}, applied to y, recovers the unique predecessor exactly, witnessing Definition 4.1 directly. □

This is the precise sense in which a Hamiltonian system's present state is sufficient for its past: not merely that reconstruction is possible in some loose sense, but that Definition 4.12 is vacuous for such a system by construction, and no history channel is ever required on admissibility grounds (Theorem 5.10), because there is never a fiber for one to be injective over. Liouville's theorem — that Hamiltonian flow preserves phase-space volume (Arnold, 1989) — is the measure-theoretic strengthening of this fact: not only is no information lost pointwise, the flow does not even compress or expand the space of possibilities locally.

Remark 10.3 (Cost and information are still independent). Corollary 10.2 shows a Hamiltonian system is never reconstructively irreversible; it says nothing about how expensive reconstruction is. A chaotic Hamiltonian system — one with positive Lyapunov exponents — can require reconstruction precision growing exponentially in t to recover $\Phi_{-t}(y)$ to any fixed tolerance, making reconstruction-cost materialization (Definition 3.1) of an approximate trajectory entirely rational even though Definition 4.12 never applies. This is Proposition 4.13 given a physically real instance: reconstruction can be made arbitrarily expensive without the system ever crossing into the regime Definition 4.11 describes, because expense and the existence of a generating input are independent properties, exactly as Proposition 4.13 claimed in the abstract.

10.2 Lagrangian Systems and Trajectory Selection

The Lagrangian formulation starts from a different primitive: not an instantaneous state, but an entire trajectory γ, evaluated through an action functional $S[\gamma] = \int L(\gamma(t), \dot{\gamma}(t))\, dt$, with physical motion characterized by the extremal condition $\delta S = 0$, yielding the Euler–Lagrange equations (Arnold, 1989). Where the Hamiltonian question was what can be recovered from a state, the Lagrangian question is which trajectory, among all those satisfying given boundary conditions, is realized.

Remark 10.4 (Selection is orthogonal to reversibility). For standard classical Lagrangians, the Legendre transform relates the Lagrangian formulation to a Hamiltonian one, and the resulting dynamics inherit Proposition 10.1: the trajectory selected by extremizing S is

itself a reversible flow in the sense of Corollary 10.2. Extremal selection is therefore not, by itself, a claim about reconstructive irreversibility; it is an additional structure — a global ranking over candidate trajectories — layered on top of dynamics that may or may not collapse fibers. The Lagrangian picture's distinctive contribution is the idea of selection by optimization: a trajectory is picked out because it extremizes a functional defined over the whole path, not because of anything local to how each state transitions to the next.

Remark 10.4 isolates exactly the feature this monograph's framework needs to contrast itself against: not reversibility, which Part II already addressed directly, but optimization as an explanatory principle. Admissibility (Definition 5.2) has so far been defined without reference to any functional to be extremized, and the next section makes that omission a deliberate claim rather than an oversight.

10.3 Continuation Systems and Admissible History

Definition 5.2 defines a state as admissible relative to a trajectory and a continuation family — a binary, relational condition, not a ranking. This section states, and defends against an obvious objection, the claim that this is a substantive difference from the Lagrangian picture, not a restatement of it in different notation.

Proposition 10.5 (Admissibility does not require a global ranking). The admissibility relation C of Definition 5.1 can be specified, and Theorem 5.10 and Theorem 4.9 proved from it, without positing any functional S over trajectories such that $C(x)$ is characterized as the extremizers of S. It suffices that $C(x)$ partition continuations into permitted and forbidden, with no requirement that permitted continuations be comparable to one another by any common measure.

Justification. Every definition and theorem from Definition 5.2 through Theorem 5.10 refers to $C(x)$ only through set membership — whether a given continuation lies in $C(x)$ — and never through a comparison between two members of $C(x)$. No step in any proof in Chapter 5 required ranking, ordering, or measuring elements of $C(x)$ against each other. □

Remark 10.6 (Answering the trivial-optimization objection). One might object that any binary partition can be trivially recast as optimization of an indicator functional, $S[\gamma] = 0$ if $\gamma \in C(x)$ and $S[\gamma] = -\infty$ otherwise, making Proposition 10.5 empty. The objection is correct as far as it goes and does not rescue the Lagrangian analogy: the substantive content of the Lagrangian picture is not merely that some functional exists whose extremizers are the realized trajectories — under the indicator construction this is true of any deterministic or admissibility-constrained system whatsoever — but that the functional is well-behaved enough to support the machinery classical mechanics builds on it: smoothness,

the Euler–Lagrange equations, comparison of nearby trajectories by their action. The indicator functional supports none of this; it is a relabeling of the partition, not a variational structure. Proposition 10.5's claim is that continuation geometry needs no such structure, trivial or otherwise, to state or prove its central results, and the theorems of Chapter 5 are the evidence for this, having been proved without it.

With optimization set aside, what remains to characterize continuation systems is exactly the machinery already built: transitions that generically do collapse fibers (Proposition 5.7, in contrast to Corollary 10.2), and persistence secured not by selecting an optimal trajectory but by repair — revising an estimate under anomaly (Definition 5.6) so as to remain within the admissible region, without any claim that the resulting trajectory extremizes anything over its history.

10.4 Three Regimes

	Hamiltonian	Lagrangian	Continuation
Primitive	instantaneous state (q, p)	whole trajectory γ	admissibility relative to a continuation family
Central question	what can be recovered from a state	which trajectory is realized	what survives reconstruction
Fiber collapse	never (Corollary 10.2)	inherited from underlying Hamiltonian dynamics; generically none	generic (Proposition 5.7)
Selection principle	none needed — flow is deterministic and invertible	global extremization, $\delta S = 0$	none required (Proposition 10.5); persistence via repair under anomaly

Table 10.1: Three descriptions of trajectories through a state space, distinguished by what they take as primitive and what they require to be well-defined.

The three questions in Table 10.1's second row are the chapter's central claim, restated once more because it is worth having in a single place: Hamiltonian mechanics asks what can be predicted from a state; Lagrangian mechanics asks which trajectory is optimal; continuation geometry asks what survives reconstruction. The first two questions are well-posed independently of anything in this monograph and have been for centuries. The third is the question Chapter 4 through Chapter 9 have been answering, and Table 10.1 is intended to make clear that it is not a special case, restriction, or generalization of the other two, but a question with its own primitive — admissibility under a continuation family — that is well-defined without borrowing invertibility from the Hamiltonian picture or extremality from the Lagrangian one.

10.5 A Note on Variational Accounts of Persistence

HYDRA's own account of its place in the wider literature notes an approximate correspondence between the Free Energy Principle and minimizing a combination of the RSVP accessibility potential and entropy field.[1] That correspondence, whatever its precise form, places the Free Energy Principle on the Lagrangian side of Table 10.1: minimizing a variational quantity is selection by optimization, in exactly the sense Remark 10.4 isolated. Continuation geometry's departure from the Lagrangian picture in this chapter is therefore also, to whatever extent the correspondence holds, a departure from variational-optimization accounts of persistence generally, including that one. This is noted here as a position within the wider landscape of theories this framework sits alongside, not as a critique; nothing in this chapter establishes that admissibility-based persistence is correct where a variational account would be wrong, only that the two are answering different questions, and that this monograph's technical results (Chapter 5 onward) did not need the variational one to be stated or proved.

[1] The correspondence is stated at the level of general resemblance, not formal equivalence, and is not re-derived here; see the discussion of related work in HYDRA's own presentation.

Chapter 11

Continuation as a Computational Resource

The chapters of Part II and Part III were organized around a distinction — reconstruction-cost versus reconstructive-irreversibility materialization — applied first abstractly, then to HYDRA's own architecture, then to a real leaked system, then to the question of transformer semantics. This chapter asks what that distinction adds up to when stated at its most general: not a classification tool for sorting structures inside a particular architecture, but a claim about a resource that computation depends on and that is not reducible to the resources computer science already has names for. The claim is stated as a theorem, not asserted as a slogan, because Proposition 4.13 already contains everything the proof needs; what this chapter adds is the framing that makes the theorem worth having stated at all.

11.1 Three Resources and a Fourth Candidate

Time, space, and energy are the resources computational complexity theory is built around, and they share a property this chapter will argue continuation does not: they are, within limits, fungible with one another. Time-space tradeoffs are a standard subject of complexity theory; energy-time tradeoffs govern physical implementations of computation. Whatever the exact exchange rate, more of one resource can typically compensate for less of another across a wide range of problems.

The question this section poses is whether a system's capacity to answer future admissibility judgments (Definition 5.2) belongs to this same family — purchasable, in sufficient quantity, by throwing more time, more space, or more energy at the problem — or whether it is a different kind of thing entirely.

11.2 Distinction Budgets

Definition 11.1 (Distinction budget). Given a system's trajectory up to time t, its distinction budget B_t is the set of collapsed fibers $\mathsf{Fib}_P(y)$, for y occurring at or before t, over which predecessor identity remains recoverable — either because a generating input still exists (Definition 4.1) or because the system's retained history channel is injective over that fiber (Definition 4.8).

Proposition 11.2 (Monotonic shrinkage). Absent a positive retention action at the moment a fiber collapses, B_t can only lose members as t increases, never regain them: once a fiber's generating input has ceased to exist and no injective history channel was recorded for it at the time of collapse, no later state of the system restores that fiber to B_t.

Proof. A fiber $\mathsf{Fib}_P(y)$ leaves B_t exactly when neither a generating input nor an injective retained channel is available for it (Definition 11.1). Both conditions concern facts fixed at or before the moment of collapse: the existence of a generating input (Definition 4.1) is a fact about what remains independently available in the world, and the injectivity of a retained channel (Definition 4.8) is a fact about the specific G applied at the transition itself (Definition 4.7). Neither fact can be altered by anything that happens afterward: a generating input that has ceased to exist cannot be un-destroyed, and a channel that failed to be injective at the moment of retention cannot become injective retroactively, since its value h_{t+1} was already fixed by $G(h_t, x_t, u_t)$ at that moment and nothing later changes what that value was. □

Remark 11.3. Proposition 11.2 has the flavor of the data processing inequality from information theory, which states that no processing of a signal can increase the mutual information it carries about its source. The resemblance is structural rather than formal — this chapter's setting concerns discrete distinguishability over admissible predecessors rather than mutual information over a channel, and no claim is made that Proposition 11.2 is a special case of the classical result — but the shared shape is worth noting: both say that a one-directional process can only destroy distinguishing information already present, never manufacture it after the fact.

11.3 The Resource Independence Theorem

Theorem 11.4 (Resource independence). Let $\mathsf{Fib}_P(y)$ be a fiber subject to reconstructive irreversibility (Definition 4.11). Then for any candidate reconstruction procedure P' and any allocation of time, space, or energy to P', no such allocation enables P' to recover the identity of the predecessor in $\mathsf{Fib}_P(y)$.

Proof. By Definition 4.11, reconstructive irreversibility over $\mathsf{Fib}_P(y)$ holds exactly when no generating input (Definition 4.1) exists for the predecessor's identity. A reconstruction procedure, however much time, space, or energy it is allocated, is still a function from some input to an output; enlarging its resource bound changes which functions are computable within that bound, but does not change whether an input exists to compute from. Since, by hypothesis, no state or artifact independent of the system's own execution history determines the predecessor's identity, there is no input, of any kind, from which any procedure — regardless of the time, space, or energy available to it — could recover that identity. The claim does not depend on complexity-theoretic bounds at all; it depends only on the absence of a domain element to compute from, which no resource allocation supplies. □

Theorem 11.4 is, in the classification this monograph has tried to maintain throughout, close to definitional given Definition 4.1 and Definition 4.11 — its proof does not require anything beyond noticing that a resource-bounded computation is still a computation from an input, and that no allocation of resources creates an input where none exists. Its value is exactly the value Chapter 6 assigned to Theorem 5.10: not surprise, but precision. It converts Proposition 4.13's abstract claim, and Remark 10.3's physical illustration of it, into a direct statement about the standard resource triad: continuation is not purchasable with time, space, or energy, in the specific and limited sense that no quantity of any of the three restores a distinction whose generating input is gone.

11.4 Why This Is Not Merely “More Memory”

The most natural objection to treating continuation as a resource in its own right is that it collapses into space — into memory, the already-recognized resource for retaining information across a computation. The objection deserves a direct answer, because the answer is what makes continuation a different kind of thing rather than a fourth item on the same list.

Proposition 11.5 (Continuation is not a quantity of memory). Possessing unbounded memory does not, by itself, guarantee that a system's retained structure is injective over any given admissibility-relevant fiber. Injectivity (Definition 4.8) is a property of what is stored — of the map G applied at the moment of collapse — not of how much storage is available to store it in.

Justification. A system with arbitrarily large memory can still apply a retention policy G that discards the specific information distinguishing two predecessors — storing, for instance, only that an edit occurred and not what the edit was — regardless of how much

unused storage capacity remains. Definition 4.8 depends only on whether G separates the fiber, which is a structural fact about G independent of the size of H. □

This is why Corollary 4.10 was necessary as early as Chapter 4: it is the specific instance of Proposition 11.5 that shows up as soon as retention is examined concretely. Memory is a scalar resource, tradeable against time and energy in the usual ways complexity theory studies. Continuation, at the level of a single fiber, is not scalar at all; it is a binary fact — injective or not — fixed by the structure of G relative to that fiber, and no amount of the scalar resource changes it. Aggregated across many fibers, the breadth of what a system's retention preserves is measurable as $|B_t|$ (Definition 11.1), which does behave something like a quantity; but $|B_t|$ grows only by making G injective over more fibers, which is a design choice about what is stored, not a purchase made with additional storage capacity on its own.

11.5 Historical Observability, Defined

The term this monograph's title and abstract have used informally throughout can now be given a definition that ties directly to the audit apparatus of Chapter 6.

Definition 11.6 (Historical observability). Given a class Q of possible future admissibility judgments a system might need to make, its historical observability with respect to Q at time t is the proportion of Q whose judgments remain decidable given the current distinction budget B_t: the fraction of queries in Q that depend only on fibers still present in B_t, rather than on fibers that have permanently exited it.

Definition 11.6 makes precise what the audit procedure of Chapter 6 was already measuring in practice: the generating-input and fiber-injectivity tests are, in effect, a way of sampling Q and checking membership in B_t without needing to enumerate Q exhaustively. A system's historical observability is not a fixed property of its hardware or its raw storage capacity; by Theorem 11.4, it cannot be increased after the fact by any allocation of time, space, or energy once a relevant fiber has left B_t. It can only be preserved, at the moment of collapse, by the specific choice — Definition 7.3's Contraction and Soundness conditions, applied with an injective G at exactly the admissibility-relevant fibers Theorem 5.10 identifies — to keep it from leaving in the first place.

This is the sense in which continuation is not merely another resource alongside time, memory, and energy, but the resource that determines which of a system's future admissibility judgments remain answerable at all, independent of how much of the other three the system is given. A system may have unlimited time to think, unlimited space to store its thoughts, and unlimited energy to compute with, and still be unable to answer a question

whose answer depended on a distinction its own dynamics discarded before anyone thought to ask it. Continuation, in the sense this monograph has developed the term, is what stands between a system and that specific kind of failure, and Theorem 11.4 is the precise sense in which nothing else substitutes for it.

Chapter 12

Toward a General Theory of Admissible Externalization

Every use of Theorem 5.10 so far has been retrospective: given a system that already exists, audit its retention pattern against admissibility-relevance and classify what is found. This chapter turns the criterion around. If a designer is building a system from scratch, under a finite budget for retained history, what does the theory developed in Part II and Part III actually recommend, and how far beyond HYDRA and the Claude Code leak does the recommendation generalize?

12.1 From Audit Criterion to Design Principle

Theorem 5.10 states a necessary and sufficient condition for a retention architecture to support exactly the admissibility judgments a system's continuation families require, at minimum cost. Read prospectively, this is a specification: a designer who can enumerate the admissibility-relevant fibers (Definition 5.8) their system's dynamics will generate has, in principle, a target to build toward — injective history channels exactly there, and nowhere else.

The qualification "in principle" is doing real work. Real systems do not have unlimited capacity to build injective channels everywhere Theorem 5.10 would credit them for building one. The theorem is silent on what to do when the set of admissibility-relevant fibers exceeds what a bounded retention budget can cover, and that silence is this chapter's starting point.

12.2 The Allocation Problem

Definition 12.1 (Retention budget and candidate fibers). Let $F_1, \ldots, F_n$ be candidate fibers a system's designer has identified as plausibly admissibility-relevant, each with an estimated relevance weight $w_i > 0$ (reflecting the severity of the undecidability failure, Definition 6.3, that would result from under-retention at F_i) and a cost $c_i > 0$ of building an injective history channel over F_i. Given a total retention budget β, the allocation problem is to choose a subset $S \subseteq \{1, \ldots, n\}$ maximizing $\sum_{i \in S} w_i$ subject to $\sum_{i \in S} c_i \leq \beta$.

Remark 12.2 (This is not a redefinition of admissibility). Definition 12.1 introduces weights and a budget constraint, and it is worth being precise about what is and is not being optimized. Chapter 10 established that the admissibility relation C itself (Definition 5.1) requires no extremal structure to be well-defined, and nothing here revises that: C is fixed before this section begins, and $F_1, \ldots, F_n$ are fibers that Definition 5.8 — itself defined without reference to any weighting — has already picked out as relevant relative to that fixed C. What is being optimized is a downstream engineering decision: given that these fibers matter and capacity is limited, which to prioritize. This is a claim about resource allocation under a fixed admissibility structure, not a claim that admissibility is itself the output of an optimization, and Remark 10.6's distinction between a genuine variational structure and mere resource bookkeeping applies again here in the designer's favor: nothing about C has been asked to be smooth, comparable, or extremal.

Definition 12.1 is an instance of the knapsack problem, well known to be NP-hard in general (standard result; see Cormen, Leiserson, Rivest, and Stein, Introduction to Algorithms, or any comparable algorithms text treating weighted knapsack). This is stated not to claim a new complexity result but to be honest about what has and has not been achieved: the allocation problem has been given a precise combinatorial shape, which it did not have before this chapter, but a precise shape is not the same as a tractable one, and Section 12.5 returns to this directly.

12.3 A Design Principle: Admissibility-Aware Externalization

Where exact solution of Definition 12.1 is infeasible, the structure of the problem still licenses a design heuristic, in the same spirit as the greedy approximation ratio available for weighted knapsack: prioritize candidate fibers by relevance-per-cost, w_i / c_i, building injective channels for the highest-ratio candidates first until the budget is exhausted.

> Admissibility-Aware Externalization (design principle). Given finite retention capacity, a system should allocate history channels to admissibility-relevant fibers in order

of relevance per unit retention cost, rather than uniformly, rather than in order of raw reconstruction difficulty (Definition 3.1, which Proposition 4.13 already showed is the wrong axis), and rather than by implementation convenience.

This is stated as a design principle, not a theorem, because its force depends on the accuracy of the weights w_i and costs c_i a designer supplies, which Chapter 6's apparatus can help estimate — w_i via the severity implied by Definition 6.3 at F_i, c_i via the ordinary engineering cost of an injective G over F_i — but cannot guarantee are correct in advance of deployment. Corollary 5.11's two failure modes are exactly what a designer following this principle imperfectly will produce: under-retention where w_i was underestimated, over-retention where it was overestimated, both detectable after the fact by the audit procedure even when they were not foreseeable before it.

12.4 Generalization Beyond HYDRA

Every application so far — Chapter 7's formal apparatus, Chapter 8's audit, Chapter 11's resource-independence argument — has concerned computational systems narrowly construed: software architectures with explicit transition dynamics. HYDRA's own presentation makes a considerably broader claim, proposing that the same quartet of trajectory space, projection system, admissibility structure, and persistence functional recurs across cognition, biology, institutions, physics, and cosmology, with the Structural Universality Conjecture as its most general statement. This monograph does not attempt that scope. What it can state, at a scope matched to what Chapter 5 through Chapter 11 actually established, is narrower and specifically about retention.

Admissible Externalization Conjecture. Any persistent system operating under finite retention capacity, whose transition dynamics are generically non-injective (Definition 4.5) and whose continuations are governed by an admissibility structure in the sense of Definition 5.2, will — to the extent its retention architecture has been shaped by selective pressure toward correct future functioning rather than by accident — exhibit a retention pattern correlating with admissibility-relevance (Definition 5.8) rather than with raw reconstruction cost, and departures from this correlation will be observable, via the audit procedure of Chapter 6, as the specific failure signatures of Definition 6.3 and over-retention.

Following the classification this monograph has used throughout: this is a conjecture, not a theorem, and it is falsifiable in exactly the sense Chapter 6 made precise, applied now across domains rather than within one architecture. Three illustrative candidates, offered

as motivating instances rather than as evidence for the conjecture, suggest where a genuine test might be conducted. Legal and financial audit trails are retained selectively around transactions and decisions with downstream liability consequences, not uniformly across all record-generating activity, which is the pattern the conjecture predicts if liability determination is treated as an admissibility judgment. Version control systems retain commit history essentially unconditionally rather than selectively, which — if the conjecture is to survive this case — would need to be explained by the extremely low marginal cost of that retention ($c_i \approx 0$ for nearly all fibers) rather than by high w_i everywhere, an alternative the allocation problem (Definition 12.1) already accommodates, since low-cost, low-relevance fibers can still be worth including when β is not binding. Biological memory consolidation is reported in the relevant literature to favor emotionally salient or consequential events over routine ones, which would instantiate the conjecture if salience tracks something like admissibility-relevance to future decisions, though establishing that correspondence rigorously is well outside this monograph's scope and is flagged here only as a direction, not a claim.

12.5 What Remains Open

Four gaps are worth naming plainly, in keeping with this monograph's practice of distinguishing what has been established from what has only been proposed.

First, Definition 12.1's NP-hardness means the design principle of Chapter 12 is a heuristic with the usual gap between greedy approximation and exact optimality; nothing here establishes how large that gap is for retention-allocation problems specifically, as opposed to knapsack problems in general.

Second, the weights w_i and costs c_i the allocation problem requires are, in any real system, estimates rather than known quantities, and this chapter has not supplied a method for estimating them beyond gesturing at Chapter 6's apparatus.

Third, Section 7.5's open question — whether a system's hypothesis space $\mathcal{A}$ is fixed in advance or expands as evidence arrives — bears directly on the allocation problem too: a designer enumerating candidate fibers $F_1, \ldots, F_n$ in advance implicitly assumes the space of possible fibers is known at design time, which may fail for the same reasons Section 7.5 flagged for $\mathcal{A}$.

Fourth, and most directly, the Admissible Externalization Conjecture has been illustrated, not tested. Chapter 8 conducted one specification-level audit of one system; the conjecture's claim is about a pattern across systems and domains that this monograph has not attempted to survey. The conjecture is stated in a form intended to be testable by exactly that kind of survey, using the audit procedure already built, rather than in a form

that would require new theoretical machinery to evaluate — but the survey itself remains undone.

Chapter 13

Variational Primacy and the Missing Axis

Chapter 10 contrasted continuation geometry against Hamiltonian and Lagrangian mechanics as those frameworks are ordinarily taught, with Lagrangian mechanics presented as, in effect, an alternative derivation of Newton's laws. A natural objection is that this undersells the Lagrangian picture: in its most philosophically serious form, Lagrangian mechanics is not a derivation of anything more fundamental, but the more fundamental framework in its own right, with Newtonian mechanics as a restrictive special case. This chapter takes that objection seriously against an actual text that argues exactly this, checks whether the distinction Chapter 10 drew survives contact with the strongest available version of the variational picture, and finds a further, more precise point of contrast along the way that the weaker version of the comparison would have missed.

13.1 A Text That Already Resists the Naive Reading

José Rachid Mohallem's Lagrangian and Hamiltonian Mechanics: A Modern Approach with Core Principles and Underlying Topics (Springer, 2024) opens its preface with a single sentence set apart from the surrounding text: "In the beginning, there was the Action Principle." The book's stated aim is to correct a specific pedagogical habit — presenting the Lagrangian of a conservative system as $L = T - V$ from the outset and treating Hamilton's Principle as, in the author's words, an alternative way of obtaining the results of Newtonian Mechanics — in favor of treating the search for Lagrangians from symmetries, via Hamilton's Principle, as what the author calls the more powerful theoretical instrument of physics. This is not the textbook-standard framing Chapter 10 contrasted itself against; it is a considerably stronger position, treating variational selection as explanatorily prior to, rather than derivable alongside, Newtonian dynamics.

13.2 Confirming the Absence

If any single text were likely to have already developed something functionally close to this monograph's distinguishability axis, a philosophically self-aware treatment organized explicitly around Hamilton's Principle as primitive, rather than around $F = ma$, would be a reasonable place to look. A search of the complete extracted text of Mohallem's book — all 152 pages — for any of the terms irreversib-, reversib-, predecessor, or distinguishab- returns no occurrences whatsoever. This is reported as an observation about one text, not a survey of the literature on classical mechanics, and no claim is made here that no treatment anywhere engages a comparable question. What can be said is narrower and still useful: even a text that explicitly rejects the reduction of Hamilton's Principle to a mere alternative route to Newton's laws, and that spends a full chapter on the Hamiltonian formulation's own conceptual apparatus, organizes its entire discussion around two questions — which trajectory is selected, and what follows from a given state — and at no point asks which predecessor states remain distinguishable from a given successor. Table 10.1's third column is not competing with a question this text already asks under different vocabulary; the question does not appear.

Remark 13.1 (A closer near-miss than the bare search suggests). A lexical search of this kind is blunter than it looks, and it is worth naming the place it comes closest to failing. The book does discuss determinism directly, in a passage noting that classical mechanics' determinism of motion is among the paradigms replaced when passing to a quantum treatment. This is, in substance, an informal statement of state-sufficiency — the Hamiltonian side of Table 10.1 — under vocabulary the original four-term search did not anticipate. It does not, on inspection, extend to this monograph's question: the passage is entirely forward-looking, about what a state determines about the future, and at no point turns the question around to ask whether a state determines its past to the same degree. The distinction survives, but the search that first suggested it was cruder than it should have been, and a reader should take the absence of the four searched terms as a starting point for Remark 13.1's closer reading, not as a substitute for it.

13.3 Conservation Is Not Distinguishability

The comparison is worth pushing one step further than a text search, because the book's own central technical apparatus — constants of motion, arising from symmetries of the Lagrangian via cyclic coordinates — initially looks like it might be a variant of this monograph's distinguishability-preservation concept wearing different notation. It is not, and the precise way it is not is worth stating formally, because the two are not merely different but,

in a specific sense, structurally opposed.

Definition 13.2 (Constant of motion). A quantity $F(q, \dot{q}, t)$ is a constant of motion along a trajectory if its total time derivative vanishes identically along that trajectory: $\dot{F}(q, \dot{q}, t) = 0$. Constants of motion typically arise, via Noether's theorem in its classical-mechanical form, from a symmetry of the Lagrangian — a cyclic coordinate not appearing explicitly in L yields a conserved conjugate momentum.

Proposition 13.3 (Constants of motion are generically non-injective on trajectory space). Let F be a constant of motion and let $\mathcal{T}$ be the space of dynamically realizable trajectories. The level sets of F — the fibers of the map sending each trajectory to its (constant) value of F — generically contain more than one distinct trajectory: except in fully integrable systems with as many independent constants of motion as degrees of freedom, knowing the value of F alone does not determine which trajectory, among those sharing that value, is realized.

Justification. This is the ordinary content of partial integrability: a single constant of motion, such as energy, restricts a trajectory to an energy shell of the phase space, generally a submanifold of dimension one less than the full phase space, and this restriction is compatible with a continuum of distinct trajectories within that shell whenever the system has more than one degree of freedom and fewer independent constants of motion than are needed to fix a trajectory uniquely. □

Proposition 13.3 is the precise sense in which tracking a constant of motion is the structural opposite of what Definition 4.8 asks a history channel to do. A history channel earns its keep, in this monograph's framework, by being injective over an admissibility-relevant fiber — by separating predecessors that would otherwise be conflated. A constant of motion is, by construction, a coarsening: it is valuable to physics precisely because it is the same for many different trajectories, which is what makes it a useful invariant to reason with rather than a complete description of the motion. Where this monograph's history channels are built to discriminate, constants of motion are built to forget everything except one shared feature.

Remark 13.4 (Conservation tracking is reconstruction-cost compression). Corollary 10.2 already established that a Hamiltonian system's raw state never loses distinguishing information under its own dynamics: the flow map is bijective, so nothing is ever irreversibly lost at the level of (q, p) itself. An observer who tracks only a constant of motion is therefore never doing anything this monograph would classify as irreversibility materialization (Definition 4.12) — the full state remains, in principle, recoverable throughout, exactly as Remark 10.3 observed can remain true even when reconstruction is computationally severe.

Tracking F instead of the full state is a reconstruction-cost choice (Definition 3.1): a deliberate, tractability- motivated compression of a quantity that was never, in the relevant sense, at risk of becoming unrecoverable, because within a genuinely Hamiltonian regime nothing is. Classical conservation laws, read through this monograph's classification, sit entirely on the reconstruction-cost side of Table 10.1, never on the continuation side — which is a further reason Table 10.1's third column was not available to a text organized, however sophisticatedly, around constants of motion and variational selection: its central technical tool is, on this framework's own terms, an instance of the axis Proposition 4.13 already distinguished from the one this monograph is about.

13.4 What This Adds to Chapter 10

Table 10.1 stands as stated: the Hamiltonian column's claim that no history channel is ever required needs no revision, and neither does the Lagrangian column's claim that extremal selection is orthogonal to reversibility. What this chapter adds is a stronger warrant for the claim that continuation geometry's organizing question is not merely absent from the textbook presentation of classical mechanics, but absent from at least one serious attempt to present the subject with variational selection as fully primary rather than as a derived convenience — and that the reason it is absent is not oversight but structural: the tool this stronger presentation relies on most heavily, the constant of motion, answers a coarsening question, and Chapter 4's distinguishability question needed a different technical apparatus because it is asking in the opposite direction.

13.5 A Remaining Caveat

One book, however carefully argued, is one data point. Mohallem's text was consulted here because it was made available for exactly this comparison and because its stated thesis is unusually well-suited to testing whether Chapter 10's contrast survives the strongest available version of the objection. Nothing in this chapter should be read as a claim that no treatment of classical or analytical mechanics, anywhere, engages a question resembling this monograph's distinguishability axis; only that this one, chosen because it was likely to be the hardest case available, does not.

A specific and more promising place to look, named but not checked when this chapter was first drafted, turns out on investigation to matter more than a passing caveat can convey, and the earlier version of this section understated the case. Constrained Hamiltonian systems — those in which the Legendre transform from velocities to momenta is singular, so that the map fails to be invertible — are a well-known site of genuinely non-injective

structure inside classical mechanics itself, formalized in the Dirac–Bergmann theory of constraints. The kernel of the singular Legendre transform is, by construction, exactly a fiber in the sense of Definition 4.5: distinct velocity configurations mapping to the same momentum. This is a substantively different phenomenon from anything Corollary 10.2 addresses, since that result concerned the time-evolution flow map for a fixed, non-degenerate Hamiltonian, not the map between Lagrangian and Hamiltonian descriptions at fixed time. Mohallem's book does not treat this case. The wider literature does, at length, and the treatment is closer to this monograph's question than a first pass suggested.

13.6 The Problem of Time and the Problem of Observables

The standard resolution of a singular Legendre transform's non-injectivity is to declare it physically empty: points related by the transformations generated by the resulting first-class constraints are said to represent the same physical state, foliating phase space into "gauge orbits," and a Dirac observable is defined as a phase-space function invariant along these orbits — formally, one whose Poisson bracket with every first-class constraint vanishes. This is, in the vocabulary Chapter 7 already established, a projection: a map constant on the fibers induced by gauge equivalence, in exactly the sense Definition 5.4 used the word before this chapter existed to complicate it. Read this way, gauge theory's resolution of its own non-injectivity is to stipulate, at the level of the theory itself, that the collapsed distinctions were never physical to begin with — not to ask, as Definition 5.8 does, whether some downstream judgment depends on them.

This stipulation is not as settled as the standard presentation makes it sound, and general relativity is where it visibly breaks. Because the Hamiltonian of general relativity in its constrained form is itself a sum of first-class constraints, the standard argument implies the Hamiltonian generates gauge transformations rather than physical evolution, so that Dirac observables — gauge-invariant by construction — do not evolve with respect to it at all. This produces the problem of time: formally, nothing happens, while observationally, quantities such as cosmological redshift plainly do change. The literature's response has not converged. Some authors maintain the orthodox reading, that the Hamiltonian constraint is pure gauge and apparent evolution must be reconstructed some other way; others argue the identification of the Hamiltonian constraint as a pure gauge generator is mistaken — one recent treatment states directly that Dirac's own argument for this identification does not hold in general, following an extended line of criticism running through Bergmann, Kuchař, Pons, Salisbury, and Anderson. Whichever position is correct, the dispute itself is the relevant fact for this chapter: physicists working on exactly this problem are, in substance, asking whether a phase-space distinction collapsed by a constraint carries physical content

or does not, and have not settled it after decades of direct engagement.

Remark 13.5 (What is actually different). Stated this plainly, the problem of observables in constrained Hamiltonian systems is close enough to this monograph's question that Chapter 13's original claim — that no treatment of classical mechanics asks anything resembling it — was too strong, and is withdrawn in that form. What survives is a narrower and more precise distinction between the two questions, not their identity. The Dirac-observable criterion, $\{F, C_a\} = 0$ for every first-class constraint C_a, is a fixed, algebraic, theory-level test: a distinction either is or is not gauge, once and for all, independent of any downstream use. Definition 5.8's criterion is explicitly relative to a continuation family C (Definition 5.1): the same fiber can be admissibility-relevant under one C and irrelevant under another, as Section 8.5 demonstrated directly for the cross-session trust fiber. Physics is attempting to answer, for general relativity, a single context-free version of the question this monograph's apparatus only ever asks relative to a specified continuation family.

Remark 13.6 (A speculative connection, flagged as such). It is tempting, and worth stating explicitly as speculation rather than as anything this monograph establishes, that Remark 13.5's distinction may bear on why the problem of time has resisted decades of direct effort by researchers considerably more expert in the relevant physics than this monograph can claim to be. If Definition 5.8-style relevance is genuinely relative to a continuation family and has no context-independent fact of the matter in general, then a research program seeking a single, theory-level, context-free answer to "does this constraint-generated distinction matter" would be expected to encounter exactly the kind of persistent, foundational disagreement the literature shows, not because the physics is unusually difficult in the ordinary sense, but because the question, posed context-free, may not have the kind of answer being sought. This is offered only as a possible reframing, not a resolution; nothing in this monograph's apparatus is equipped to adjudicate a live dispute in the foundations of canonical general relativity, and no such adjudication is attempted here.

This qualifies Chapter 10 and the earlier sections of this chapter rather than undermining them. The axis Table 10.1 identifies is not simply absent from physics; it surfaces exactly where constrained Hamiltonian systems push hardest against the gauge-orbit resolution, in one of theoretical physics's most famously unresolved foundational problems. That the question turns out to be live rather than absent is, if anything, a stronger indication that it is a real question than the original, overstated version of this chapter claimed.

Chapter 14

Conclusion

The argument began with a narrow question: whether finding a parser inside Claude Code's leaked source told us anything about whether the model surrounding it understands code. Chapter 2 answered that question in one chapter and did not need the apparatus built afterward to do it. Everything from Chapter 3 onward exists because a more interesting question was sitting underneath the one the public debate was asking: not whether systems decompose into learned and non-learned parts, which was never seriously in doubt, but whether there is a principled account of which computations get externalized as retained structure and which do not. This chapter states what was actually established in pursuit of that question, classifies each claim by how much weight it can bear, and says plainly where the argument could turn out to be wrong.

14.1 What This Monograph Established

Four pieces of apparatus, built in sequence, carry the argument.

Chapter 4 separated two reasons a structure persists — reconstruction cost and reconstructive irreversibility — and showed, via the Historical Distinction Theorem (Theorem 4.9), that the two are not points on a shared scale: one concerns whether a generating input exists, the other concerns whether a retained channel is injective over a collapsed fiber, and no amount of the first substitutes for the second.

Chapter 5 made admissibility relational, in the sense HYDRA's own architecture requires, and proved the Selective Retention Theorem (Theorem 5.10): a system need only preserve distinguishability where some future admissibility judgment actually depends on it, not everywhere reconstruction happens to be impossible. Chapter 7 then gave the estimate side of this apparatus a precise treatment as hypothesis pruning, proved exactly when repair converges to the truth rather than merely stabilizing (Theorem 7.8), and exhibited

a constructed case where it does not.

Chapter 6 turned the theorem into something checkable: an independence criterion for admissibility-relevance (Definition 6.1) that does not consult a system's retention architecture to determine what that architecture should contain, an audit procedure built on it, and two named conditions under which the whole apparatus would be considered wrong for a given system rather than merely inconvenient.

Chapter 11 gave the argument its most general statement: the Resource Independence Theorem (Theorem 11.4), showing that no allocation of time, space, or energy substitutes for an injective history channel once its generating input is gone, which is the precise sense in which continuation is not a fourth member of the usual resource family but a different kind of thing.

Everything else — the leak audit, the HYDRA comparison, the transformer-semantics reframing, the classical-mechanics contrast, the design principle and its generalization — is these four results doing work against specific material: a real leaked architecture, a formal cognitive architecture, an open question in AI, and two carefully chosen physics texts.

14.2 The Status of These Claims

Not everything in this monograph carries the same evidentiary weight, and the classification below is offered so that no claim is mistakenly read as stronger than what was actually shown.

- Proved from stated definitions, close to tautological given them. The Historical Distinction Theorem, the Selective Retention Theorem, the Non-Reducibility and Resource Independence propositions, and the Convergence Characterization Theorem. Their proofs are short because their content is mostly already present in the definitions; their value, as stated repeatedly throughout, is organizational rather than surprising. None of these can be falsified by any observation, because none of them makes a claim about the world independent of the definitions that generate it.

- Framework definitions, not claims. Admissibility, continuation families, projections, generating inputs, fibers, distinguishability preservation, historical observability. These are not true or false; they are the vocabulary the rest of the monograph's claims are stated in, and their only defense is that they turned out to be usable — that real distinctions (compaction, checkpoints, trust state) could be sorted with them without the sorting collapsing into vacuity.

- An empirical claim about one real system, checked at reduced confidence. Chapter 8's audit of the Claude Code leak. This is the closest thing in the monograph to a test

against reality, and it was explicit throughout about being a specification-level audit — sampled, not exhaustive; drawing on public reporting, not source access; producing at least one case (context compaction) the audit could not cleanly resolve rather than forcing a resolution.

- A proposed protocol, not a result. Chapter 9's audit procedure for transformer semantics. Nothing in this monograph ran it. Its contribution is decomposing one blurred question into two checkable ones, not answering either.
- A conjecture, explicitly falsifiable, not tested beyond illustration. The Admissible Externalization Conjecture (Chapter 12). The candidate domains offered alongside it — audit trails, version control, memory consolidation — were named as motivating instances, not evidence.
- A comparison against specific, named sources, not a literature survey. Chapter 10 and Chapter 13's contrast with classical mechanics, checked against one textbook chosen for being the hardest available case, plus a follow-up check of the constrained-Hamiltonian-systems literature that partially retracted the chapter's original claim: the distinguishability question turns out to be live, under the vocabulary of gauge orbits and Dirac observables, in the unresolved problem of time in canonical general relativity, though the specific context-relative reframing Remark 13.6 offers remains explicitly speculative. The determinism passage named in Remark 13.1 remains a near miss rather than an instance of the question.

14.3 A Note on Terminology

One further piece of accounting belongs here rather than in any single chapter, because it concerns the whole document's vocabulary rather than any specific claim. Reconstructive irreversibility and irreversibility materialization are the names Chapter 4 gave the phenomenon this monograph studies, and they are the names that connect the technical apparatus back to the motivating question in the title. They are not, however, the vocabulary the load-bearing proofs are actually stated in. Chapter 5's Selective Retention Theorem and Chapter 7's convergence results — the two chapters doing the most proof-theoretic work in the book — are stated and proved entirely in terms of fibers, injectivity, and distinguishability preservation, and do not cite the irreversibility definitions directly at any point; neither does Chapter 9's protocol. The irreversibility vocabulary is concentrated instead in Chapter 4 itself, where it is defined; in Chapter 8, where it functions as a classification label for the audit table; and in Chapter 10 and Chapter 13, where it is the named category being contrasted against classical mechanics.

This is not a defect, and no revision follows from it: renaming the central term to track the apparatus precisely — something like "distinguishability materialization" — would be more technically accurate and would cost the document its connection to the question that motivated it in the first place, since "what survives reconstruction" is a question about irreversibility, asked in the vocabulary a reader arrives with, and the technical machinery of fibers and injectivity is this monograph's answer to how that question gets made precise rather than a replacement for it. The observation is recorded here because a careful reader is likely to notice the migration independently, and because noticing it is itself a small piece of evidence that Chapter 4's central move — replacing "can this be reconstructed" with "does this remain distinguishable" — was a genuine sharpening rather than a relabeling: the sharpened primitive is what the rest of the book actually needed, often enough that the original naming stopped being where the mathematics lived, even though it remains where the book's motivation does.

14.4 What This Monograph Does Not Establish

Stating the classification above plainly implies a list of things this monograph has not shown, and it is worth making the list explicit rather than leaving it to be inferred.

It has not shown that any real system's designers built retention architecture with admissibility-relevance in mind, as opposed to the retention pattern merely correlating with it for unrelated reasons — Section 8.8 named this limit directly and it was never closed. It has not shown that HYDRA's hypothesis-space apparatus satisfies the descending chain condition Proposition 7.5 requires, for HYDRA or for any other real implementation. It has not shown that any transformer is or is not functionally adequate, architecturally transparent, both, or neither, over any specific fiber; Chapter 9 built the instrument and did not use it. It has not surveyed institutional record-keeping, version control practice, or biological memory research seriously enough to count as evidence for Chapter 12's conjecture, only illustration of it. And it has not resolved the compaction case in Section 8.4, deliberately: that case was left open because the available evidence does not settle it, not because the apparatus was insufficient to state the question precisely.

14.5 How This Could Be Wrong

Three specific ways the monograph's non-tautological claims could fail, stated as concretely as the material allows.

The Admissible Externalization Conjecture fails if a survey of real systems — audit trails, version control, institutional record- keeping, or any other domain satisfying Chap-

ter 12's hypotheses — finds retention patterns statistically uncorrelated with externally tested admissibility-relevance (Definition 6.1), across enough systems that the uncorrelated cases cannot be dismissed as measurement noise. This is exactly Section 8.7's falsification-by- uncorrelated-retention condition, applied at the scale the conjecture actually requires rather than to the single system this monograph audited.

The claim that HYDRA's projection machinery offers a functionally adequate and architecturally transparent account of admissible computation fails if it turns out, on implementation, that architectural transparency of the kind Definition 9.2 describes is not achievable at any scale competitive with the systems it would need to compete against — if, in other words, every system capable of the functional adequacy Chapter 9 describes turns out to purchase it at the cost of exactly the opacity HYDRA's explicit hypothesis sets were meant to avoid. This monograph has not ruled this out; Section 7.5's open questions about fixed versus expanding hypothesis spaces are precisely where this failure would first appear.

The reconstruction-cost/reconstructive-irreversibility distinction itself fails as a general-purpose tool, independent of any specific conjecture built on it, if a significant class of real systems turns out to have retention patterns that neither classification captures cleanly — not because the audit was insufficiently thorough, the way Section 8.4's cases were merely difficult, but because a third kind of materialization exists that this monograph's two-part classification has no room for. Nothing in Part II through Part IV rules this out; the monograph's strongest claim is only that the two kinds identified so far are real and distinguishable, not that they are exhaustive.

14.6 Open Problems

Collected from where they were first raised: whether real hypothesis spaces satisfy the descending chain condition (Proposition 7.5); whether HYDRA's repair machinery operates over a fixed or expanding hypothesis space (Section 7.5); how large the gap is between the greedy allocation heuristic and exact solutions to Definition 12.1; how the weights and costs that heuristic requires would be estimated in a real deployment; whether a genuine replay-level audit of Claude Code, rather than this monograph's specification-level one, would confirm or overturn any row of Table 8.1; whether the compaction case in Section 8.4 is admissibility-aligned in the sense of Definition 8.3, which requires access this monograph did not have; whether Chapter 9's protocol, actually run, finds functional adequacy, architectural transparency, both, or neither, for any specific linguistic fiber; and whether the literature on constrained Hamiltonian systems has already developed something resembling this monograph's distinguishability axis under its own vocabulary, which Chapter 13 flagged and did not check; and, now that the check has been made and found the problem of time

and problem of observables to be the closest existing analogue, whether Remark 13.6's speculative reframing — that the dispute persists because a context-relative question is being asked in a context-free form — has any actual purchase on the physics, which would require expertise and engagement this monograph has not attempted.

14.7 Closing

The original public argument this monograph responds to treated the discovery of a parser as evidence against understanding. The rebuttal available from the outset — that architectural decomposition is not ontological decomposition, that syntax has long been understood to possess computational autonomy from semantics — was sufficient to dispose of that argument in one chapter, and did not require anything built afterward. What the remaining chapters pursued was a different question, one the original debate did not think to ask: not whether a system decomposes, but whether there is a principled account of what gets retained when it does, and whether that account can be checked against a real system rather than merely asserted about one. Wherever this monograph reached a result it could state as a theorem, the theorem tended to be close to definitional — a mark, if the argument in Chapter 6 was right, not of triviality but of having found the correct primitive to define. Wherever it reached a claim that was not definitional — the leak's actual retention pattern, the conjecture that the pattern generalizes, the question of what a transformer's internals actually do — it said so, and left the claim exactly as open as the evidence available for it left it. A theory of what survives reconstruction should not, on its own terms, claim more permanence for its conclusions than its evidence can support. This one has tried not to.

Appendix A

Notation and Standing Assumptions

This appendix collects, in one place, the notation introduced across Part II and Part III. It is a reference, not an introduction; every item below is defined properly at its first occurrence in the main text, cited here by label.

Symbol	Meaning
X, U, F	State space, input space, transition function of a transition system (X, U, F), $x_{t+1} = F(x_t, u_t)$ (Definition 4.4).
P	A set of admissible predecessor pairs $(x, u) \in X \times U$ under consideration for a given fiber.
$\mathsf{Fib}_P(y)$	The fiber of y under F restricted to P: $\{(x, u) \in P : F(x, u) = y\}$ (Definition 4.5).
s	A generating input: a state independent of execution history from which X is reproducible (Definition 4.1).
H, G	History space and history-update function of a history-augmented transition system, $h_{t+1} = G(h_t, x_t, u_t)$ (Definition 4.7).
$\mathsf{GenInput}(X, t)$	Indicator of whether a generating input for X remains accessible at time t (Definition 8.1).
$\mathcal{X}, C(x)$	Trajectory space and the continuation family assigning to each state its permitted continuations (Definition 5.1).
$\hat{\pi}_t, \hat{\Pi}$	A system's time-indexed admissibility estimate and the space of such estimates (Definition 5.4); distinguished from the canonical quotient π^*.
$\pi^*, \mathcal{A}, A^*$	The canonical admissibility projection, the hypothesis space of candidate admissibility structures, and the true structure, per the Minimal Projection Theorem (Definition 7.1).
$H_t \subseteq \mathcal{A}$	The hypothesis set surviving after repair through time t (Definition 7.2); H_∞ its eventual stable value (Proposition 7.5).
$R, \models$	Repair operator and fit relation between estimates and evidence (Definitions 5.5 and 5.6).
w_i, c_i, β	Relevance weight, retention cost, and total budget in the allocation problem (Definition 12.1).
B_t	A system's distinction budget at time t: collapsed fibers still recoverable at t (Definition 11.1).
S_G, w_G	The subset of fibers over which a compressive transition G is injective, and its admissibility-alignment status (Definition 8.3).

A.1 Standing Assumptions

Two assumptions recur across Part II and Part III and are stated here together rather than re-derived at each use.

- Well-foundedness where invoked. Any claim depending on Proposition 7.5 (chain stabilization) assumes the relevant hypothesis space satisfies the descending chain condition. This is flagged as an assumption, not a theorem, at every point it is used; Section 7.5 discusses when it can fail.
- Admissible inputs are exogenously fixed. The input space U of a transition system is treated as given rather than derived; nothing in this monograph offers an account of which inputs a system ought to accept, only of how retention should behave given a fixed admissibility structure over the inputs it does accept.

Appendix B

Full Proofs from Part II

The main text proves Theorem 4.9 and Proposition 4.6 in full; this appendix supplies the detail elided from Proposition 4.13's proof sketch and from Remark 10.3's chaotic-Hamiltonian illustration, in each case stating exactly which standard results are being invoked.

B.1 Non-Reducibility, in Full

Full proof of Proposition 4.13. Let X be materialized on reconstruction-cost grounds (Definition 3.1). By Proposition 4.3, there exists s satisfying Definition 4.1: s is independent of the system's execution history and some process P produces X from s without reference to any intermediate state. Let $\kappa : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ be any monotonic cost transformation — a change in the time, space, or energy charged for running P. Applying κ changes the number $\mathsf{cost}(P, s)$ but does not act on s or on P as mathematical objects: s remains a state independent of execution history, and P remains a process taking s to X. Hence s continues to satisfy Definition 4.1 after the transformation, for every κ, so X remains a reconstruction- cost case at every cost level. Since Definition 4.11 requires the absence of a generating input, and s's presence is invariant under κ, X never enters the reconstructive-irreversibility regime under any cost transformation.

Conversely, let a history channel h be materialized on reconstructive-irreversibility grounds over a fiber $\mathsf{Fib}_P(y)$ (Definition 4.12), so by Definition 4.11 no s satisfying Definition 4.1 exists for the predecessor's identity. Suppose, for contradiction, that some cost transformation κ brought X into the reconstruction-cost regime. This would require producing a generating input s' where none existed. But κ acts only on the domain of cost values assigned to existing processes; it is not itself a process, has no output in X's state space, and in particular is not a state independent of the system's execution history in the sense Definition 4.1 requires. No application of κ therefore manufactures s'. Contradiction; no

such κ exists. □

B.2 The Chaotic Hamiltonian Case, in Full

Remark 10.3 asserted that a chaotic Hamiltonian system can require exponentially growing reconstruction precision while remaining reconstructively reversible throughout. The precise statement:

Proposition B.1 (Exponential reconstruction cost under positive Lyapunov exponents). Let Φ_t be the flow of a Hamiltonian system with largest Lyapunov exponent $\lambda > 0$. Suppose a reconstruction procedure has access to $\Phi_t(y)$ only to finite precision ε_0 (measurement or representation error). Then the precision $\varepsilon(t)$ to which the predecessor $\Phi_{-t}(y)$ can be recovered by applying Φ_{-t} to the imprecise measurement degrades as

$$\varepsilon(t) \sim \varepsilon_0 \, e^{\lambda t},$$

so that recovering the predecessor to a fixed target precision ε^* requires initial measurement precision $\varepsilon_0 \sim \varepsilon^* e^{-\lambda t}$, decaying exponentially in t.

Justification. This is the standard definition of the largest Lyapunov exponent applied in reverse time: for nearby trajectories $\Phi_t(y)$ and $\Phi_t(y) + \delta_0$ with δ_0 small, the Lyapunov exponent governs $\|\delta_t\| \sim \|\delta_0\| e^{\lambda t}$ under forward flow for chaotic (positive-exponent) directions. Applying Φ_{-t} to a measurement perturbed by ε_0 evolves the perturbation under the time-reversed flow, which exhibits the same exponential separation in the reversed time direction whenever the forward flow is chaotic in the corresponding directions, giving $\varepsilon(t) \sim \varepsilon_0 e^{\lambda t}$ as claimed. □

Corollary B.2 (Cost and irreversibility remain independent under Proposition B.1). Proposition B.1 exhibits ε_0 (and hence reconstruction cost, however cost is measured against required precision) growing without bound as the target precision ε^* is held fixed and $t \to \infty$, while Corollary 10.2 guarantees $\mathsf{Fib}_P(y)$ remains a singleton for every t: the flow map's bijectivity (Proposition 10.1) does not depend on λ and holds regardless of whether the system is chaotic. The two quantities — required precision and fiber cardinality — are governed by unrelated properties of the dynamics (the Lyapunov spectrum versus the invertibility of Φ_t as a map), which is the precise sense in which Proposition 4.13 holds for this example: cost can be driven to infinity without the system ever entering Definition 4.11's regime.

Appendix C

The Hypothesis-Pruning Formalism: Extended Convergence Theory

Chapter 7 proved stability under a descending chain condition (Proposition 7.5) without giving a bound on how long stabilization takes, and proved convergence is not automatic (Theorem 7.8) without addressing what happens when $\mathcal{A}$ is infinite but still well-founded. Both gaps are closed here.

C.1 A Rate Bound for Finite Hypothesis Spaces

Proposition C.1 (Stabilization bound). If $\mathcal{A}$ is finite, $|\mathcal{A}| = N$, then the hypothesis sequence $H_0 \supseteq H_1 \supseteq \cdots$ stabilizes after at most $N-1$ strict contractions: there exists $T \leq N-1$ such that $H_t = H_T$ for all $t \geq T$.

Proof. Each strict contraction $H_t \supsetneq H_{t+1}$ removes at least one element of $\mathcal{A}$ from consideration, by Contraction (Definition 7.3). Since $|H_0| \leq N$ and every strict contraction strictly decreases $|H_t|$ by at least 1, there can be at most $N-1$ strict contractions before $|H_t|$ reaches its minimum possible value under Soundness, namely 1 (if the evidence stream is fully discriminating) or some larger fixed value (if it is not); in either case no further strict contraction is possible once H_t stops changing, so stabilization occurs within $N-1$ steps of the first repair event. □

This bound is tight: a hypothesis space in which each piece of evidence eliminates exactly one candidate realizes it exactly.

C.2 Well-Ordered Infinite Hypothesis Spaces

Proposition 7.5 assumed the descending chain condition, which finite $\mathcal{A}$ satisfies automatically by Proposition C.1 but infinite $\mathcal{A}$ need not. A weaker and still useful sufficient condition covers a natural class of infinite cases.

Proposition C.2 (Stabilization under well-ordering). If there exists a well-ordering $\prec$ of $\mathcal{A}$ such that Contraction always removes an $\prec$-minimal element of the currently remaining candidates that is inconsistent with new evidence (a "eliminate the least-preferred surviving candidate first" repair policy), then the hypothesis sequence stabilizes: there is no infinite strictly decreasing chain, by the defining property of a well-ordering, that every nonempty subset has a $\prec$-least element, which rules out an infinite descending chain of distinct nonempty subsets under this specific elimination policy.

Justification. This restates the descending chain condition's content for the specific case where $\mathcal{A}$ is well-ordered and the repair policy respects that order: a strictly decreasing chain of subsets of a well-ordered set, each obtained from the last by removing an element, corresponds to a strictly decreasing sequence of ordinals (the order type of what remains), and no strictly decreasing sequence of ordinals is infinite. □

Remark C.3. Proposition C.2 is not free: it requires the repair policy to respect a specific well-ordering, which is an assumption about how R is implemented, not a fact automatically true of an arbitrary repair operator satisfying only Definition 7.3. Repair policies that eliminate candidates in an order unrelated to any well-ordering of $\mathcal{A}$ are not covered, and Section 7.5's concern about whether HYDRA's actual implementation satisfies any such condition is not resolved by this appendix.

C.3 A Second Non-Convergence Family

Example 7.9 gave one construction with $|\mathcal{A}| = 2$. The construction generalizes.

Proposition C.4 (Non-convergence persists under enlargement). For any $N \geq 2$, there exists a hypothesis space $\mathcal{A}$ with $|\mathcal{A}| = N$, a true structure $A^* \in \mathcal{A}$, and an evidence stream such that H_t stabilizes at $|H_\infty| = N - 1 \geq 1$ without converging, whenever the evidence stream is silent on exactly one pairwise disagreement among the N candidates.

Construction. Let $\mathcal{X} = \{x_1, x_2, x_3\}$ as in Example 7.9, and let $\mathcal{A} = \{A^*, A_2, \ldots, A_N\}$ where A^* equates x_1 and x_2, each A_i for $i \geq 3$ disagrees with A^* on some classification unrelated to x_1, x_2 and is eliminated by evidence addressing that disagreement, and A_2 agrees with every A_i, $i \geq 3$ eliminated candidate on all points except x_1, x_2, where it distinguishes them,

matching Example 7.9's A' exactly. An evidence stream that addresses every disagreement except the A^*-versus-A_2 disagreement over x_1, x_2 eliminates $A_3, \ldots, A_N$ in finitely many steps (by Proposition C.1, since only finitely many candidates remain to eliminate) and stabilizes at $H_\infty = \{A^*, A_2\}$, reproducing Example 7.9's non-convergence at this larger scale. □

This shows Example 7.9 is not a boundary artifact of the smallest possible hypothesis space; non-convergence with stabilization survives arbitrary enlargement of $\mathcal{A}$, provided the evidence stream leaves even one pairwise disagreement permanently unaddressed.

Appendix D

The Allocation Problem: Complexity and Approximation

Chapter 12 named Definition 12.1 an instance of weighted knapsack and asserted NP-hardness by citation. This appendix gives the reduction and proves a concrete approximation guarantee for the greedy design principle of Chapter 12, closing the gap left by stating the heuristic without a bound on how far it can fall short of optimal.

D.1 NP-Hardness

Proposition D.1 (Definition 12.1 is NP-hard). The decision version of Definition 12.1 — given $F_1, \ldots, F_n$ with weights w_i, costs c_i, budget β, and a target W, does a subset S exist with $\sum_{i \in S} w_i \geq W$ and $\sum_{i \in S} c_i \leq \beta$ — is NP-hard.

Reduction from 0/1 Knapsack. The decision version of 0/1 knapsack is: given items with values v_i and weights ω_i, a capacity Ω, and a target V, does a subset exist with total value at least V and total weight at most Ω? This is NP-complete (Cormen, Leiserson, Rivest, and Stein, Introduction to Algorithms, standard result). Given any instance of 0/1 knapsack, construct an instance of Definition 12.1 by setting n equal to the number of knapsack items, $w_i := v_i$, $c_i := \omega_i$, $\beta := \Omega$, and $W := V$. A subset S satisfies the allocation problem's constraints for this instance if and only if it satisfies the knapsack instance's constraints, by construction; the reduction is computable in linear time. Since 0/1 knapsack is NP-complete, Definition 12.1 is NP-hard. □

D.2 A Proved Approximation Bound

Chapter 12's design principle recommended sorting candidates by w_i/c_i and taking them greedily until the budget is exhausted. The following makes precise how far this can fall short of optimal, and gives a simple fix with a proved guarantee.

Definition D.2 (Modified greedy allocation). Sort candidates so that $w_1/c_1 \geq w_2/c_2 \geq \cdots \geq w_n/c_n$. Let k be the largest index such that $\sum_{i=1}^{k} c_i \leq \beta$, and let $S_{\text{greedy}} = \{1, \ldots, k\}$. Let $j = k+1$ be the first candidate that does not fit. The modified greedy allocation selects $\arg\max\big(\sum_{i \in S_{\text{greedy}}} w_i,\ w_j\big)$: whichever of the greedy prefix or the single next-best candidate has higher total weight.

Theorem D.3 (Modified greedy is a 2-approximation). Let OPT denote the optimal value of Definition 12.1 for a given instance, and let ALG denote the value achieved by Definition D.2. Then $\text{ALG} \geq \text{OPT}/2$.

Proof. Consider the fractional relaxation of the problem, in which each candidate may be partially included. The optimal fractional solution OPT_f is achieved by taking candidates in decreasing order of w_i/c_i — exactly the sorted order above — filling the budget completely, taking a fractional amount of the first candidate that does not fit whole. This is a standard property of the fractional knapsack problem: greedy-by-ratio is optimal for the fractional relaxation. The fractional optimum therefore equals $\sum_{i \in S_{\text{greedy}}} w_i + \theta w_j$ for some $\theta \in [0, 1)$, where S_{greedy} and j are as in Definition D.2. Since $\text{OPT} \leq \text{OPT}_f$ (the integer optimum cannot exceed the fractional relaxation's optimum), and $\text{OPT}_f \leq \sum_{i \in S_{\text{greedy}}} w_i + w_j$ (bounding $\theta < 1$ by 1), we have

$$\text{OPT} \leq \sum_{i \in S_{\text{greedy}}} w_i + w_j \leq 2 \max\Big(\sum_{i \in S_{\text{greedy}}} w_i,\ w_j \Big) = 2 \cdot \text{ALG}.$$

Dividing by 2 gives $\text{ALG} \geq \text{OPT}/2$. □

Remark D.4 (What this adds to Chapter 12). Theorem D.3 upgrades the design principle from a heuristic with no stated guarantee to one with a proved worst-case bound: a designer following Definition D.2 — the ordinary greedy allocation, augmented with a single check against the best individual candidate — never does worse than half of what an omniscient allocator could achieve, regardless of how the weights and costs are distributed. This does not close Section 12.5's first gap entirely, since a factor of two is often too coarse a guarantee for a real deployment decision, but it replaces "a heuristic with an unknown gap" with "a heuristic with a proved gap of at most this size," which is the precise sense in which this appendix answers the question Section 12.5 left open rather than merely restating it.

Appendix E

Hamiltonian Flow: Existence, Uniqueness, and Liouville's Theorem

Chapter 10 cited standard ODE theory for Proposition 10.1 and stated Liouville's theorem without derivation. Both are supplied here.

E.1 Existence, Uniqueness, and the Flow Map

Theorem E.1 (Picard–Lindelöf / Cauchy–Lipschitz). Let $f : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ be Lipschitz continuous in its first argument, uniformly in the second, on a neighborhood of (z_0, t_0). Then the initial value problem $\dot{z} = f(z, t)$, $z(t_0) = z_0$ has a unique solution on some interval containing t_0.

This is classical and not reproved here; it is the standard existence- uniqueness theorem for ordinary differential equations. For a Hamiltonian system, $z = (q, p)$ and $f(z, t) = (\partial H/\partial p, -\partial H/\partial q)$, Lipschitz whenever H is C^2.

Proof of Proposition 10.1. Fix t. Uniqueness (Theorem E.1) gives that distinct initial conditions $z_0 \neq z_0'$ produce distinct solutions $\Phi_t(z_0) \neq \Phi_t(z_0')$ for as long as both solutions are defined: if $\Phi_t(z_0) = \Phi_t(z_0')$ for some t, then running the unique solution through $(\Phi_t(z_0), t)$ backward to time t_0 would have to recover both z_0 and z_0', contradicting uniqueness unless $z_0 = z_0'$. This gives injectivity of Φ_t. Surjectivity onto its domain of definition follows by running the flow forward from every point; smoothness of Φ_t and of its inverse Φ_{-t} follows from smooth dependence of solutions of an ODE on initial conditions, a standard strengthening of Theorem E.1 under C^1 (here C^2, since $H \in C^2$) hypotheses on f. □

E.2 Liouville's Theorem

Theorem E.2 (Liouville). Let Φ_t be the flow of a Hamiltonian vector field $X_H = (\partial H/\partial p, -\partial H/\partial q)$ on phase space M. Then Φ_t preserves the standard volume form: for any measurable region $\Omega \subseteq M$, $\mathrm{vol}(\Phi_t(\Omega)) = \mathrm{vol}(\Omega)$ for all t.

Proof. By the divergence theorem applied to the flow of a vector field, the rate of change of volume under the flow of X_H is governed by the divergence of X_H:

$$\frac{d}{dt}\mathrm{vol}(\Phi_t(\Omega)) = \int_{\Phi_t(\Omega)} \nabla \cdot X_H \, d\mathrm{vol}.$$

Compute the divergence directly:

$$\nabla \cdot X_H = \sum_i \left(\frac{\partial}{\partial q_i}\frac{\partial H}{\partial p_i} + \frac{\partial}{\partial p_i}\left(-\frac{\partial H}{\partial q_i}\right)\right) = \sum_i \left(\frac{\partial^2 H}{\partial q_i \partial p_i} - \frac{\partial^2 H}{\partial p_i \partial q_i}\right) = 0,$$

by equality of mixed partial derivatives (Clairaut's theorem, given $H \in C^2$). Since the divergence vanishes identically, the rate of change of volume is zero for every Ω and every t, so volume is preserved. □

Remark E.3 (Relation to Corollary 10.2). Theorem E.2 is the continuous, measure-theoretic strengthening of Corollary 10.2's pointwise claim. Bijectivity of Φ_t (Proposition 10.1) says no two points are ever identified by the flow; Liouville's theorem says, in addition, that the flow does not even locally compress or rarefy the space of possibilities — volume in phase space is neither created nor destroyed. Neither statement requires the system to be integrable or non-chaotic; both hold identically for the chaotic system considered in Proposition B.1, which is precisely why Corollary B.2 could assert unbounded reconstruction cost coexisting with zero information loss — volume preservation guarantees the information is never destroyed, while saying nothing about how finely that preserved volume becomes folded and stretched, which is exactly what governs reconstruction cost.

Appendix F

The Dirac–Bergmann Algorithm

Chapter 13 used the vocabulary of primary constraints, gauge orbits, and Dirac observables without deriving the algorithm that produces them. This appendix gives the construction in outline, precisely enough to support the claims Remark 13.5 makes about its logical form.

F.1 Primary Constraints from a Singular Hessian

Let $L(q, \dot{q})$ be a Lagrangian on configuration space with coordinates q^i, $i = 1, \dots, n$. The Legendre transform defines momenta $p_i := \partial L / \partial \dot{q}^i$. The transform is invertible for $\dot{q}^i$ in terms of (q, p) exactly when the Hessian

$$W_{ij} := \frac{\partial^2 L}{\partial \dot{q}^i \partial \dot{q}^j}$$

is non-singular. When $\det W = 0$, the Hessian has a kernel of some dimension $m > 0$: there exist m independent null vectors $v^i_{(a)}$, $a = 1, \dots, m$, with $W_{ij} v^j_{(a)} = 0$. Each null vector generates a relation among the momenta not involving the velocities, a primary constraint

$$\phi_a(q, p) \approx 0, \qquad a = 1, \dots, m,$$

where $\approx$ (weak equality) denotes equality only on the constraint surface, not as an identity on the full phase space — the distinction Dirac's formalism depends on throughout, since quantities that vanish weakly may still have nonzero Poisson brackets with other phase space functions.

F.2 Consistency and Secondary Constraints

The canonical Hamiltonian $H_c = p_i\dot{q}^i - L$ is well-defined on the constraint surface despite the non-invertibility, since the velocity-dependence of $p_i\dot{q}^i - L$ drops out in directions along $\ker W$. The total Hamiltonian is $H_T = H_c + \lambda^a\phi_a$, with λ^a undetermined Lagrange multipliers. Consistency of the primary constraints under time evolution, $\dot{\phi}_a = \{\phi_a, H_T\} \approx 0$, either determines some of the λ^a, is automatically satisfied, or generates a new, secondary constraint $\phi'_a \approx 0$ not implied by the primary constraints alone. The procedure iterates — checking consistency of each new constraint — until no further constraints are generated; this is the Dirac–Bergmann algorithm.

F.3 First- and Second-Class Constraints

Constraints are classified by the Poisson bracket algebra they generate on the constraint surface.

Definition F.1 (First- and second-class constraints). A constraint ϕ (primary or secondary) is first-class if its Poisson bracket with every other constraint vanishes weakly: $\{\phi, \phi_b\} \approx 0$ for all constraints ϕ_b. A constraint that is not first-class is second-class.

First-class constraints are, in the standard interpretation this monograph's Chapter 13 follows, the generators of gauge transformations: the transformation $\delta z = \epsilon\{z, \phi\}$ generated by a first-class constraint ϕ maps points on the constraint surface to other points on the constraint surface satisfying the same equations of motion, and is standardly interpreted as relating physically identical states — the gauge orbit structure Remark 13.5 discusses.

Second-class constraints, by contrast, do not generate gauge transformations; they are eliminated by passing to the Dirac bracket,

$$\{F, G\}^* := \{F, G\} - \{F, \chi_m\}(M^{-1})^{mn}\{\chi_n, G\},$$

where χ_m ranges over the second-class constraints and $M^{mn} := \{\chi_m, \chi_n\}$ is invertible on the second-class constraint surface by definition of second-class. The Dirac bracket restricts dynamics to the surface where the second-class constraints hold identically.

F.4 Dirac Observables

Definition F.2 (Dirac observable). A phase space function $\mathcal{O}$ is a Dirac observable if $\{\mathcal{O}, \phi_a\} \approx 0$ for every first-class constraint ϕ_a: it is invariant under every gauge transformation the theory admits.

This is the criterion Remark 13.5 contrasted directly with Definition 5.8: membership is decided by an algebraic condition against the full set of first-class constraints, fixed once the theory is fixed, with no reference to any downstream task, continuation family, or future admissibility judgment. Chapter 13's claim that this differs in kind from Definition 5.8's relativized criterion rests on exactly this contrast: $\{\mathcal{O}, \phi_a\} \approx 0$ is a single computation with a single answer, while Definition 6.1's counterfactual test is parameterized by a choice of continuation family that need not be, and in the disputed general-relativistic case discussed in Chapter 13 demonstrably has not been, agreed upon.

F.5 The Free Particle Example, Condensed

The reparametrization-invariant free relativistic particle, used in Chapter 13's source material as a worked illustration, has Lagrangian $L = \frac{1}{2n}\dot{q}^2 - \frac{1}{2}n$ with q^μ the particle's spacetime coordinates and n an auxiliary lapse variable. The momentum conjugate to n vanishes identically, $\pi = \partial L/\partial \dot{n} = 0$, giving one primary constraint directly. Consistency of $\pi \approx 0$ under time evolution generates the secondary constraint $p^2 + 1 \approx 0$ (the mass-shell condition), and the algorithm terminates: both constraints are first-class, since their mutual Poisson bracket vanishes, and together they generate the reparametrization gauge freedom of the model. This is the smallest nontrivial instance of the structure Chapter 13 scales up to general relativity, where the analogous constraints are the Hamiltonian and momentum constraints whose interpretation — pure gauge, or partly physical — is the substance of the dispute Remark 13.5 and Remark 13.6 describe.

Bibliography

[1] Anderson, E. (2017). The Problem of Time: Quantum Mechanics Versus General Relativity. Fundamental Theories of Physics, vol. 190. Cham: Springer.

[2] Arnold, V. I. (1989). Mathematical Methods of Classical Mechanics (2nd ed.). New York: Springer.

[3] Bergmann, P. G., & Komar, A. (1960). Poisson brackets between locally defined observables in general relativity. Physical Review Letters, 4, 432–433.

[4] Chomsky, N. (1957). Syntactic Structures. The Hague: Mouton.

[5] Chomsky, N. (1965). Aspects of the Theory of Syntax. Cambridge, MA: MIT Press.

[6] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3rd ed.). Cambridge, MA: MIT Press.

[7] Cover, T. M., & Thomas, J. A. (2006). Elements of Information Theory (2nd ed.). Hoboken, NJ: Wiley.

[8] Dirac, P. A. M. (1964). Lectures on Quantum Mechanics. Belfer Graduate School of Science Monographs Series No. 2. New York: Yeshiva University.

[9] Goswami, M. (2026, July 6). Claude Code, Chomsky, and the Illusion of the Wizard Behind the Curtain. LinkedIn. `https://www.linkedin.com/pulse/claude-code-chomsky-illusion-wizard-behind-curtain-mirtunjaya-goswami-qhr5c/`

[10] Kuchař, K. V. (1992). Time and interpretations of quantum gravity. In G. Kunstatter, D. E. Vincent, & J. G. Williams (Eds.), Proceedings of the 4th Canadian Conference on General Relativity and Relativistic Astrophysics. Singapore: World Scientific. Reprinted in International Journal of Modern Physics D, 20 (Supp. 1), 3–86 (2011).

[11] Lextrait, V. (2026). Behind Claude Code's Curtain: Generative Grammars Doing Heavy Lifting. LinkedIn. Cited as reported in Goswami (2026); the original post's exact wording could not be independently verified for this monograph.

[12] Mohallem, J. R. (2024). Lagrangian and Hamiltonian Mechanics: A Modern Approach with Core Principles and Underlying Topics. Cham: Springer.

[13] Pons, J. M., & Salisbury, D. C. (2005). The issue of time in generally covariant theories and the Komar–Bergmann approach to observables in general relativity. arXiv:gr-qc/0503013.