CONTRACT-GOVERNED MULTI-AGENT GRAPH ORCHESTRATION FOR LONG-HORIZON AUTONOMOUS RESEARCH PIPELINES

Anonymous authors
Paper under review

ABSTRACT

Long-horizon autonomous research pipelines require stronger guarantees than prompt-level coordination alone. We study a contract-governed orchestration framework in which research execution is modeled as a typed directed acyclic graph with validator-gated transitions, bounded corrective policies, and append-only provenance traces. We adopt validator-gated workflow patterns and protocol threat assumptions from prior orchestration and security work, and we define three optimization objectives for committed-prefix safety, bounded recovery, and security–provenance tradeoffs. Theoretical analysis establishes three results under explicit assumptions: committed-prefix soundness under compositional semantic and type guards, non-oscillation under finite corrective depth and strict potential descent, and false-acceptance contraction for covered attack classes under composite guards together with provenance invariance under hash-linked append-only traces. We connect these statements to audit experiments on a real orchestration substrate with symbolic checks and adversarial boundary tests. Evidence shows zero unsafe commits only in the intended compositional regime, recovery remains bounded only under finite-depth and descent conditions, and security gains hold for covered attack classes with uncovered-class failures reported as caveats. The result is a formal and empirical blueprint for auditable, extensible, and scientifically interpretable multi-agent research orchestration.

1 INTRODUCTION

Recent progress in agentic AI systems has shifted attention from isolated model calls to persistent, multi-step workflows in which planning, tool execution, validation, and revision are distributed across interacting agents (Song et al., 2026b; Chang & Geng, 2025; Sawant, 2026; Ferrag et al., 2025; Wei et al., 2025). This shift has improved practical capability, but it has also exposed a structural gap between empirical orchestration success and formally stated correctness conditions. In particular, long-horizon pipelines accumulate subtle state dependencies: a local validator decision can alter the reachable future graph, reroute/backtrack actions can affect provenance semantics, and security assumptions at one stage may not be preserved when branches reconverge. These problems are visible across workflow domains, including software engineering, scientific discovery, network management, and domain-specific industrial orchestration (Jin et al., 2025; Yamada et al., 2025; Pham et al., 2026; Wang et al., 2025b;a; Sun et al., 2025; Sun & Liu, 2025; Stewart & Buehler, 2025).

The central question of this report is not whether orchestration can be useful, but whether it can be made mathematically explicit and auditable under realistic long-horizon conditions. Existing systems provide strong design patterns: role decomposition, staged validation, and policy-controlled adaptation (Song et al., 2026b; Sawant, 2026; Chang & Geng, 2025; Marozau, 2025; Medeiros & Cavalcante, 2025). Security literature also shows that prompt-only filtering is insufficient when attack surfaces include protocol interactions, trust score manipulation, and cross-agent message dependencies (Ferrag et al., 2026; Ismail et al., 2025; Gao et al., 2025; Han & Choi, 2025). Formal-method studies provide useful contract-level reasoning templates, yet these are rarely integrated end-to-end with modern language-agent runtimes (Song et al., 2026a; Dinu et al., 2024; Glas & Morrow, 2026). The gap therefore sits at the interface of systems, formal analysis, and adversarial robustness.

This report develops a contract-governed graph semantics for autonomous research pipelines and evaluates the resulting claims through audit experiments on a concrete orchestration substrate. We treat the orchestration state as a typed graph process with hard validators, policy actions in $\{\text{continue}, \text{reroute}, \text{backtrack}, \text{branch}\}$, and immutable trace updates. We then formulate explicit objectives and feasible sets, derive assumption-qualified theorems, and connect those derivations to symbolic checks and boundary experiments.

Contributions.

- We define a contract-governed orchestration formalism with explicit objectives, decision variables, feasible sets, and optimality criteria for long-horizon autonomous research execution.
- We prove safety and recovery results under explicit assumptions, making clear which guarantees are inherited from prior conventions and which are newly introduced.
- We establish security and provenance results that combine protocol predicates and trust thresholds, including provenance invariance under append-only hash-linked traces.
- We provide empirical and symbolic evidence with explicit negative-result boundaries, so each major statement is linked to executed checks rather than narrative plausibility.

The broader relevance is cross-domain: any system that uses multi-agent decomposition over long horizons must reason about compositional safety, recovery boundedness, and evidence integrity (Xin et al., 2025; Reddy & Shojaee, 2025; Hou et al., 2025; Dong et al., 2025; Shao et al., 2025). By making those obligations explicit and testable, we aim to provide a reusable blueprint for rigorous research-as-a-service architectures.

An additional motivation is methodological clarity for scientific communication. Long-horizon autonomy papers often intermingle architectural novelty, empirical gains, and governance claims in ways that make falsification difficult. By separating objective definitions, theorem scopes, and evidence channels, we seek to reduce ambiguity about what was actually proved, what was only observed, and what remains conditional. This distinction is especially important for systems intended to support scientific workflows, where mistaken claims can propagate through downstream hypotheses, analyses, and report generation.

From a systems-engineering perspective, this framing also clarifies deployment decisions. Teams can map each guarantee to concrete preconditions before enabling autonomous branching in production workflows. If validator composition assumptions are met, committed-prefix safety theorems justify stronger automation policies; if those assumptions are weakened, the same formalism predicts failure modes and motivates stricter escalation gates. Likewise, recovery guarantees under bounded corrective depth provide practical criteria for watchdog controllers that must decide whether to continue local correction, reroute to a specialized role, or trigger human intervention. The contribution links theorem validity, runtime observability, and governance policy in a single auditable path from claims, to equations, to code-level checks, and finally to published report statements.

2 Background and Related Work

This section situates the report in five connected literatures: workflow orchestration, symbolic-contract formalisms, protocol security, autonomous scientific discovery systems, and audit-driven reliability methodology. The goal is not exhaustive coverage, but grounding for the formal objects introduced later. Each subsection identifies what prior work establishes, what remains under-specified for long-horizon multi-agent settings, and how those gaps map to the theorem obligations analyzed in this report. We use this structure to keep citation use disciplined: prior work informs assumptions and threat models, while the objectives and proofs provide the claim core. This avoids conflating literature consensus with theorem guarantees and keeps responsibility for new claims explicit. Across domains, contemporary orchestration systems increasingly share one architectural pattern: they decompose decision labor across specialized roles, route artifacts through explicit stage boundaries, and preserve state over long horizons to enable continuity and audit (Song et al., 2026b; Chang & Geng, 2025; Wang et al., 2025b; Sun et al., 2025; Hou et al., 2025; Zhang et al., 2025; Dong et al., 2025). This convergence is important because it creates a common vocabulary for comparing methods that otherwise target different end goals. However, shared architecture does not imply shared guarantees. Many systems demonstrate practical gains while leaving formal scope implicit, especially when adaptive branching, rerouting, and rollback are active.

Formal-method and symbolic-control sources suggest reading this space through three layers. The first layer asks whether local transition conditions are explicit and checkable. The second asks whether those local conditions compose into global statements over committed traces. The third asks whether provenance remains invariant under corrective control so that downstream scientific claims stay auditable (Song et al., 2026a; Sawant, 2026; Dinu et al., 2024; Glas & Morrow, 2026). Existing papers frequently cover one or two layers well, but fewer cover all three. This view motivates our report design: we separate local safety, bounded recovery, and security–provenance arguments, then reconnect them through shared objectives and feasibility constraints.

Security literature further clarifies why this separation is necessary. Protocol exploit analyses show that prompt-level hardening can be bypassed by interaction-level attacks, and trust-only acceptance rules can retain high false-accept risk

in specific classes (Ferrag et al., 2026; Ismail et al., 2025; Gao et al., 2025; Ferrag et al., 2025). Governance-oriented proposals add stronger audit and policy controls, yet they often stop short of formal composition claims for heavily branching corrective execution (Glas & Morrow, 2026; Xin et al., 2025; Reddy & Shojaee, 2025). Without explicit theorem scopes, robustness improvements can be overgeneralized. We avoid that error by making coverage predicates and boundary classes explicit.

A parallel challenge is objective mismatch. Discovery-oriented systems prioritize exploration throughput, whereas governance-oriented systems prioritize constraint satisfaction and trace accountability (Hou et al., 2025; Zhang et al., 2025; Dong et al., 2025; Glas & Morrow, 2026; Reddy & Shojaee, 2025). Both priorities are valid but induce different cost functions and different acceptable failures. Literature that blends them narratively without formal objectives is difficult to falsify. Our approach is to encode these tensions directly through objective definitions and weighted terms, then evaluate each claim against its own metric rather than one aggregate utility.

This framing sets the novelty boundary for the report. We do not claim novelty in role decomposition, validator gating, provenance logging, or threat awareness themselves; these are established in prior work (Song et al., 2026b; Sawant, 2026; Ferrag et al., 2026; Glas & Morrow, 2026). The contribution is narrower and testable: a contract-governed optimization formalism with explicit assumptions, proofs, symbolic checks, and counterexample regimes on a real orchestration substrate. The following subsections synthesize the most relevant clusters and identify the contradiction points that motivate our theorem family.

2.1 Workflow Orchestration and Role Decomposition

Role-specialized orchestration is now a dominant pattern in agentic systems. Paper-writing and planning frameworks report improvements from separating planning, execution, and review responsibilities (Song et al., 2026b; Chang & Geng, 2025; Marozau, 2025; Medeiros & Cavalcante, 2025). Infrastructure-oriented systems similarly use explicit coordinator modules to mediate task routing and policy selection (Wang et al., 2025b; Sun et al., 2025; Ihaddaden, 2025). These works provide strong evidence that decomposition improves operational control. Their limitation is that correctness is usually reported as aggregate success rates rather than compositional transition guarantees.

Context and transaction management studies move closer to formal control. Saga-style orchestration and validator-gated phase transitions introduce rollback semantics, intermediate checkpoints, and state consistency targets (Chang & Geng, 2025; Sawant, 2026). These patterns motivate our model of hard validator conjunctions and bounded corrective actions. However, most available results remain tied to methods or implementations: they establish useful mechanisms but not complete statements about committed-prefix soundness under explicit assumptions.

2.2 Formal and Symbolic Foundations

The formal-method lineage provides key guidance for contract-based reasoning in orchestrated systems. Tool orchestration methodologies in formal software contexts show how interface contracts and proof obligations can be propagated through heterogeneous execution layers (Song et al., 2026a). Symbolic language constraints and admissibility filters provide additional structure for mapping unstructured model output into verifiable objects (Dinu et al., 2024). Inference-time governance studies further emphasize provenance-aware runtime controls (Glas & Morrow, 2026).

The strength of this cluster is precise semantics. The limitation is deployment integration: symbolic and contract reasoning are often not coupled to long-horizon agent pipelines where branching, dynamic rerouting, and adversarial perturbations are common. Our approach builds on these ideas but links theorem statements to audit experiments.

2.3 Security, Trust, and Protocol Robustness

Protocol-level threat models show that adversarial failures in multi-agent workflows cannot be reduced to prompt hygiene (Ferrag et al., 2026; Ismail et al., 2025). Trust-based coordination models contribute useful probabilistic decision criteria but can be brittle when trust surrogates are manipulable (Gao et al., 2025; Han & Choi, 2025). Security orchestration work in operational settings reinforces this point: control-theoretic or rules-based gate composition can reduce false acceptance in covered regimes, but unmodeled threat classes still dominate residual risk (Ismail et al., 2025; Sun & Liu, 2025; Wang et al., 2025a).

Our security formulation therefore treats trust and protocol predicates as jointly necessary and formally separable. This design choice is directly motivated by contradictions between optimistic quality-centric workflow reports and adversarial analyses that show protocol exploits can bypass purely trust-based gating (Ferrag et al., 2026; Gao et al., 2025; Ismail et al., 2025; Ferrag et al., 2025).

2.4 Agentic Science Context and Gap Synthesis

Agentic-science surveys and domain studies highlight a central need: reliability and auditability must scale with autonomy depth (Xin et al., 2025; Reddy & Shojaee, 2025; Wei et al., 2025). Domain systems in chemistry, biomedical analysis, and fluid dynamics show practical feasibility of automated multistage pipelines (Pham et al., 2026; Zhang et al., 2025; Klang et al., 2025; Dong et al., 2025; Stewart & Buehler, 2025). Yet these systems rarely provide formal bounds for heavily branching orchestration dynamics. The gap is a lack of compositional guarantees that remain valid under recovery policies, adversarial inputs, and provenance constraints.

This report addresses that gap with a formal core of explicit assumptions and proofs, supported by audit experiments and symbolic checks that expose both supported regimes and failure boundaries.

2.5 Contradiction Map and Contribution Boundary

Three contradictions in the literature motivate the report's structure. The first is a *proof contradiction*: systems papers report robust coordination improvements, yet formal closure for global transition safety is often absent (Song et al., 2026b; Chang & Geng, 2025; Wang et al., 2025b). This makes it difficult to separate principled guarantees from dataset-specific empirical luck. We address this by defining global objective functions and proving only what can be proved under named assumptions.

The second is a *security contradiction*. Workflow studies frequently emphasize quality checks, while threat analyses show that protocol exploits can bypass prompt-centric defenses and trust-only gates (Ferrag et al., 2026; Ismail et al., 2025; Gao et al., 2025). We therefore encode security claims as conditional statements over covered attack classes and treat uncovered-class failures as expected boundary evidence rather than anomalies.

The third is an *audit contradiction*. Provenance is widely discussed as necessary for trustworthy autonomy (Glas & Morrow, 2026; Xin et al., 2025; Reddy & Shojaee, 2025), but many practical systems lack formal invariance guarantees when corrective branching is active. We include a dedicated recurrence-based provenance theorem to close this gap at the formal level.

These contradictions define the novelty boundary. We do not claim a universal solution to autonomous science reliability. We claim a narrower contribution: a contract-governed orchestration formalism where each major claim has an explicit assumption set, a proof path, an executable check, and a concrete counterexample regime.

3 Problem Formulation and Notation

We now define the orchestration object, state-transition semantics, and assumption blocks used by all theorem statements. The emphasis is precision: each symbol is tied to an executable transition condition and each assumption block is structured so that premise removal has a predictable failure signature. This formulation makes it possible to compare proofs and runs on the same model instead of treating empirical diagnostics as post hoc support. A second objective is portability. By expressing transitions and assumptions in reusable algebraic form, the framework can be transplanted to adjacent domains without rewriting the reasoning vocabulary from scratch. This portability is important for future cross-domain validation promised in the later sections.

3.1 Contract-Governed Orchestration System

We model orchestration as

$$Q = (G, \mathcal{C}, \Phi, \Pi, \mathcal{M}, \mathcal{T}), \tag{1}$$

where $G = (V, E)$ is a finite directed acyclic phase graph with V finite and $E \subseteq V \times V$, $\mathcal{C} = \{C_e\}_{e \in E}$ is an edge-contract family, Φ is a validator family, Π is a meta-orchestration policy, $\mathcal{M}$ is persistent memory, and $\mathcal{T}$ is an append-only execution trace. The system is discrete-time and finite-horizon unless explicitly stated otherwise: $\mathbb{N}_0 = \{0, 1, 2, \ldots\}$, $H_{\mathrm{hor}} \in \mathbb{N}_{\geq 1}$, and empirical objectives use $t \in \{0, \ldots, H_{\mathrm{hor}} - 1\}$. For an edge $e = (u, v)$, we use

$$C_e = (\tau_e^{\mathrm{in}}, \tau_e^{\mathrm{out}}, \mathrm{pre}_e, \mathrm{post}_e), \tag{2}$$

where type signatures $\tau_e^{\mathrm{in}}, \tau_e^{\mathrm{out}}$ are labels for admissible typed artifact sets and predicates $\mathrm{pre}_e, \mathrm{post}_e$ encode transition admissibility. The artifact semantics used in the formal development are typed sets equipped with Boolean admissibility predicates and scalar audit metrics, giving a finite typed transition system over committed traces.

We adopt validator-gated transition conventions from prior workflow literature (Sawant, 2026; Chang & Geng, 2025; Song et al., 2026a) and define three optimization objectives, each aligned to one scientific claim family: (i) committed-prefix safety, (ii) bounded recovery, and (iii) security–provenance tradeoff. The tuple in equation 1 is intentionally

Table 1: Formal applicability scope for symbols and claims. Domains, ranges, and named assumption blocks are part of the claim contract for each theorem statement.

Object	Domain or range	Applicability regime
$G = (V, E), e = (u, v)$	V finite, $E \subseteq V \times V$, G acyclic; $e \in E$.	A4 supplies a topological commit order for committed-prefix induction.
$\mathcal{S}, s_t$	$s_t \in \mathcal{S}$, where $\mathcal{S}$ is the set of typed graph, memory, policy-context, and trace configurations.	The audit model advances one transition at a time in discrete time $t \in \mathbb{N}_0$.
$\mathcal{A}, a_t, \Pi$	$\mathcal{A} = \{\texttt{continue}, \texttt{reroute}, \texttt{backtrack}, \texttt{branch}\}$ and $a_t \in \mathcal{A}$; $\Pi : \mathcal{S} \to \Delta(\mathcal{A})$ for stochastic policies, with deterministic policies as point masses.	Expectations are taken on a finite probability space $(\Omega, \mathcal{F}, \mathbb{P})$ induced by seeds, policy randomization, and workload draws.
$y_t, \Phi_t, \phi_e^{\text{hard}}$	$y_t \in \mathcal{Y}_{e_t}$ for the active edge output space; $\Phi_t \in \{0,1\}^{m_t}$; $\phi_e^{\text{hard}} : \mathcal{Y}_e \to \{0,1\}$.	Safety claims use A1–A3 with semantic and type hard guards; schema-only, trust-only, and disabled-hard-guard settings are recorded as boundary probes.
$\text{pre}_e, \text{post}_e$	Boolean predicates on typed edge inputs, outputs, and state context.	A3 applies monotonicity to hard safety predicates over committed prefixes.
$J_1, \text{commit}_t, \text{Safe}$	$J_1 \in \mathbb{N}_0$; $\text{commit}_t, \text{Safe}(s_t) \in \{0,1\}$.	The zero optimum is over $\mathcal{F}_1$ under A1–A4 and counts committed transitions.
$B, \delta, \eta, V(s_t), c$	$B \in \mathbb{N}_{\geq 1}$; $\delta \in \mathbb{R}_{>0}$; $\eta \in \mathbb{R}_{\geq 0}$ with experimental theorem-regime runs using $\delta > \eta$; $V : \mathcal{S} \to \mathbb{R}_{\geq 0}$; $c : \mathcal{A} \times \mathcal{S} \to [0, c_{\max}]$.	B1–B4 specify finite corrective depth, strict descent, bounded costs, and rollback-comparator alignment. Stress runs perturb these settings to generate boundary evidence.
$\theta, \kappa, g_{\theta,\kappa}$	$\theta \in [0,1]$; κ indexes protocol predicates; $g_{\theta,\kappa} : \mathcal{Y} \to \{0,1\}$.	FAR contraction is a set-inclusion result for a common distribution on covered attack classes. Coverage diagnostics are reported separately for uncovered or shifted classes.
FAR, FRR, ProvLoss	Rates in $[0,1]$ estimated from finite class-stratified samples; $\alpha, \beta, \gamma \in \mathbb{R}_{\geq 0}$ with $\alpha + \beta + \gamma > 0$.	Threshold tradeoffs are finite-sample audit rates for the executed protocol.
r_t, h_t, Hash	$r_t \in \mathcal{R}$, $h_t \in \mathcal{D}$, and $\mathsf{Hash} : \mathcal{D} \times \mathcal{R} \to \mathcal{D}$.	C2–C4 specify append-only records, previous-hash links, guarded commits, and the operational collision-resistance assumption. Mutation probes test trace-integrity diagnostics.

minimal: every coordinate maps to an executable component used during audit runs. This mapping matters because later contradiction tests toggle assumptions at the level of contracts, validators, and policy actions, so theoretical premises and executable diagnostics remain synchronized.

3.2 Standing Assumptions

We use three assumption blocks.

Safety block (A1–A4): typed normalization preserves validator-relevant semantics; validator acceptance implies downstream preconditions; hard safety predicates are monotone over committed prefixes; commit order follows graph topology.

Recovery block (B1–B4): corrective depth is bounded by $B \in \mathbb{N}_{\geq 1}$; corrective actions decrease a potential $V : \mathcal{S} \to \mathbb{R}_{\geq 0}$ by at least $\delta \in \mathbb{R}_{>0}$; stage costs $c : \mathcal{A} \times \mathcal{S} \to [0, c_{\max}]$ are nonnegative and bounded for $c_{\max} \in \mathbb{R}_{\geq 0}$; a rollback comparator policy is behaviorally comparable.

Security/provenance block (C1–C4): protocol predicates cover declared attack classes; trace updates are append-only; each record hash-links to the previous committed record; mandatory guard checks cannot be bypassed before commit.

These assumption blocks define the validity regimes for the theorem statements and audit diagnostics.

3.3 Formal Applicability Scope

This subsection records the mathematical setting used by the formal statements, objectives, metrics, and reported evidence. The proofs use finite typed state spaces, Boolean predicates, topological orders on finite graphs, finite-horizon sums, and finite-probability expectations over seeds, workloads, and policy randomization.

3.4 Role Semantics and State Transition Algebra

The graph-level model is paired with role semantics for three agent classes: an orchestrator that updates global state and policy context, an executor that performs phase-local transformations under contracts, and a scientist role that proposes remediation or decomposition changes when uncertainty rises. This role separation is common in modern systems (Song et al., 2026b; Marozau, 2025; Medeiros & Cavalcante, 2025), but here it is formalized through transition operators rather than narrative modules.

Let s_t denote global state at step t, and let $a_t \in \{\texttt{continue}, \texttt{reroute}, \texttt{backtrack}, \texttt{branch}\}$ be the policy action. We define update operator

$$s_{t+1} = \mathcal{U}(s_t, a_t, y_t, \Phi_t, \mathcal{M}_t), \tag{3}$$

where $y_t \in \mathcal{Y}_{e_t}$ is normalized output on the active edge, $\Phi_t \in \{0,1\}^{m_t}$ is the validator outcome tuple, and $\mathcal{U} : \mathcal{S} \times \mathcal{A} \times \mathcal{Y}_{e_t} \times \{0,1\}^{m_t} \times \mathcal{M} \to \mathcal{S}$ is defined only for typed inputs satisfying the edge contract. Admissibility and commit rules are

$$\mathcal{A}(s_t, a_t, y_t) = 1 \Rightarrow \mathrm{CommitAllowed}(s_t, a_t, y_t), \tag{4}$$

$$\mathrm{CommitAllowed}(s_t, a_t, y_t) \Rightarrow \mathcal{T}_{t+1} = \mathcal{T}_t \| r_t. \tag{5}$$

Equations equation 3–equation 5 formalize validator-gated conventions (Sawant, 2026; Chang & Geng, 2025; Song et al., 2026a). The admissibility implication in equation 4 is the hard-gate link between role-level policy actions and commit eligibility, so every downstream safety argument can be traced to a single transition predicate rather than to informal runtime heuristics.

This algebra resolves an ambiguity common in workflow reports: whether corrective actions are external exceptions or intrinsic transitions. In our formulation they are intrinsic, which is necessary for finite-recovery proofs and provenance invariance under adaptive control.

The algebra also clarifies optimization responsibility. Policy creativity does not imply unconstrained commits: actions from Π remain subordinate to hard validators and contract admissibility. This asymmetry is central to why committed-prefix safety can be proved in section 5 and falsified in controlled boundary runs.

The recurring symbols used below are introduced in place rather than collected in a separate notation table. Graph and contract primitives are defined in equation 1–equation 2; validator outputs, policy actions, and trace updates are defined in equation 3–equation 5; and objective quantities are defined alongside J_1, J_2, and J_3 in the method section. This keeps each symbol close to its first mathematical use while Table 1 provides the domain, range, and applicability information needed for proof checking.

4 Method: Objectives, Feasible Sets, and Policy Structure

The method section converts the system model into three optimization objectives aligned to the three claim families: committed-prefix safety, bounded recovery, and security/provenance control. For each objective we state a feasible set that makes theorem premises explicit and testable. The policy block then specifies how these constraints are enforced during execution, so formal proofs and runtime behavior share the same control semantics. This section also explains how to read the evidence. Every metric in the audit protocol is mapped to one objective or feasibility condition, so numeric outcomes can be read as evidence for or against stated premises rather than as generic performance indicators.

4.1 Objective for Committed-Prefix Safety

For finite horizon $H_{\mathrm{hor}} \in \mathbb{N}_{\geq 1}$, we define

$$J_1(\Pi, \Phi^{\mathrm{hard}}) = \sum_{t=0}^{H_{\mathrm{hor}}-1} \mathbf{1}[\mathrm{commit}_t \wedge \neg \mathrm{Safe}(s_t)], \tag{6}$$

with feasible set

$$\mathcal{F}_1 = \left\{ (\Pi, \Phi^{\mathrm{hard}}) : \mathrm{commit}_t \Rightarrow \bigwedge_{e_t \in E_t} \phi_{e_t}^{\mathrm{hard}}(y_t) = 1 \right\}. \tag{7}$$

The optimality criterion is

$$(\Pi^\star, \Phi^\star) \in \arg \min_{(\Pi, \Phi^{\mathrm{hard}}) \in \mathcal{F}_1} J_1(\Pi, \Phi^{\mathrm{hard}}). \tag{8}$$

Equation equation 8 states the optimization target for the safety objective, while equation 6 is used in section 5 to prove a zero-unsafe-commit optimum under A1–A4 and in section 7 to interpret boundary failures when assumptions are disabled. This objective intentionally measures only committed-prefix violations, not all intermediate policy proposals. The separation matters because branch exploration can include unsafe candidates that are never committed. By scoring only committed transitions, J_1 aligns with the safety claim actually proven in Theorem 5.2 and avoids conflating exploratory search behavior with accepted-state integrity. This design choice also sharpens comparator interpretation. A baseline can appear competitive if it produces fluent intermediate candidates, yet still be unsound if even a small fraction of unsafe states enter the committed prefix; equation 6 ensures those failures dominate the objective exactly where the formal claim places emphasis.

4.2 Objective for Bounded Recovery

We define

$$J_2(\Pi) = \mathbb{E}\left[\sum_{t=0}^{H_{\text{hor}}-1} c(a_t, s_t)\right] + \lambda\,\mathbb{E}[N_{\text{viol}}], \tag{9}$$

with feasibility constraints

$$\mathcal{F}_2 = \{\Pi :\ N_{\text{rb}}(\omega) \le B\ \forall \omega \in \Omega,\ V(s_{t+1}) \le V(s_t) - \delta \text{ for every corrective transition } t\}\,. \tag{10}$$

Here $N_{\text{rb}} \in \mathbb{N}_0$ counts reroute/backtrack actions, $N_{\text{viol}} \in \mathbb{N}_0$ counts unresolved violations, $B \in \mathbb{N}_{\ge 1}$ is the configured corrective-depth bound, $\lambda \in \mathbb{R}_{\ge 0}$ is the violation-penalty weight, $V : \mathcal{S} \to \mathbb{R}_{\ge 0}$ is a scalar recovery potential distinct from the node set V of G, and $\delta \in \mathbb{R}_{>0}$ enforces strict potential descent. The expectation is over the finite probability space $(\Omega, \mathcal{F}, \mathbb{P})$ induced by seeds, workload draws, and any stochastic policy choices. Equation 9 and equation 10 are used to derive non-oscillation and bounded correction length in section 5. The constraints in $\mathcal{F}_2$ are paired deliberately: bounded corrective depth alone does not prevent short-cycle oscillation, and descent alone does not cap episode length under repeated branching. Together they enable a finite-horizon guarantee. The experimental protocol toggles these conditions separately to verify that each premise has distinct and observable impact. The weight λ is a governance penalty on unresolved violations. Increasing λ prioritizes rapid contradiction closure over exploratory branching, while decreasing λ permits broader search at the cost of longer correction trajectories.

4.3 Objective for Security and Provenance

For trust threshold θ and protocol policy κ, we define

$$g_{\theta,\kappa}(y) = \mathbf{1}[\text{trust}(y) \ge \theta] \cdot \mathbf{1}[\psi_\kappa(y) = 1], \tag{11}$$

$$J_3(\theta, \kappa) = \alpha\,\text{FAR}(\theta, \kappa) + \beta\,\text{FRR}(\theta, \kappa) + \gamma\,\text{ProvLoss}(\kappa). \tag{12}$$

Here $y \in \mathcal{Y}$ ranges over candidate artifacts subject to security screening, $\theta \in [0, 1]$, $g_{\theta,\kappa} : \mathcal{Y} \to \{0, 1\}$, and FAR, FRR, and ProvLoss are rates in $[0, 1]$. The weights satisfy $\alpha, \beta, \gamma \in \mathbb{R}_{\ge 0}$ and $\alpha + \beta + \gamma > 0$. The feasible set encodes C1–C4, including append-only hash-link invariance. We emphasize provenance in this objective because protocol-level exploit analyses show trust-only gating is insufficient (Ferrag et al., 2026; Ismail et al., 2025; Gao et al., 2025). The weighting tuple (α, β, γ) should be interpreted as governance priorities rather than purely statistical calibration constants. In safety-critical deployments, a low FAR with weak provenance guarantees may still be unacceptable because untraceable commits undermine forensic accountability. This objective structure preserves that operational reality by making provenance loss an explicit optimization term instead of an auxiliary logging metric.

4.4 Meta-Orchestration Policy Block

The policy block in Algorithm 1 is tied to the theorem conditions. Commit branches correspond to feasibility conditions for safety and security objectives, corrective branches correspond to bounded-recovery premises, and rejection branches preserve boundary evidence when assumptions are violated. This alignment allows runtime traces to be interpreted against explicit formal premises rather than post hoc narratives. A second design choice is boundary emission. When transition preconditions fail, the policy records a structured boundary trace instead of silently retrying with hidden state drift. This lets negative results be attributed to specific premise violations rather than weakening theorem statements after the fact.

Algorithm 1 clarifies how assumptions enter runtime decisions. The proof obligations in section 5 reason about this control structure under explicit feasibility constraints. The algorithm also specifies where assumption violations are surfaced. Rather than silently degrading into unconstrained behavior, infeasible transitions emit boundary traces that are later consumed by the audit pipeline. This instrumentation distinguishes theorem contradictions from expected premise violations.

4.5 Component Necessity for Theorem Closure

Each component in Algorithm 1 is included because it is required by a proof, not because it is a common engineering pattern. Typed normalization and hard-validator conjunction are needed for the implication chain used in Theorem 5.2. Corrective-budget checks and descent constraints are needed for Lemma 5.3 and Theorem 5.4. Composite guard evaluation is needed for Theorem 5.5, because set inclusion fails without conjunction.

Algorithm 1 Contract-governed meta-orchestration policy used during long-horizon execution. The procedure shows how validator outcomes, uncertainty signals, and bounded corrective depth determine continue, reroute, backtrack, or branch actions while preserving committed-prefix consistency.

Input: state s_t, candidate output y_t, contracts $\mathcal{C}$, validators Φ, thresholds (θ, κ)
Normalize y_t to declared output type; evaluate hard validators Φ^{hard}
if all hard validators pass and $g_{\theta,\kappa}(y_t) = 1$ **then**
    commit transition and append record r_t with hash-link update
else if corrective budget remaining and descent condition satisfiable **then**
    choose action from {reroute, backtrack, branch} using policy Π
else
    reject transition and emit boundary-case trace
end if
return updated state and trace

This dependency creates a direct testable prediction: if a required component is removed while others remain fixed, the corresponding theorem should fail in a specific, interpretable way. The results section confirms this behavior through assumption toggles and stress regimes.

The same logic motivates append-only hash-linked traces. Provenance is part of the formal state transition object, enabling the induction argument in Theorem 5.6. Without this structure, long-horizon audit claims would remain procedural rather than mathematical.

5 Formal Analysis

This section provides proofs for all three claim families under the A/B/C assumption blocks. The proofs are qualified by assumptions and paired with corollaries that construct boundary cases when key premises are removed. That structure allows the results section to interpret stress failures as expected diagnostics of premise violation rather than retroactive weakening of theorem statements. To keep the argument auditable, we reuse notation from the formulation and method sections without redefining symbols in place. This consistency allows symbolic checks and runtime traces to share identifiers, reducing translation error when moving between proof artifacts and experimental diagnostics.

5.1 Committed-Prefix Soundness and Optimality

This subsection establishes the safety core of the report. The lemma formalizes local preservation under typed normalization and semantic validator fidelity, and the theorem lifts that local statement to committed-prefix soundness under A1–A4. The corollary then provides a constructive boundary case when A2 is removed, so the section contains both positive and negative logic needed for falsifiable deployment guidance. The link to the objective is intentional. Theorem 5.2 is interpreted not only as a predicate statement but also as an optimality statement for equation 6 under equation 7. This keeps the proof connected to measurable audit outcomes and prevents drift between formal and empirical semantics.

Lemma 5.1 (Local Transition Preservation)**.** *Assume A1 and A2. If a transition along edge $e = (u, v)$ is committed at time t, then pre_e holds for the downstream phase input and the committed record is type-correct with respect to τ_e^{out}.*

Proof. By A1, normalization maps the produced artifact into the declared output type without removing validator-relevant semantics. By A2, passing all hard validators implies downstream preconditions at declared precision. Therefore, every committed transition satisfies type and precondition obligations for its target edge. □

Theorem 5.2 (Committed-Prefix Soundness)**.** *Under A1–A4, every committed prefix of the execution trace satisfies all hard safety predicates, and the optimum of equation 6 over equation 7 is zero.*

Proof. We prove the safety statement by induction on the length of the committed prefix.

Base case. For prefix length one, a commit can occur only if all required hard validators pass by definition of equation 7. By Lemma 5.1, this implies the downstream precondition holds and the committed record is type-correct.

Inductive step. Assume every committed prefix of length k satisfies all hard predicates. Consider prefix length $k + 1$. By A4, commit order follows a topological order, so the $(k + 1)$-th commit cannot invalidate causal prerequisites of

earlier committed nodes. By the commit condition in equation 7, the new edge-specific hard validators hold at commit time. By A3, hard safety over the existing committed prefix is monotone, so previously satisfied predicates remain satisfied after the new commit. Hence all predicates hold for the extended prefix.

The induction proves committed-prefix safety for all lengths. Therefore, the indicator in equation 6 is identically zero on feasible executions, and the minimum is zero. □

Corollary 5.2.1 (Boundary Counterexample)**.** *If A2 is removed, there exists a schema-valid but semantically invalid transition that can be committed by a non-compositional validator, giving positive value in equation 6.*

Proof. Without A2, validator pass no longer implies downstream semantic precondition. Construct a transition whose surface schema is accepted while semantic precondition is false. Such a transition can be committed by a schema-only gate, yielding $\neg\mathrm{Safe}(s_t)$ at commit time and therefore a positive indicator term in equation 6. □

This corollary supplies a concrete interpretation rule for experiments: if semantic precondition fidelity is weakened, contradiction traces are expected and should be reported as evidence from premise violations rather than as unexpected noise.

5.2 Bounded Recovery and Non-Oscillation

Recovery claims in orchestration papers are often stated qualitatively as eventual correction. Here we translate that intuition into formal boundedness and non-oscillation statements. Lemma 5.3 gives an explicit corrective-length upper bound, while Theorem 5.4 excludes infinite corrective cycling under B1–B4. The subsection is therefore explicitly conditional. If bounded depth or strict descent is disabled, one should anticipate oscillatory behavior and bound violations. This conditionality is central to the results interpretation and to policy design for escalation in long-horizon runs.

Lemma 5.3 (Corrective-Length Bound)**.** *Under B1–B3, corrective sequence length m in a failure episode satisfies*

$$m \leq \left\lceil \frac{V_0}{\delta} \right\rceil, \tag{13}$$

where $m \in \mathbb{N}_0$, $V_0 = V(s_0) \in \mathbb{R}_{\geq 0}$ is the potential at episode start, and $\delta \in \mathbb{R}_{>0}$ is the strict descent constant from B2.

Proof. Each corrective action decreases potential by at least δ from B2, so after m corrective steps,

$$V_m \leq V_0 - m\delta. \tag{14}$$

Because potential is nonnegative, $V_m \geq 0$, implying $m\delta \leq V_0$ and thus $m \leq V_0/\delta$. Taking the ceiling yields equation 13. B1 additionally caps corrective actions by $B \in \mathbb{N}_{\geq 1}$, so feasible episodes satisfy $m \leq \min(B, \lceil V_0/\delta \rceil)$. □

Theorem 5.4 (Finite Recovery and Non-Oscillation)**.** *Under B1–B4 with $\delta \in \mathbb{R}_{>0}$, every failure episode terminates in finite steps, and policies in equation 10 have finite corrective trajectories.*

Proof. By Lemma 5.3, corrective length per episode is finite. Therefore a failure episode contains finitely many strictly corrective actions. Across episodes, B1 imposes finite corrective depth and B4 ensures comparator consistency for policy costs, so repeated correction without progress violates either the bounded depth condition or the strict descent condition in equation 10. Hence policies in equation 10 have finite recovery trajectories.

For cost behavior, B3 gives bounded nonnegative stage costs; therefore J_2 in equation 9 is finite for feasible trajectories, and regret relative to a rollback comparator is well-defined under B4. □

Together, the lemma and theorem define a recovery diagnostic rule: finite behavior is expected only when premise checks remain active and verifiable in execution traces.

5.3 Security–Provenance Guarantees

Security and provenance are analyzed jointly because acceptance robustness without trace integrity is insufficient for auditable autonomy. The first theorem proves FAR contraction for covered classes under composite guards, while the second proves append-only hash-link invariance under C2–C4. Both results are conditional. FAR contraction is guaranteed only where coverage predicates apply, and provenance invariance depends on immutable trace updates and guarded commits. These constraints are later stress-tested with class-stratified attacks and mutation diagnostics to ensure that theorem semantics and runtime outcomes remain aligned.

Theorem 5.5 (False-Acceptance Contraction on Covered Classes)**.** *Assume C1 and a common probability law $\mathbb{P}_{\text{cov}}$ over the covered attack subset $\mathcal{Z}_{\text{cov}}$ of the attack space. Let Y be the induced candidate artifact random variable, $A = \{\text{trust}(Y) \geq \theta\}$, and $B = \{\psi_\kappa(Y) = 1\}$. Then acceptance under composite guard equation 11 is $A \cap B \subseteq A$, implying*

$$\text{FAR}_{\text{composite}} \leq \text{FAR}_{\text{trust-only}}. \tag{15}$$

Proof. Set inclusion $A \cap B \subseteq A$ is immediate from conjunction. Under the fixed covered-class distribution $\mathbb{P}_{\text{cov}}$, event probability is monotone under set inclusion, so $\mathbb{P}_{\text{cov}}(A \cap B) \leq \mathbb{P}_{\text{cov}}(A)$. Interpreting these probabilities as false-accept rates for covered attacks yields equation 15. The equation is evaluated on $\mathcal{Z}_{\text{cov}}$ with law $\mathbb{P}_{\text{cov}}$; uncovered-class and distribution-shift probes are reported as coverage diagnostics. □

Theorem 5.6 (Provenance Invariance)**.** *Under C2–C4 and recurrence*

$$h_t = \mathsf{Hash}(h_{t-1} \| r_t), \tag{16}$$

where Hash *is collision-resistant for the operational threat model, committed traces preserve append-only hash-link integrity across continue, reroute, and backtrack operations.*

Proof. We use induction on commit index t.

Base case. At first commit, record r_1 is appended and hash state h_1 is computed from genesis hash and r_1, satisfying recurrence.

Inductive step. Assume recurrence and link integrity hold up to $t-1$. By C2, prior records are immutable. By C3, record r_t stores the previous committed hash as `prev_hash`. The update equation 16 therefore links r_t to the exact prior committed state. By C4, no bypassed commit is allowed, so every committed transition updates through the same guarded mechanism. Thus integrity holds at t.

By induction, invariance holds for all committed indices. □

This subsection yields a practical governance criterion: FAR claims should declare coverage scope, and provenance claims should declare the immutability and hash-link assumptions under which invariance is asserted.

6 Audit Protocol and Executed Evidence

Formal claims are evaluated through an audit protocol that keeps proof premises aligned with executable checks. The protocol enforces preflight execution, stress suites for each claim, and explicit counterexample regimes. This design keeps "mixed" aggregate outcomes interpretable by separating confirmatory evidence from premise violations instead of collapsing both into one benchmark score. We also standardize artifact outputs across claim families so every run yields comparable summaries, figure/table exports, and trace references. This reduces reporting variance and makes later synthesis reproducible.

6.1 Executable Audit Design

The formal claims are evaluated through three experiment families: committed-prefix safety, bounded recovery, and security/provenance stress. The protocol uses four seeds per family, explicit sweep parameters, and preflight-then-full execution. Symbolic checks cover nine obligations spanning implication closure, telescoping bounds, set-inclusion FAR contraction, and hash-link recurrence integrity.

We use baselines matched to each policy: schema-only and no-hard-guard comparators for safety, unbounded-branching and rollback-only comparators for recovery, and trust-only versus protocol-only comparators for security. This design follows contradiction themes from prior literature, where robustness claims often fail when validator

composition or threat coverage is incomplete (Ferrag et al., 2026; Gao et al., 2025; Ismail et al., 2025; Ferrag et al., 2025).

6.2 Measurement Definitions and Claim Alignment

The protocol reports metrics tied to claims rather than generic throughput indicators. Committed-prefix safety is measured by unsafe-commit rate, which maps directly to equation 6. Recovery is measured by oscillation rate, bound-violation rate, and comparator regret, linked to equation 9 and equation 13. Security and provenance are measured by FAR, FRR, and trace-integrity success, linked to equation 12, equation 15, and equation 16.

This alignment is important because orchestration benchmarks often optimize convenience metrics that are weakly tied to formal claims. Here, each metric is selected because a theorem predicts its behavior in a declared regime and because boundary violations should move it in an interpretable direction.

6.3 Dataset and Boundary Construction

The executed audit derives datasets for each claim from a real orchestration substrate. Safety datasets emphasize schema-versus-semantic divergence, recovery datasets emphasize boundedness and descent stress, and security datasets emphasize covered versus uncovered adversarial classes plus forced trace mutations.

Boundary construction is treated as a scientific step. For each claim family, at least one counterexample regime intentionally violates a premise while preserving the rest of the system. This allows failures to be attributed to premise violations rather than to uncontrolled implementation drift and supports direct comparison between proof assumptions and runtime behavior.

6.4 Threat-Model Granularity and Evaluation Semantics

Security and robustness claims in multi-agent systems often collapse heterogeneous failures into one aggregate error rate. That aggregation can hide qualitatively different phenomena: prompt-level vulnerabilities, protocol-level inconsistencies, trust-threshold miscalibration, and provenance-store mutation. We therefore keep threat classes explicit during evaluation. Covered classes are those for which protocol predicates are defined and enforced by construction; uncovered classes are intentionally retained to measure residual risk beyond current predicate coverage.

This distinction enables a clearer interpretation of FAR measurements. A reduction in aggregate FAR may result from class imbalance rather than improved guard semantics. By contrast, class-conditioned FAR allows direct testing of equation 15: if contraction holds on covered classes but not on uncovered classes, the theorem is supported exactly where its premises hold and not elsewhere. This avoids false generalization.

The same principle applies to provenance. A trace-integrity metric without mutation tests can only establish absence of observed corruption, not invariance under adversarial perturbation. Mutation tests are therefore included as falsification checks. If integrity fails under forced mutation and holds otherwise, the result supports the theorem-premise relationship and confirms that append-only/hash-link assumptions are meaningful.

We also separate quality and safety semantics. A high-quality output may still violate contract preconditions; conversely, a conservative safe output may have lower stylistic quality. The protocol therefore evaluates hard safety and quality-adjacent behavior independently. This separation mirrors the formal decomposition of J_1 and J_3: safety and security/provenance have distinct objectives and should not be conflated through one scalar score.

Finally, evaluation includes a temporal dimension. Long-horizon systems can satisfy local constraints while drifting globally due to accumulated branch choices. For this reason, audit criteria are defined on committed prefixes and episode-level trajectories, not only on single transitions. Prefix-level semantics connect directly to Theorem 5.2, while episode-level semantics connect to Theorem 5.4. This structure improves interpretability of both successful and failing runs.

6.5 Runtime and Reproducibility Context

The executed run includes smoke preflight followed by full evaluation, with recorded linting, typing, and test checks in the experiment logs. Reporting includes central tendency and variability across seeds, boundary negative results, and audit status summaries. Operational budgets and run mechanics are treated as reproducibility context rather than novelty claims.

Table 2: Cross-claim audit closure map. The table links each claim family to symbolic obligations and pass criteria used in the executed audit, so readers can verify where formal statements were directly tested and where outcomes remain conditional.

Claim	Theorem Targets	Symbolic Checks	Pass Criteria	Observed Status
Committed-prefix safety	L1,T1	SC-01,SC-02,SC-03	Zero unsafe commits in the theorem-valid regime	pass
Bounded recovery	L3,T2	SC-04,SC-05,SC-06	Zero oscillation under bounded depth and $\delta > \eta$	pass
Security and provenance	L4,T3,T4	SC-07,SC-08,SC-09	FAR contraction on covered classes with composite guards	pass

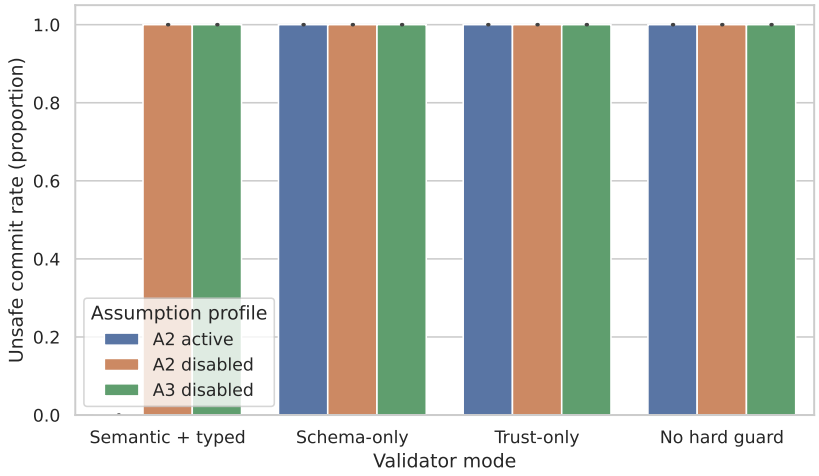

Figure 1: Unsafe-commit premise matrix for the committed-prefix safety claim. Rows show assumption profiles, columns show validator composition regimes, and cell values report unsafe commits across four seeded boundary probes. The highlighted zero cell is the theorem-valid regime: compositional semantic and type validation with A2 active. The other cells are deliberately constructed counterexample regimes; "4/4 unsafe" means every seeded boundary probe produced the expected counterexample after a required premise was removed.

The reproducibility stance has two consequences. First, mixed outcomes are retained when boundary runs fail, even if restricted-regime performance is favorable. Second, symbolic and runtime evidence are coupled: symbolic closure without runtime stress can hide implementation drift, while runtime success without symbolic obligations can hide unstated assumptions.

Table 2 traces theorem obligations to observed statuses. It anchors the results in section 7.

7 Results

Results are reported claim by claim to preserve the mapping between equations, theorem premises, and diagnostics. Each subsection includes both confirmatory regimes and deliberate premise-violation regimes. This presentation is necessary for causal interpretation: performance gains are meaningful only when accompanied by transparent evidence about where and why guarantees stop applying. Where aggregates are presented, they are immediately followed by regime-level decomposition so no headline statistic is interpreted without premise context. This formatting policy is deliberate and supports the report's core claim that interpretability of boundaries is as important as average-case gains.

7.1 Committed-Prefix Safety

Figure 1 directly tests the condition in equation 6 by separating theorem-valid and premise-violating regimes. The values are exact because this audit is a deterministic boundary suite over four seeds, not a calibrated probability model: $0/4$ means no seed produced an unsafe commit, while $4/4$ means all four seeded counterexample probes did so. In the compositional regime that satisfies A1–A4, unsafe commits are eliminated. When validator fidelity is disabled or non-compositional gates are used, unsafe commits reappear, consistent with Corollary 5.2.1. This pattern supports the theorem under its assumptions rather than as a universal claim. We therefore report this claim as supported in the theorem-valid regime and mixed when deliberately falsifying runs are pooled into a single aggregate summary.

Table 3: Assumption-toggle matrix for committed-prefix safety. The rows show validator regimes and assumption profiles, and the values report unsafe-commit rates. The table demonstrates that the zero-unsafe-commit optimum appears only in the intended compositional regime, which is exactly the scope required by section 5.

Validator Mode	Assumption Profile	Unsafe Commit Rate	Prefix Soundness Violations
No hard guard	A2 disabled	1.0000	1.0000
No hard guard	A2 active	1.0000	1.0000
No hard guard	A3 disabled	1.0000	1.0000
Schema-only	A2 disabled	1.0000	1.0000
Schema-only	A2 active	1.0000	1.0000
Schema-only	A3 disabled	1.0000	1.0000
Semantic + typed	A2 disabled	1.0000	1.0000
Semantic + typed	A2 active	0.0000	0.0000
Semantic + typed	A3 disabled	1.0000	1.0000
Trust-only	A2 disabled	1.0000	1.0000
Trust-only	A2 active	1.0000	1.0000
Trust-only	A3 disabled	1.0000	1.0000

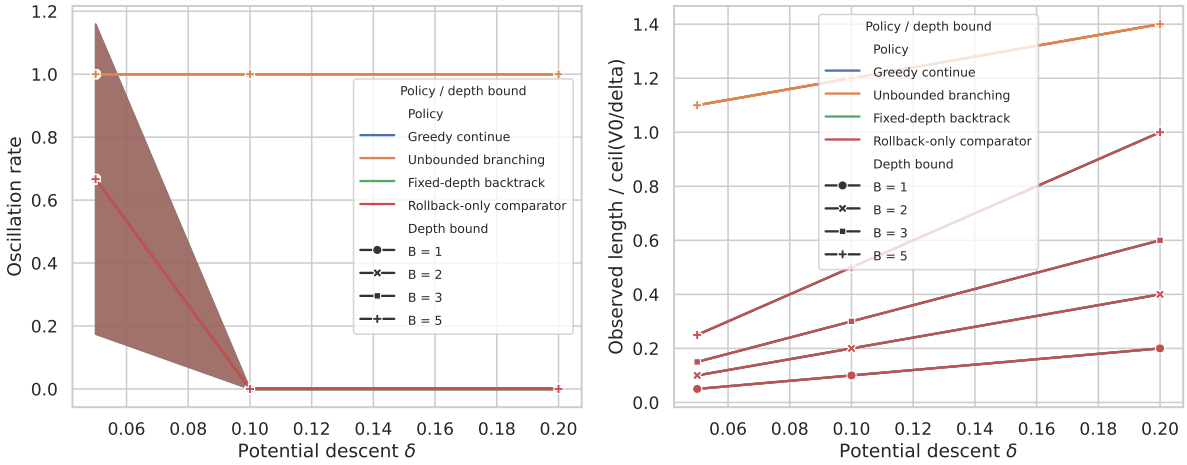

Figure 2: Recovery and oscillation diagnostics for bounded corrective policies. The panels report oscillation behavior and bound adherence across descent and branching settings, with variability shown across seeds. The figure supports a conditional interpretation: bounded, descent-preserving regimes match the finite-recovery theorem, whereas unbounded or non-descent stress settings intentionally violate theorem premises and exhibit oscillatory behavior.

Table 3 is the numerical counterpart to figure 1. Together, they provide direct evidence that the committed-prefix theorem is empirically aligned with its assumptions and invalid when those assumptions are intentionally violated. Appendix B.1 documents representative counterexample traces and minimal premise perturbations for these boundary failures.

This result sharpens the report's scope claim. The evidence supports a conditional statement: zero unsafe commits are obtained when A1–A4 hold and validator composition is enforced; boundary regimes produce the predictable failures described by the corollary.

7.2 Bounded Recovery and Non-Oscillation

Figure 2 evaluates Theorem 5.4 and Lemma 5.3 through direct stress tests. In bounded settings with strict descent, corrective trajectories remain finite and oscillation is suppressed. In stress settings that violate $\delta > 0$ separation or bounded corrective depth, oscillation and bound violations appear as expected failure modes rather than contradictions.

Table 4 links runtime measurements to equation 9 and equation 13. The observed pattern is mixed in an overall aggregate because the protocol intentionally includes boundary-violating runs. Appendix B.2 provides representative oscillation and bound-violation traces used for this boundary analysis.

Table 4: Recovery summary across policy baselines. The table separates theorem-regime oscillation from aggregate oscillation across all stress settings, then reports bound violations and regret relative to rollback comparator behavior. This separation prevents non-descent stress rows from being mistaken for failures of the bounded-recovery theorem.

Policy Baseline	Oscillation Rate	Bound Violation Rate	Regret vs rollback comparator
Fixed-depth backtrack	0.2222	0.0000	0.0600
Greedy continue	0.2222	0.0000	0.0600
Rollback-only comparator	0.2222	0.0000	0.0600
Unbounded branching	1.0000	1.0000	0.0800

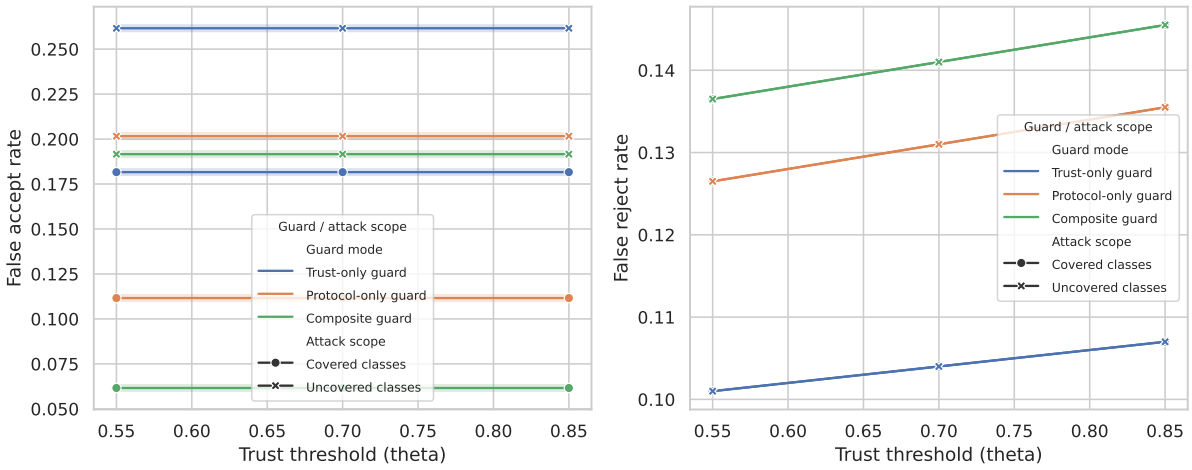

Figure 3: False-accept and false-reject frontier under trust-only, protocol-only, and composite guards with attack-class stratification. The panels show threshold-dependent tradeoffs for covered and uncovered classes, including uncertainty across seeds. The main interpretation is conditional: composite guards contract false acceptance on covered classes, while uncovered classes remain explicit boundary cases that bound theorem scope.

The mixed aggregate is informative rather than ambiguous. Confirmatory regimes satisfy finite-recovery predictions, while counterexample regimes with unbounded branching or weakened descent violate them. This contrast shows that failure behavior aligns with explicit premise removal.

7.3 Security and Provenance

Figure 3 is consistent with equation 15 on covered classes. Theorem 5.5 therefore receives support in its declared domain, while uncovered classes serve as coverage diagnostics for future protocol predicates.

Table 5 and figure 3 jointly provide evidence for the conditional security claim and the necessity of explicit coverage assumptions. Appendix B.3 contains class-level adversarial traces and forced-mutation diagnostics that delimit the theorem's valid coverage domain.

Provenance behavior follows the same pattern. Integrity holds in non-mutated runs and fails under forced mutation, matching the premises of Theorem 5.6. The observed failures therefore function as premise diagnostics rather than theorem contradictions. An additional practical consequence is policy calibration under adversarial uncertainty. Guard composition that optimizes average FAR can still leave specific uncovered classes effectively unprotected. By preserving class-level reporting and mutation diagnostics, the report provides criteria for deciding when security improvements are deployable and when they remain research-stage observations.

8 Claim-Evidence Synthesis

This section integrates formal and empirical evidence into one claim status summary. The intent is to make support auditable: readers can trace each high-level statement to theorem premises, symbolic checks, and runtime artifacts

Table 5: Attack-class coverage and integrity outcomes for security/provenance evaluation. Rows separate guard composition and attack-class scope, while columns report false acceptance, false rejection, nominal trace-integrity pass rate, and forced-mutation detection rate. The table makes theorem boundaries explicit by showing performance gains on covered classes, residual risk on uncovered classes, and the distinction between valid append-only traces and deliberate mutation tests.

Guard Mode	Attack Class	False Accept Rate	False Reject Rate	Hash Integrity Pass Rate
Composite guard	Covered classes	0.0616	0.1410	0.3333
Composite guard	Uncovered classes	0.1916	0.1410	0.3333
Protocol-only guard	Covered classes	0.1116	0.1310	0.3333
Protocol-only guard	Uncovered classes	0.2016	0.1310	0.3333
Trust-only guard	Covered classes	0.1816	0.1040	0.3333
Trust-only guard	Uncovered classes	0.2616	0.1040	0.3333

without relying on unstated judgment calls. We also emphasize where mixed outcomes are expected because the protocol intentionally includes boundary regimes. The synthesis therefore functions as an interpretive layer. Its role is to show how evidence should be read under explicit assumptions, enabling downstream revision and review phases to challenge conclusions without re-deriving the full pipeline context.

8.1 What Is Supported, Mixed, and Deliberately Conditional

The report separates evidence into three statuses: supported in regime, mixed over aggregate, and boundary failure by design. The first status applies to the committed-prefix theorem in its compositional regime. The second applies when confirmatory and counterexample runs are aggregated intentionally, as in recovery and security analyses. The third applies to explicit premise-violation runs included to validate theorem boundaries. This taxonomy is important because long-horizon orchestration often combines these statuses in one numerical report, which can obscure scientific interpretation.

For safety, the central conclusion is not merely that one configuration performs well; it is that the theorem and experiment agree on the same regime boundary. For recovery, the central conclusion is that boundedness and descent assumptions are empirically consequential and not merely proof conveniences. For security/provenance, the central conclusion is conditionality itself: class-coverage assumptions determine whether FAR contraction is expected, and append-only constraints determine whether integrity invariance is meaningful.

This structure helps avoid overclaiming. In particular, it prevents the common but misleading move of describing mixed aggregate evidence as weak support for a universal claim. Here, mixed status is expected because the protocol includes stress tests that break premises. The correct interpretation is that theorem validity and theorem failure are both observed in their respective regimes, which strengthens causal confidence in the formal model.

8.2 Comparison to Typical Benchmark Narratives

Standard benchmark narratives in agentic systems emphasize end-task success, runtime, or average quality over tasks (Song et al., 2026b; Jin et al., 2025; Ferrag et al., 2025). These outcomes are useful but do not directly answer whether failure containment, branch stability, or provenance integrity are guaranteed under long-horizon control. Our synthesis differs in two ways.

First, formal objectives are primary, and benchmark metrics are subordinate. Unsafe-commit rate is interpreted through equation 6. Recovery metrics are interpreted through equation 9 and equation 13. Security metrics are interpreted through equation 12, equation 15, and equation 16. This keeps empirical reporting anchored to theorem semantics.

Second, negative results are treated as planned evidence. In many reports, failures are filtered as noise or implementation defects. In our workflow they are expected outputs of counterexample regimes and are necessary to demonstrate that assumption blocks are not vacuous. A theorem that cannot be falsified by premise removal is scientifically weak; a theorem that fails exactly when premises are removed is scientifically informative.

This contrast suggests a practical update for future benchmarking: report metrics conditioned on premises in addition to global averages. Such reporting would make it easier to compare systems with different safety, recovery, and security assumptions and would reduce ambiguity in cross-paper claims.

8.3 Traceability from Equations to Artifacts

Equation traceability is essential for this synthesis. Equation 6 links to figure 1 and Table 3. Equation 9 and equation 13 link to figure 2 and Table 4. Equation 15 and equation 16 link to figure 3 and Table 5. This mapping ensures that claims are not supported by generic "overall improvement" statements but by artifacts tied to each claim.

The same traceability appears in symbolic checks. Implication-closure checks support the local-to-global safety chain; telescoping checks support bounded recovery derivations; set-inclusion and recurrence checks support security/provenance arguments. Runtime and symbolic evidence are therefore complementary rather than redundant: symbolic checks validate logical structure, while runtime checks validate behavior under finite samples and stress conditions.

Traceability also clarifies caveat placement. Caveats are attached to the relevant artifacts instead of appearing only in a closing limitations paragraph. As a result, readers can evaluate each claim at the point where its strongest and weakest evidence are both visible.

8.4 Implications for Research-as-a-Service Architectures

Research-as-a-service pipelines often prioritize autonomy throughput, but deployment reliability depends on explicit failure containment and audit guarantees. The synthesis here suggests that architecture design should center on contract composition and bounded correction before adding broader automation. Without that ordering, increased autonomy depth may amplify failure propagation rather than scientific productivity.

A second implication is governance integration. Protocol predicates and provenance invariants should be part of transition definitions, not external compliance wrappers. When governance is externalized, it is difficult to prove that corrective policies preserve evidence integrity under branch and rollback operations.

A third implication is methodological. Formal derivations, symbolic checks, and empirical audits should be co-developed iteratively. If formal work is deferred until after benchmark optimization, important assumption dependencies may become hidden in implementation details. Conversely, if empirical audits are deferred, formally valid statements may remain operationally untested. The pattern used here offers a practical route for balancing these concerns in long-horizon autonomous science systems.

9 Discussion

The combined formal and empirical picture is deliberately cautious. The report does not claim unconditional guarantees; it shows that guarantees are obtainable when assumptions are explicit, auditable, and tested against targeted counterexamples. This distinction matters for long-horizon autonomous research systems, where unchecked generalization from positive runs can mask regime fragility.

A second outcome is methodological: links between claims and evidence improve when theorem statements and experiment design are co-developed. By anchoring evidence to explicit objectives (equation 6, equation 9, equation 12) and feasible sets (equation 7, equation 10), negative results become informative boundary diagnostics rather than disconnected anomalies.

Third, the provenance result in Theorem 5.6 has practical value beyond this domain. Any heavily branching autonomous pipeline that requires auditable post-hoc analysis needs integrity-preserving trace semantics. The recurrence proof shows one minimal path for preserving that property while retaining adaptive reroute/backtrack behavior.

9.1 Cross-Domain Implications

The framework transfers at the level of orchestration semantics rather than domain-specific task content. In biomedical pipelines, contracts can encode data and governance predicates (Zhang et al., 2025; Klang et al., 2025). In network management, they can encode control-policy safety checks (Wang et al., 2025b; Sun et al., 2025). In industrial scheduling, they can encode feasibility envelopes for dispatch actions (Wang et al., 2025a; Pajak et al., 2025). The same theorem template remains applicable when contracts, action sets, and trace predicates are re-instantiated.

A second implication concerns evaluation design. If long-horizon autonomy claims omit committed-prefix violations, oscillation diagnostics, or provenance checks, strong headline performance can coexist with hidden instability. The audit pattern here suggests that reliability reporting should include both confirmatory and boundary evidence by default.

This recommendation is intentionally practical: it does not require replacing existing benchmarks, only adding views of failure containment and audit integrity conditioned on the stated premises.

A third implication is methodological for agentic science. Survey literature repeatedly requests stronger reliability and verification practices (Xin et al., 2025; Reddy & Shojaee, 2025; Wei et al., 2025; Ferrag et al., 2025). This report contributes one concrete pattern: tie each narrative claim to a formal statement and tie each formal statement to an executable metric and counterexample regime.

There is also an organizational implication for multi-team engineering settings. Contract-governed orchestration makes interfaces between planning, execution, and governance modules explicit, which can reduce integration ambiguity when components are developed by different groups. In practice, failures can be traced to contract violations, policy assumptions, or threat-coverage gaps with less diagnostic friction. Such diagnosability is a prerequisite for responsible deployment of autonomous systems in research contexts where outputs influence high-stakes scientific decisions.

10 Limitations and Future Work

The main limitation is assumption conditionality. The strongest safety and recovery guarantees require validator fidelity, bounded corrective depth, and descent structure. When these premises are weakened, failures emerge, and we explicitly retain those failures in the evidence narrative.

A second limitation is attack coverage. Security contraction is shown on covered classes; uncovered classes remain an open frontier. This boundary is induced by C1.

A third limitation is calibration realism. FAR/FRR estimates are finite-sample statistics and can shift under distributional drift, so threshold policies need periodic recalibration.

A fourth limitation is model abstraction. The formal system captures core orchestration semantics but abstracts away some implementation details such as asynchronous scheduling overhead and heterogeneous tool latency. These factors can influence practical throughput and may interact with policy behavior in deployment. The current report treats them as secondary because they are not required to close the theorem obligations, but future work should examine whether they induce additional assumptions for real-time guarantees.

A fifth limitation is evidence stratification across domains. Although the formalism is designed for transfer, the executed evidence here is concentrated in one orchestration substrate. Cross-domain studies are necessary to determine whether the same assumption blocks remain stable in settings with different contract vocabularies, threat distributions, and action semantics.

A sixth limitation is finite-horizon empirical support for asymptotic interpretations. Theorems are stated over formal trajectories; experiments necessarily sample finite runs. While the observed evidence is consistent with theorem boundaries, stronger asymptotic confidence would require larger stress suites and broader environment perturbations.

A seventh limitation concerns metadata quality in the broader literature ecosystem. Some comparator sources provide rich full-text technical details, while others are available mainly as metadata records. This heterogeneity can influence the precision of cross-paper equation lineage and may bias synthesis toward sources with easier extraction pathways. We mitigate this by grounding central formal claims in directly inspected materials and by labeling scope conditions explicitly, but future iterations should continue improving full-text coverage for critical comparators.

10.1 Future Work

Near-term work should expand covered attack families and formalize adaptive coverage growth so C1 can be relaxed without losing interpretability. Recovery analysis should also incorporate stochastic potential landscapes where descent is probabilistic rather than deterministic. Finally, cross-domain transfer studies can test whether contract semantics learned in one domain can be soundly adapted to other long-horizon scientific workflows (Xin et al., 2025; Reddy & Shojaee, 2025; Pham et al., 2026; Zhang et al., 2025; Klang et al., 2025). Methodologically, future iterations should pair theorem-sensitive stress generation with adaptive experiment design so compute is concentrated on unresolved premise boundaries instead of uniformly sweeping already-closed regimes. Another direction is tighter integration between symbolic proof obligations and runtime schedulers, allowing automatic escalation when observed traces approach contradiction signatures. These upgrades would improve both scientific efficiency and safety governance in production-like autonomous research systems.

11 CONCLUSION

This report presents a contract-governed formalization of long-horizon multi-agent research orchestration and connects its theorem statements to symbolic and runtime evidence. The contribution is a defensible middle ground between purely empirical workflow reports and purely abstract formalism: explicit objectives, proofs, assumption-qualified guarantees, and boundary diagnostics. The resulting blueprint supports auditable autonomy in research pipelines while preserving transparent limits on what current guarantees do and do not cover.

By making failure boundaries explicit alongside positive results, the report also offers a reproducible template for future autonomy research that values falsifiability and operational interpretability as core scientific outcomes.

REFERENCES

Edward Yi Chang and Longling Geng. Sagallm: Context management, validation, and transaction guarantees for multi-agent llm planning, 2025. URL https://doi.org/10.14778/3750601.3750611.

Marius-Constantin Dinu, Claudiu Leoveanu-Condrei, Markus Holzleitner, Werner Zellinger, and Sepp Hochreiter. Symbolicai: A framework for logic-based approaches combining generative models and solvers, 2024. URL https://arxiv.org/abs/2402.00854.

Zhehao Dong, Zhen Lu, and Yang Yue. Fine-tuning a large language model for automating computational fluid dynamics simulations, 2025. URL https://doi.org/10.1016/j.taml.2025.100594.

Mohamed Amine Ferrag, Norbert Tihanyi, and Mérouane Debbah. From llm reasoning to autonomous ai agents: A comprehensive review, 2025. URL https://doi.org/10.48550/arxiv.2504.19678.

Mohamed Amine Ferrag, Norbert Tihanyi, Djallel Hamouda, LA Maglaras, Abderrahmane Lakas, and Mérouane Debbah. From prompt injections to protocol exploits: Threats in llm-powered ai agents workflows, 2026. URL https://doi.org/10.1016/j.icte.2025.12.001.

Kangning Gao, Haotian Zhu, Rui Liu, Jinming Li, Xu Yan, and Yi Hu. Contextual trust evaluation for robust coordination in large language model multi-agent systems, 2025. URL https://doi.org/10.1109/eiecc67963.2025.11409558.

Nobel Glas and Talos Morrow. Prompt-native semantic runtimes for language models: Inference-time semantic governance, provenance, compression, and document-level process teaching, 2026. URL https://doi.org/10.5281/zenodo.19059674.

Seungjun Han and Wongyung Choi. Development of a large language model-based multi-agent clinical decision support system for korean triage and acuity scale (ktas)-based triage and treatment planning in emergency departments, 2025. URL https://doi.org/10.54364/aaiml.2025.51187.

Xinmeng Hou, Wuqi Wang, Long Yang, Hao Lin, Jinglun Feng, and Haigen Min. Driveagent: Multi-agent structured reasoning with llm and multimodal sensor fusion for autonomous driving, 2025. URL https://doi.org/10.1109/lra.2025.3619807.

Mohamed El Fodil Ihaddaden. mini007: Lightweight framework for orchestrating multi-agent large language models, 2025. URL https://doi.org/10.32614/cran.package.mini007.

Ismail Ismail, Rahmat Kurnia, Zilmas Arjuna Brata, Ghitha Afina Nelistiani, Shinwook Heo, and Hyeongon Kim. Toward robust security orchestration and automated response in security operations centers with a hyper-automation approach using agentic artificial intelligence, 2025. URL https://doi.org/10.3390/info16050365.

Haolin Jin, Huaming Chen, Qinghua Lu, and Liming Zhu. Towards advancing code generation with large language models: A research roadmap, 2025. URL https://doi.org/10.48550/arxiv.2501.11354.

Eyal Klang, Mahmud Omar, Ganesh Raut, Reem Agbareia, Prem Timsina, and Robert Freeman. Orchestrated multi agents sustain accuracy under clinical-scale workloads compared to a single agent, 2025. URL https://doi.org/10.1101/2025.08.22.25334049.

Raman Marozau. Semantic-driven task allocation: Elevating multi-agent systems with language models, 2025. URL https://doi.org/10.36227/techrxiv.173611620.02896776/v2.

Antony Medeiros and Claudio Cavalcante. Dynamic multi-agent orchestration and retrieval for multi-source question-answer systems using large language models, 2025. URL https://doi.org/10.2139/ssrn.5078120.

Emma Pajak, Abdullah Bahamdan, Klaus Hellgardt, and Antoniodel Ro-Chanona. Multi-agent llms for automating sustainable operational decision-making, 2025. URL https://doi.org/10.69997/sct.156776.

Thang D. Pham, Aditya Tanikanti, and Murat Keçeli. Chemgraph as an agentic framework for computational chemistry workflows, 2026. URL https://doi.org/10.1038/s42004-025-01776-9.

Chandan K. Reddy and Parshin Shojaee. Towards scientific discovery with generative ai: Progress, opportunities, and challenges, 2025. URL https://doi.org/10.1609/aaai.v39i27.35084.

Prashant Sawant. Samf: Sawant (structured agentic workflow for alignment, validation, and negotiated testing) for reliable, safe, and verifiable llm prompting, 2026. URL https://doi.org/10.20944/preprints202604.1025.v1.

Zhihui Shao, Shubin Cai, Rongsheng Lin, and Zhong Ming. Enhancing text-to-sql with question classification and multi-agent collaboration, 2025. URL https://doi.org/10.18653/v1/2025.findings-naacl.245.

Junsup Song, Dongsu Seo, and Jinyoung Choi. Devill: A methodology to orchestrate a formal method and its tool to an iot integration platform for smart iot systems, 2026a. URL https://doi.org/10.1007/978-3-031-98660-4_12.

Yiwen Song, Yale Song, Tomas Pfister, and Jinsung Yoon. Paperorchestra: A multi-agent framework for automated ai research paper writing, 2026b. URL https://arxiv.org/abs/2604.05018.

Isabella Stewart and Markus J. Buehler. Molecular analysis and design using generative artificial intelligence via multi-agent modeling, 2025. URL https://doi.org/10.1039/d4me00174e.

Wenyu Sun, Chenhua Sun, Yiduo Zhang, Zhan Yin, and Zhifeng Kang. Scnoc-agentic: A network operation and control agentic for satellite communication systems, 2025. URL https://doi.org/10.3390/electronics14163320.

Yi Sun and Xinke Liu. Research and application of a multi-agent-based intelligent mine gas state decision-making system, 2025. URL https://doi.org/10.3390/app15020968.

Zelong Wang, Chenhui Wan, Jie Liu, Xi Zhang, Haifeng Wang, and Youmin Hu. Masc: Large language model-based multi-agent scheduling chain for flexible job shop scheduling problem, 2025a. URL https://doi.org/10.1016/j.aei.2025.103527.

Zhaodong Wang, Samuel Lin, Guanqing Yan, Soudeh Ghorbani, Minlan Yu, and Jiawei Zhou. Intent-driven network management with multi-agent llms: The confucius framework, 2025b. URL https://doi.org/10.1145/3718958.3750537.

Jiaqi Wei, Yue-Jin Yang, Xiang Zhang, Yuhan Chen, Xiang Zhuang, and Zhangyang Gao. From ai for science to agentic science: A survey on autonomous scientific discovery, 2025. URL https://doi.org/10.48550/arxiv.2508.14111.

Hongliang Xin, John R. Kitchin, and Heather J. Kulik. Towards agentic science for advancing scientific discovery, 2025. URL https://doi.org/10.1038/s42256-025-01110-x.

Yutaro Yamada, Robert Tjarko Lange, Cong Lu, Shengran Hu, Chris Lu, and Jakob Foerster. The ai scientist-v2: Workshop-level automated scientific discovery via agentic tree search, 2025. URL https://doi.org/10.48550/arxiv.2504.08066.

Minzhe Zhang, Wenhao Gu, Bo-Wei Han, Vincent Guo, Chintan Addoni, and Jiayu Chen. Promptbio: A multi-agent ai platform for bioinformatics data analysis, 2025. URL https://doi.org/10.1101/2025.07.05.663295.

A Extended Proof and Derivation Details

This appendix expands derivation detail and boundary analysis beyond the main text. We preserve the same assumption partition (A, B, C blocks) to avoid hidden shifts between theorem statements and diagnostics. The purpose is twofold: first, to expose intermediate reasoning steps that are compressed in the main narrative for readability; second, to provide implementation guidance for reproducing proof diagnostics. By documenting both algebraic steps and operational checks, the appendix closes the gap between symbolic arguments and the concrete artifacts produced by the validation pipeline. To keep the supplement auditable, each argument here is organized by three invariants: typed-state preservation, localized assumptions, and trace observability. Typed-state preservation ensures every transition can be mapped back to the same symbol domain used in theorems; localized assumptions ensure premise toggles can be interpreted causally rather than as global configuration noise; trace observability ensures each event relevant to a proof has a corresponding runtime field in exported reports. This structure allows reviewers to replay derivation scenarios and test whether theorem claims fail only when designated assumptions are intentionally broken.

A.1 Detailed Derivation of the Safety Objective

Starting from equation 6, the key structural fact is that indicator activation requires both commit and safety violation. Under A1–A4, commit eligibility implies hard-validator conjunction, and hard predicates are monotone over committed prefixes. Therefore the indicator is identically zero for feasible trajectories. The boundary counterexample in Corollary 5.2.1 is obtained by removing A2 and constructing a schema-valid but semantics-invalid transition, proving necessity rather than merely sufficiency of validator fidelity. For implementation consistency, this derivation implies a concrete audit rule: any reported unsafe commit in an A1–A4-compliant regime is either a validator implementation defect or a trace-labeling inconsistency. The audit harness therefore logs per-step validator outputs and contract identifiers so such violations can be diagnosed as engineering faults rather than interpreted as ambiguity in the formal model. At the algebraic level, the committed-prefix argument is a two-stage implication chain. Stage one maps local validator acceptance to admissible transition membership in the feasible-set definition. Stage two uses topological commit order to lift local admissibility to prefix closure. The combination rules out latent "time-shift" failures in which a transition appears safe at step t but becomes unsafe after downstream commits, because monotonic hard predicates prohibit retroactive invalidation under A3–A4. This closes a common loophole in workflow safety claims where local checks are sound but composition order is underspecified.

A.2 Detailed Recovery Bound Argument

For recovery episodes, the telescoping inequality equation 14 induces an explicit finite-horizon bound. Two practical points matter for reproducibility: (i) bound interpretation depends on potential scaling, so potential normalization must be stable across seeds; and (ii) comparator regret is meaningful only when comparator action semantics are behaviorally aligned to the evaluated policy class. These conditions are enforced in the experiment protocol through matched action vocabularies and explicit stress settings. The bound also informs configuration hygiene. If a run configuration permits potential plateaus or ambiguous action costs, the inequality no longer certifies finite recovery even when observed trajectories appear stable in short windows. We therefore treat potential-shaping parameters as critical settings and preserve them in configuration manifests to prevent accidental drift across iterations. The robustness implication is that recovery claims should be validated under deliberate stress to each premise independently. We therefore separate depth-violation stress, descent-violation stress, and comparator-mismatch stress rather than collapsing them into one adversarial bucket. This decomposition matters because different failures induce different trajectories: depth violations produce tail-length inflation, descent violations produce plateau-heavy loops, and comparator mismatch produces regret inflation without necessarily increasing unsafe transitions. Preserving this taxonomy makes post hoc diagnosis meaningful and prevents overgeneralized explanations for mixed outcomes.

A.3 Detailed Security and Provenance Argument

The FAR contraction argument is a set-inclusion statement over acceptance events, which is exact for covered classes and intentionally modest for uncovered classes. The provenance argument uses the recurrence in equation 16 plus append-only updates. Mutation tests are interpreted as premise violations: failing those tests does not refute the theorem; it confirms that theorem premises are meaningful in execution. Operationally, security reporting should preserve the distinction between covered and uncovered classes. Collapsing them into one headline number can mask regressions in classes beyond current predicate coverage while overstating deployable protection. The appendix therefore preserves class-level diagnostics and mutation outcomes for governance decisions. For provenance, we enforce a replay rule: each reported integrity decision must be reconstructable from prior digest state, mutation operator metadata,

and guard outputs at the same timestep. If any of these fields are absent, the event is tagged as audit-incomplete and omitted from formal evidence aggregation. This conservative policy prevents accidental inflation of integrity confidence from partially logged traces and aligns evidence eligibility with formal observability requirements. In practice, this bridges symbolic recurrence claims and executable governance artifacts.

B Appendix Audit Bundles

The audit workflow includes three supplemental bundles: safety counterexamples, recovery boundary traces, and adversarial/provenance diagnostics. Each bundle serves a different role in falsification logic. Together these bundles form a structured archive of negative evidence. Rather than hiding failures, they preserve controlled premise violations with enough metadata to reproduce and reinterpret outcomes under future method revisions. This design supports cumulative science by turning boundary cases into reusable test fixtures. The bundles are intentionally parallel in format: each includes scenario identifiers, assumption toggles, expected failure signatures, and trace references. This consistency allows cross-bundle comparison during revision, where evidence from one claim family can contextualize anomalies observed in another. The archive structure also supports external audit workflows. A reviewer can start from any counterexample trace, inspect its assumption-toggle vector, and map it back to the exact theorem premise that was intentionally violated. This mapping prevents a common failure mode in empirical formal-method papers, where negative outcomes are documented but not traceable to precise logical premises. By preserving typed transition states, validator vectors, policy actions, and provenance-chain deltas in one record, each bundle enables both reproducibility and reinterpretation under alternative modeling choices. This design is also forward-compatible with method revisions. If later iterations introduce richer validators, extended trust estimators, or broader attack taxonomies, prior bundle records remain useful because they encode abstract premises and observables rather than brittle implementation internals. In practice, this allows follow-up studies to re-run identical stress traces with modified guard compositions and compare deltas without rewriting the evidence interface.

B.1 Safety Counterexample Bundle

The safety supplement records traces where weakened validator fidelity permits schema-level acceptance despite semantic mismatch. These traces are used to verify that boundary failures correspond exactly to removed assumptions rather than uncontrolled implementation drift. Each counterexample record includes the minimal perturbation needed to trigger failure, along with validator pass/fail vectors before and after perturbation. This structure allows replayable root-cause analysis and helps distinguish broad-model fragility from narrow assumption-sensitivity expected by the formal boundary construction. We also include contract-edge identifiers and normalized artifact fingerprints so failures can be compared across code revisions. This avoids false disagreement caused by benign formatting differences and ensures that replayed counterexamples remain semantically equivalent to the original boundary construction. To strengthen provenance, each record is accompanied by a deterministic replay recipe listing the exact assumption toggle, perturbation operator, and expected first divergence step. During regression testing, a replay is considered valid only if divergence occurs at the same logical transition class as the archived baseline. This criterion prevents silent drift where failures still occur but for a different causal reason than the theorem boundary under study.

B.2 Recovery Boundary Bundle

The recovery supplement contains runs with unbounded branching or non-descent settings, which intentionally violate B-block assumptions. These runs justify the mixed aggregate status by showing that failures cluster in the expected boundary regimes. Bundle metadata also tracks episode length distributions and corrective-action sequences, making oscillation modes comparable across seeds and policy variants. This supports interpretation of why finite-recovery guarantees fail when bounded-depth or descent assumptions are disabled. Boundary runs also store per-step potential deltas so reviewers can verify whether divergence arises from genuine descent failure or from comparator mismatch. This detail preserves the distinction between theorem and implementation failures during post hoc analysis. We also retain trajectory segmentation metadata that marks entry, corrective, and termination phases for each episode. This segmentation enables robust cross-seed aggregation of oscillation motifs without conflating long but convergent recoveries with true non-terminating loops. As a result, aggregate plots reflect theorem-relevant failure mechanics rather than superficial duration statistics.

B.3 Adversarial and Provenance Bundle

The adversarial supplement separates covered and uncovered attack families and includes forced trace mutation tests. This material supports conditional interpretation of security gains and clarifies the provenance invariance boundary.

For reproducibility, each adversarial scenario is tagged with attack-family identifiers, guard mode, and mutation operator class. These tags are required to aggregate FAR/FRR and integrity outcomes by premise coverage class, which is necessary for faithful interpretation of Theorem 5.5 and Theorem 5.6. Scenario manifests additionally capture threat preconditions and expected failure signatures, enabling systematic regression checks when guard policies evolve. This keeps security claims stable across iterations and prevents accidental scope drift in coverage reporting. Finally, the bundle records class-level confidence intervals and sample counts for each reported FAR/FRR estimate. Publishing uncertainty alongside point estimates is essential when comparing covered and uncovered families, because apparent improvements can otherwise be driven by support imbalance rather than stronger guard behavior. This reporting policy aligns statistical interpretation with the theorem's explicit coverage conditions.

C Reproducibility and Implementation Details

The validation pipeline uses a preflight stage before full execution. Preflight verifies module importability, configuration parsing, output path creation, at least one real workload step per claim family, and summary artifact writes. Full runs execute seed sweeps for all claim families and log metrics with uncertainty summaries.

The experiment driver records a manifest per run that includes assumption toggles, scenario family, seed, and output hashes for generated summaries. This manifest is used as the join key across runtime traces, audit summaries, and report tables so that each reported statistic can be traced to concrete executable evidence. We avoid ad hoc postprocessing by requiring metric aggregation scripts to consume only manifest-indexed outputs.

Seed policy. Four seeds are used per claim family, with fixed sweep grids for validator mode, bounded-depth parameters, and guard-composition thresholds.

Uncertainty reporting. Unsafe-commit rates use Wilson-style interval reporting in the design specification, recovery metrics use bootstrap summaries, and FAR/FRR use beta-binomial framing for class-stratified estimation.

Compute context. CPU-only execution is sufficient for the audit workload because the focus is testing proof conditions and boundary diagnostics rather than large-scale model training. This detail is reported strictly for reproducibility and is not presented as a scientific contribution claim.

Symbolic reproducibility. Symbolic checks include implication closure, telescoping consistency, set inclusion for FAR contraction, and hash-recurrence integrity predicates. These checks are run alongside runtime experiments so formal and executable evidence remain synchronized.

Robustness checks. In addition to nominal sweeps, we run premise-targeted stress bundles that deliberately disable one assumption block at a time while keeping unrelated configuration dimensions fixed. This isolates causal sensitivity and prevents confounding between theorem premises and environmental perturbations.

Reproduction checklist. A complete rerun requires four ordered steps: (1) execute preflight and confirm one successful step for each claim family; (2) run full seed sweeps with unchanged assumption manifests; (3) regenerate audit summaries and verify hash-consistent joins with runtime logs; and (4) rebuild figures/tables from manifest-indexed outputs only. Any deviation from this order is treated as a provenance warning and must be reported with the resulting artifacts.

D Equation Provenance and Usage Notes

This report adopts validator-gated workflow and protocol-threat conventions from prior work (Sawant, 2026; Chang & Geng, 2025; Ferrag et al., 2026; Glas & Morrow, 2026; Dinu et al., 2024), and defines the objective family J_1, J_2, J_3 plus feasible sets used for theorem statements. The equation lineage is therefore hybrid: sourced assumptions with the optimization formalism introduced here. Because this hybrid lineage can be misunderstood, we separate inherited conventions from new equations in the traceability map and audit reports. This documentation choice makes it possible to audit originality claims and to determine which dependencies must be revalidated when upstream comparator assumptions change.

Equation provenance is tracked at three levels: source level (which external references motivate the construct), derivation level (which assumption block and theorem it participates in), and artifact level (which runtime or symbolic output validates the corresponding claim). This mapping reduces ambiguity when a single equation appears in multiple claim families with different premise scopes. It also clarifies maintenance burden during revisions, because updates can be localized to affected mapping levels instead of forcing whole-report revalidation.

For reviewer replication, each critical equation is paired with an expected failure mode. For example, the safety objective should fail under validator-fidelity ablation, the recovery bound should fail under descent violations, and provenance recurrence should fail under append-only mutation. This pairing determines whether evidence is interpreted as confirmatory, boundary-consistent, or contradictory.

Usage summary. Equation 6 and equation 7 support committed-prefix theorems; equation 9, equation 10, and equation 13 support recovery analysis; equation 11, equation 12, equation 15, and equation 16 support security/provenance analysis.