RIGHTSOURCE: JOINT PLANNING OF EVIDENCE AND AUTHORIZATION FOR TOOL-USING AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Tool-using agents must determine not only whether an action satisfies its task requirements, but also whether the appropriate party has authorized it. These decisions become difficult when relevant information is incomplete, evidence comes from different sources, and confirmations apply only to particular actions. A successful factual check does not establish permission, while requesting every possible confirmation can introduce unnecessary interaction. We introduce **RightSource**, a framework for synthesizing verifiable decision policies that distinguish *factual evidence* from *action-scoped authorization*. RightSource represents uncertainty as a set of possible worlds and records confirmations separately according to their source and scope. It plans which observations and confirmations to acquire, allowing subsequent queries to depend on earlier outcomes. Before permitting an action, the framework checks that its safety conditions hold in every remaining possible world and that all required authorizations have been obtained. Within the specified finite model and registered sources, the resulting policies provide checkable guarantees for permitted actions. We evaluate RightSource on retail workflows adapted from τ-bench and prompt-injection scenarios from AgentDojo. The results show fewer retail queries with similar observed task-completion rates, and fewer argument-scope violations in the tested injection scenarios, supporting joint planning of evidence acquisition and authorization.

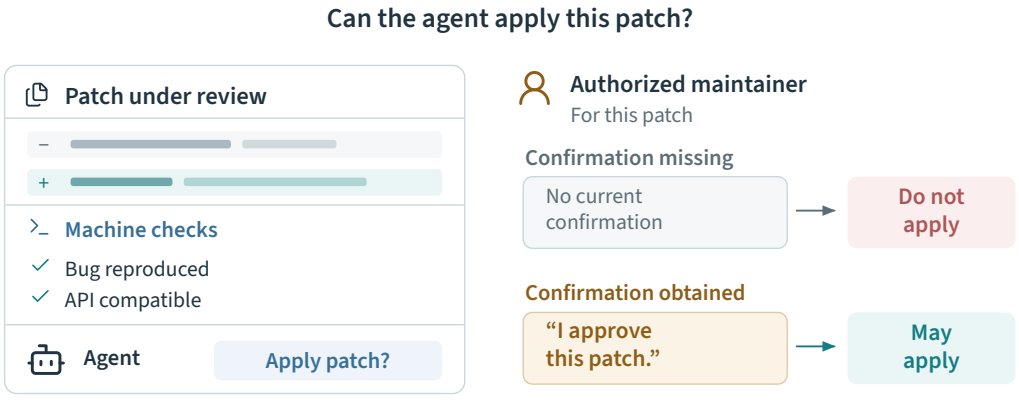

Figure 1: Illustrative repository repair. Both situations share the same patch and machine-check results. Under the example's requirements, the patch remains ineligible until the authorized maintainer's current confirmation is acquired. The two cases contrast action eligibility; they do not prescribe a query order.

1 INTRODUCTION

A repair agent may reproduce a bug and verify that a patch preserves API compatibility, yet lack the repository owner's current approval. Asking for approval is wasteful if a cheap test rules the patch

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

out. The agent must choose which source to consult while distinguishing factual evidence from the authority to confirm an action.

Runtime monitors, action authorizers, and retrieval-stopping methods protect adjacent boundaries (Wang et al., 2026; Zhang et al., 2026; Delattre et al., 2026; Roh & Han, 2026). RIGHTSOURCE selects machine and human sources by cost while keeping factual evidence separate from authorized confirmation. A policy file may identify *whom to ask*, but is not that person's current answer. Actions require safety and authorized confirmation throughout the remaining worlds. Guarantees are relative to the declared registry, source contracts, and trusted initial evidence.

A deterministic compiler selects source queries and certified actions after signed activation checks the registry and observation-map bindings. An optional symbolic path checks implicit-support abstractions; its proposer orders refinement coordinates, not runtime queries or actions.

Matched controls separate predictive information from qualifying evidence: context changes which proof is worth acquiring without making an action eligible. Exhaustive fixed-order comparisons identify localized reordering benefits; a simple context-first conditional index policy is optimal throughout the tested context-access conditions. Retail compares planners under a shared gate; separate system comparisons cover retail costs and prompt-injection outcomes.

2 THE TWO CONDITIONS FOR ACTING

Running example. Repository-repair worlds differ in bug reproduction, compatibility, and the owner's response. Each world fixes that response; K records acquired confirmations. Tests remove worlds but cannot confirm approval. The owner's confirmation may leave B unchanged if its outcome is already determined. A failed compatibility test permits rejection without asking the owner; a safe patch still requires confirmed approval.

Compatibility evidence certifies compatibility, not owner approval. The full source-effect comparison is in Appendix A.

State. The registry's *support* is its set of possible worlds. World w declares safe actions $A^\star(w)$ and action prerequisites $R(w,a)$. State (B,K,U) records remaining worlds, prerequisites confirmed by authorized sources, and unused queries ($U_0 = \mathcal{Q}$). Initial confirmations K_0 are trusted as authorized; otherwise $K_0 = \varnothing$. Appendix A collects the notation.

What a query can do. An outcome may shrink B or add confirmations $K_q(o)$ within source authority $\mathrm{Auth}(q)$. Thus, truthful observations need not grant approval, and approval need not reduce uncertainty. Query q with public outcome o updates the state as

$$\begin{aligned} B_{q,o} &:= \{w \in B : g_q(w) = o\}, \\ (B,K,U) &\mapsto \big(B_{q,o}, K \cup K_q(o), U \setminus \{q\}\big), \\ K_q(o) &\subseteq \mathrm{Auth}(q). \end{aligned} \tag{1}$$

The belief update is descriptive; the subset condition enforces the registered source scope.

Stopping rule. For a remaining cell B, let $\mathrm{Safe}(B)$ be the common safe actions, and let $\mathrm{Req}_a(B)$ collect the prerequisites of action a across that cell. The certifiable actions are

$$\begin{aligned} \mathrm{Safe}(B) &:= \bigcap_{w\in B} A^\star(w), & \mathrm{Req}_a(B) &:= \bigcup_{w\in B} R(w,a), \\ \mathrm{CAct}(B,K) &:= \{a \in \mathrm{Safe}(B) : \mathrm{Req}_a(B) \subseteq K\}, & \mathrm{Cert}(B,K) &\iff \mathrm{CAct}(B,K) \neq \varnothing. \end{aligned} \tag{2}$$

The compiler identifies an action safe in every remaining world and checks its required confirmations against K. An authorized repair and a confirmation-free defer action can coexist without charging the repair's approval to the defer action. Source scopes and present commitments are formalized in Appendix A.

Proposition (Strictly stronger than ordinary decision-region stopping)**.** *Let* $\mathrm{DR}(B)$ *denote the ordinary stopping rule* $\mathrm{Safe}(B) \neq \varnothing$*. Then*

$$\mathrm{Cert}(B,K) \iff \bigvee_{a\in\mathrm{Safe}(B)} [\mathrm{Req}_a(B) \subseteq K], \qquad \mathrm{Cert}(B,K) \Longrightarrow \mathrm{DR}(B). \tag{3}$$

The converse fails when $\mathrm{Safe}(B) \neq \varnothing$ *but no action in* $\mathrm{Safe}(B)$ *has all its required confirmations. For an ordinary decision-region policy with terminal action* a_ℓ *at leaf* ℓ*, its unauthorized-action mass is*

$$M_{\mathrm{unauth}}(\pi) = \sum_{\ell \in \mathcal{L}} P(\ell)\, \mathbf{1}[a_\ell \notin \mathrm{CAct}(B_\ell, K_\ell)]. \tag{4}$$

A two-world instance with one shared action and one unconfirmed action prerequisite has $\mathrm{DR}(B) = 1$ *but* $\mathrm{Cert}(B, K) = 0$ *and* $M_{\mathrm{unauth}} = 1$*. Proposition 4 in Appendix A.2 gives the proof.*

Certification cannot depend on B alone: in a one-world instance, a constant-outcome authority query leaves B unchanged but confirms the sole action prerequisite. State (B, K, U) gives an ordinary finite stochastic-shortest-path formulation, with augmented action region $\widehat{\mathcal{D}}_a = \{(w, K) : a \in A^\star(w), R(w, a) \subseteq K\}$. DRD or HEC augmented with confirmations can simulate this formulation; the world-only comparators do not. In a fixed registry, reachable (B, U) determines K. Explicit K supports accounting, not minimal state representation (Appendix A).

Admissible queries and objective. A query is admissible if it can either split the current world cell or confirm an unresolved prerequisite of an action that may become safe after a later split. Define $\mathrm{RelReq}(B) := \bigcup_{w \in B} \bigcup_{a \in A^\star(w)} R(w, a)$. Then

$$\begin{aligned}\mathrm{Adm}(B, K, U) := \{q \in U : \exists w, w' \in B,\ & g_q(w) \neq g_q(w') \\ & \vee\ \exists o \text{ reachable},\ K_q(o) \cap (\mathrm{RelReq}(B) \setminus K) \neq \varnothing\}.\end{aligned} \tag{5}$$

The conservative union in RelReq preserves queries needed by actions that are not yet common to B. Admissibility does not imply positive decision value after query cost. For an admitted query, let its exact continuation cost be

$$\mathcal{C}_q(B, K, U) := c(q) + \sum_{o : B_{q,o} \neq \varnothing} p(o \mid B, q) V(B_{q,o}, K \cup K_q(o), U \setminus \{q\})\,. \tag{6}$$

Outcomes with $B_{q,o} = \varnothing$ are unreachable and omitted. If any reachable child has value $+\infty$, then $\mathcal{C}_q = +\infty$ and the compiler discards q. The deterministic compiler then solves the finite dynamic program

$$V(B, K, U) = \begin{cases} 0, & \mathrm{CAct}(B, K) \neq \varnothing, \\ \min\limits_{q \in \mathrm{Adm}(B,K,U)} \mathcal{C}_q(B, K, U), & \substack{\mathrm{CAct}(B,K)=\varnothing, \\ \mathrm{Adm}(B,K,U)\neq\varnothing,} \\ +\infty, & \text{otherwise.} \end{cases} \tag{7}$$

Here $p(o \mid B, q)$ is the exact branch probability induced by the registered prior. Deterministic total maps make repeated queries uninformative and add no confirmation. With nonnegative costs, deleting repetitions loses no value, so the non-repeating policy restriction preserves the optimum.

Main Guarantee. *Assume finite registered worlds and queries, strictly positive priors, deterministic total outcome maps, nonnegative query costs, valid authorized initial confirmations* K_0*, and contracts satisfying* $K_q(o) \subseteq \mathrm{Auth}(q)$*. Every action exposed by the compiler is safe in every remaining world and has authorized confirmation for that action's prerequisites in every remaining world. For the fixed full registered query set* $\mathcal{Q}$ *and initial state, Equation 7 minimizes expected query cost over grounded, non-repeating policies that act only at certified states. A proposer cannot bypass certification; preserving the full-registry optimum after query retention requires the separate equivalence checks in Appendix D.6.*

The base objective treats every certified action equally. An application-specific loss $L(w, a) \in \mathbb{Q}_{\geq 0}$, fixed before compilation, trades query cost against expected terminal loss. With an empty minimum defined as $+\infty$, let

$$\begin{aligned} T_L(B, K) &:= \min_{a \in \mathrm{CAct}(B,K)} \sum_{w \in B} p(w \mid B) L(w, a), \\ \mathcal{C}^L_q(B, K, U) &:= c(q) + \sum_{o : B_{q,o} \neq \varnothing} p(o \mid B, q) \\ &\qquad \cdot V_L(B_{q,o}, K \cup K_q(o), U \setminus \{q\}), \\ V_L(B, K, U) &:= \min\left\{ T_L(B, K), \min_{q \in \mathrm{Adm}(B,K,U)} \mathcal{C}^L_q(B, K, U) \right\}. \end{aligned} \tag{8}$$

The compiler either stops with the best certified action or queries for a lower-loss one. Selection remains inside CAct; the deletion argument preserves optimality for query cost plus loss. SourceGround-8 and the constructed-loss study use this objective; Algorithm A2 is the separate $L \equiv 0$ reference routine. Appendix A gives the proofs.

3 Verified Search and the Trusted Boundary

3.1 Checked Synthesis and Execution

The complete finite registry is compiled into a policy over admissible source queries (Figure 2).

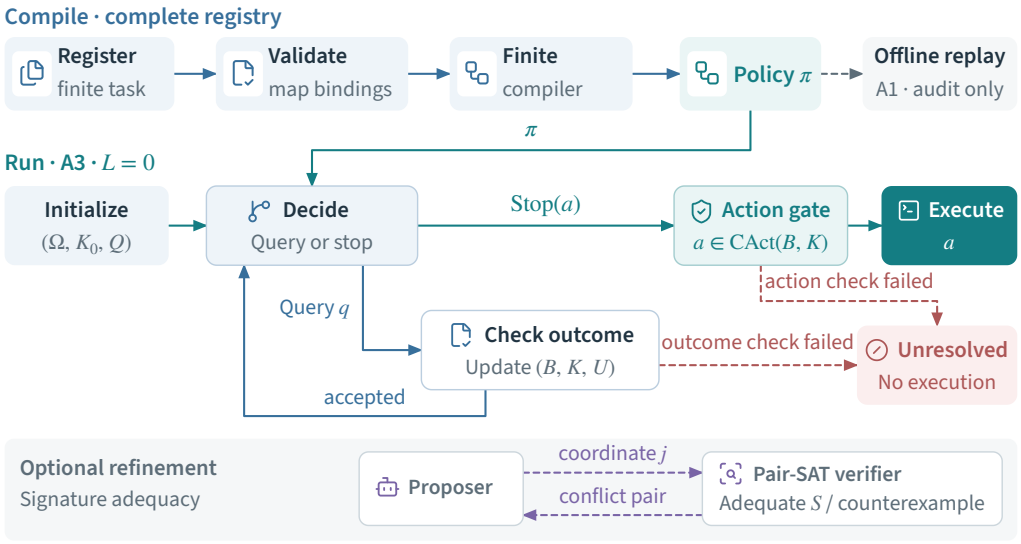

Figure 2: Finite compilation, offline audit, and runtime action exposure have distinct paths. Activation binds the registry and maps; runtime checks human receipts against authorized signers and the current execution, while raw machine authenticity remains an external premise. The optional Pair-SAT path checks signature adequacy, not cost-optimality equivalence. Only a certified Stop can expose an action.

Trusted boundary and semantic assumptions. The executable trusted computing base includes schema and registry binding, extractors and outcome maps, authority-scope checks, signature and quotient encoders, the SAT solver, leaf evaluation, the compiler, and public projection. Soundness requires complete worlds, correct source scopes, action labels and outcome semantics, and valid initial confirmations. Input binding authenticates the checked artifacts, not their semantics.

3.2 Grounding and Verification

The grounder rejects unregistered outcomes, malformed support, and out-of-scope confirmations, but can miss consistent label permutations. The *signed activation gate* checks the registered contract and observation-map conformance before compilation. Human confirmations must match the authorized source, query, and current execution; failed checks expose no action. Raw-observation authenticity and unseen-value semantics remain external assumptions.

3.3 Decision Objectives and Execution Scope

The checked finite replay and action-exposure interfaces (Algorithms A1–A3) use the zero-loss routine. If Defer is always certifiable, this objective permits immediate stopping. The loss-aware finite studies instead compare its registered terminal loss with query continuations under Equation 8;

certification alone does not force a stop. These finite interfaces do not specify a complete tool-agent loop.

In the agent evaluation, the controller checks registered action categories before dispatch, not the semantic scope of open-ended arguments. Certifying an email-sending category, for example, does not certify a generated recipient or message body. The parameter-scope violation metric therefore concerns an execution boundary outside the finite guarantee. A separate Retail ablation holds the model, registry, argument checks, and authorization gate fixed while varying Dev-fixed, L2, and EX (Section 5.5). The original system comparisons, including AgentDojo, do not isolate query-planning benefits from gating effects.

3.4 Using the admissible set

Equation 5 admits either descriptive or authority progress. A query making neither can be deleted without changing reachable certification, removing its nonnegative cost. Appendix A proves this deletion rule and optimality for the unconstrained finite objective.

3.5 Optional implicit-support verification

Pair-SAT checks action–requirement signatures over implicit support; an untrusted proposer orders refinement coordinates, not runtime queries. Its encoding and singleton-action restriction are given in Appendix A, Equation 22; the finite compiler implements general action-indexed prerequisites. Signature adequacy does not preserve acquisition cost after query removal, so finite replay checks policy equivalence separately. Large-support experiments measure symbolic verification and materialization, not full-reference replay over every latent world. The implementation does not enforce deployment caps or establish budget-constrained optimality; adversarial resource exhaustion is not evaluated.

4 Evaluation Design

Within each study, framework-internal comparisons share task inputs, costs, and verification code. Finite replay and implicit-support scaling use different execution paths. A method that reaches the construction cap is recorded as censored. Four executable finite studies test evidence-dependent eligibility (AuthorityWitness-1), information-guided proof acquisition (InformationRouting-1), query ordering (SourceGround-8), and loss-aware stopping (constructed loss). End-to-end studies include a paired, shared-gate Retail comparison and separate system-level comparisons; Appendix E specifies their statistical and accounting limits. Complete finite-study inputs, grids, and the validation and record inventory appear in Appendix G.

SourceGround-8 uses pinned public code to define factual configurations, with scripted owner responses and authority contracts. It is a finite planning benchmark rather than execution of complete repositories. Retained scaling, provider, and external-system records are distinguished from the executable studies in Appendix G.

Common-authority planning baselines. Machine-first (MF) follows a fixed query priority and optimizes stopping. MF+skip additionally permits permanent skips. Best-fixed+skip (BF) optimizes stopping and skipping within every complete query permutation, then chooses a permutation at the initial prior. Its relative order is fixed before observation; skipping and stopping remain observation-dependent. RightSource's exact planning policy (EX) can also choose the next query without this order restriction. Receding two-step lookahead (L2) replans after each observation. All five methods share the same observation model, confirmation requirements, costs, and terminal losses. Appendix C.1 gives the cursor recursion and the independent checks of these comparisons.

5 RESULTS

5.1 VERIFIED EVIDENCE AT FIXED FACTUAL SUPPORT

AuthorityWitness-1 connects confirmation acquisition to the independent in-toto verifier (Torres-Arias et al., 2019). A single-world instance fixes the artifact and the verification policy. The descriptive payload is already known, but `AcceptCurrentGate` requires successful verification of the signed evidence; Defer is always available.

The unmodified in-toto 3.1.0 verifier accepts valid evidence and its relay and rejects 11 negative controls. Four accepted boundary probes delimit its guarantee (Appendix C.2). During planning replay, each proof is checked under the original fixed context before K grows. All 12 querying rows leave $B = \{w\}$ unchanged but add `AcceptCurrentGate` to CAct; Defer remains available. MF, L2, and EX agree on all 9 conditions and 27 traces, attaining $\min(c, L_{\text{defer}})$. This one-protocol study tests the eligibility mechanism, not planning performance, current human approval, or release safety.

5.2 INFORMATION FOR FINDING THE REQUIRED EVIDENCE

InformationRouting-1 combines source selection with native proof acquisition. Two candidate artifacts each have two proof routes, costing 1 and 2. Admitting the preferred artifact, admitting the fallback, and deferring have losses 0, 1, and 8. A query costing 0.1 reads a pre-existing descriptor that predicts route availability but supplies no confirmation. Its correlation with availability is specified by the model; reading it changes neither the available files nor their validity.

Table 1 compares four structures. Holding the entire route distribution fixed, descriptor access reduces EX's primary objective from 2.715 to 2.514, a net information value of $257/1280 \approx 0.201$, including its acquisition cost. Descriptor A makes the preferred artifact's cheap proof route uniquely optimal next; descriptor B selects the fallback's cheap route. Both outcomes leave K empty. Admission becomes eligible only after the corresponding proof is acquired and natively verified.

Table 1: InformationRouting-1 primary objectives. C0, C1, and N share the same joint proof availability; N makes the descriptor independent. Index uses the reservation rule of Weitzman (1979): directly in I and, after acquiring the descriptor, conditionally in C1 (Appendix C.3). Lower is better.

Structure	BF	EX	Index
I: independent routes	3.062500	3.062500	3.062500
C0: correlated routes, no descriptor	2.714844	2.714844	–
C1: predictive descriptor available	2.537500	2.514063	2.514063
N: independent descriptor available	2.714844	2.714844	–

BF exceeds EX by $3/128 \approx 0.023$ in C1. For the selected optimal fixed order, the entire gap arises on a history of probability $1/32$: both preferred proofs are missing, but the skipped fallback routes cannot be revisited. The descriptor-first conditional-index control matches EX in all five descriptor settings. In N, EX and BF skip the uninformative descriptor and recover C0's objective. Complete costs, strict decision branches, and the sensitivity grid appear in Appendix C.3.

5.3 SOURCEGROUND-8: ACQUISITION AND DEFERRAL

SourceGround-8 contains eight bounded software tasks, including target-list selection, formatting options, and redirect-header policy. All methods receive the same worlds, observations, authority scopes, and action-specific requirements. They optimize Equation 8 with uniform priors, machine cost 1, owner-query cost 4, and terminal loss 8 for Defer or 0 for a certified concrete resolution. Rejections and owner cancellations are resolutions, not successful modifications.

RightSource (EX) reduces mean query cost from 3.128 for Machine-first to 2.989, a 4.44% reduction across the eight primary tasks (Table 2). All methods have zero terminal loss and the same mean number of human queries, so the saving comes from machine queries. BF obtains 3.003, leaving a 0.46% reduction relative to this stronger control; L2 matches EX. Percentages use unrounded

equal-case means. These are expected query costs under the declared weights, not measured API expenditure. The MF–EX objective gap decomposes as

$$\frac{5}{36} = \underbrace{0}_{\text{skipping}} + \underbrace{\frac{1}{8}}_{\text{fixed-order selection}} + \underbrace{\frac{1}{72}}_{\text{order restriction}} . \tag{9}$$

Selecting a better fixed relative order explains 90% of that gap. BF matches EX on seven primary tasks; GTasks retains a difference of $1/9$. Across the unchanged nine-setting grid, BF is worse in six of 72 conditions from GTasks and Rich, while L2 matches EX throughout. Appendix C.4 summarizes the primary and sensitivity results, including the two settings where MF benefits from skipping; complete condition-level results and ties are supplied in the artifact.

In GTasks, the agent resolves a target-list choice. After inspecting selection parameters, EX may ask the owner before listing targets, or list targets before asking. The selected BF policy lists before asking. Relative to BF, EX saves one unit-cost list query in three of the 27 equally likely configurations, where the owner cancels: $3/27 = 1/9$. Enumeration of all fixed orders establishes the strict gap. This localized benefit does not establish a general need for full-horizon search.

Table 2: SourceGround-8 primary setting. Objective includes query cost and terminal loss; all methods share the same certification and stopping contracts.

Method	Objective	Query cost	Terminal loss	Human queries	Total queries
MF	3.128	3.128	0.000	0.412	1.891
MF+skip	3.128	3.128	0.000	0.412	1.891
BF	3.003	3.003	0.000	0.412	1.766
L2	2.989	2.989	0.000	0.412	1.752
EX	2.989	2.989	0.000	0.412	1.752

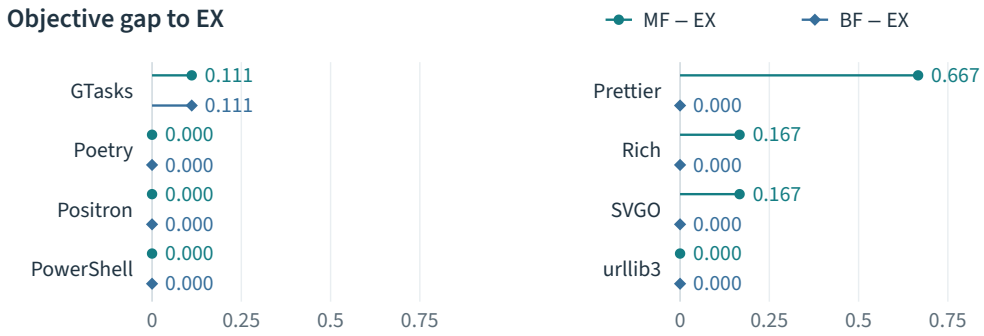

Figure 3: SourceGround-8 primary setting. Each task has two objective gaps: MF minus EX and best-fixed+skip minus EX. All eight tasks, including zero gaps, are shown. Best-fixed+skip chooses one relative query order before observation but can skip and stop adaptively.

5.4 When to test, ask, or defer

With repair, reject, and defer available, a separate constructed study evaluates Equation 8 under identical authority semantics for all methods. BF matches EX in all 504 primary and all 1,008 cost-sensitivity conditions. Their primary mean objective is 0.628080, compared with 0.631340 for L2 and 0.628698 for MF and MF+skip. L2 is worse than EX in 30 of the complete 1,512 conditions, but BF reaches the optimum in each. The gaps show limits of L2 without establishing a need for dynamic query reordering. BF's equality with EX does not imply that model-free ordering or wholly nonadaptive decisions suffice. BF and EX ask people more often than MF on average in the primary study while achieving lower expected query cost plus terminal loss (Appendix C.5).

5.5 END-TO-END AGENT EVALUATION

We evaluate retail workflows adapted from τ-bench (Yao et al., 2024) and AgentDojo injection scenarios (Debenedetti et al., 2024). The Retail ablation holds the model, registry, argument checks, and authorization gate fixed. Acquisition interactions I count read-only tool observations, user confirmations, and factual follow-up questions; they differ from total tool calls. Each method has 345 trajectories over 115 tasks and three seeds.

Table 3: Retail planning with a shared gate. Success and I are means $\pm$ SEM across 115 task-level averages. Times are trajectory means in seconds; planning time is included in latency. Appendix E.2 gives counts, paired inference, deferrals, violations, and cost accounting.

Planner	Success ↑	Interactions I ↓	Planning (s) ↓	Latency (s) ↓
Dev-fixed	0.716 ± 0.024	4.214 ± 0.12	0.12	18.4
L2	0.719 ± 0.024	4.148 ± 0.11	1.85	19.8
EX	0.722 ± 0.024	4.133 ± 0.11	3.40	20.9

Planning benefit and overhead. EX uses 28 fewer acquisition interactions than Dev-fixed (1.93%; paired mean difference -0.0812, 95% CI $[-0.150, -0.012]$). Its success difference is $+0.58$ percentage points; the one-sided 95% lower bound of -1.78 points exceeds the prespecified -3-point non-inferiority margin. The EX–L2 interaction difference is unresolved (-0.0145, 95% CI $[-0.061, 0.032]$). Reference-priced cold cost is 1.41% lower than Dev-fixed's, but mean latency is 13.6% higher. Table 22 reports the separate Retail system comparison, including assumed policy reuse.

Table 4: AgentDojo results across Workspace (40 tasks), Banking (16), Slack (21), and Travel (20), with an experiment-defined fixed injection pairing per task and three repeats. Each method has 291 trajectories per clean/attacked condition. Values are means $\pm$ task-level SEM. ASR and $M^{\text{arg}}_{\text{unauth}}$ use percentages (SEM in percentage points); utilities are task-success proportions.

Method	ASR (%) ↓	Clean utility ↑	Attacked utility ↑	$M^{\text{arg}}_{\text{unauth}}$ (%) ↓
Unprotected Agent	54.3 ± 2.9	0.725 ± 0.026	0.261 ± 0.026	56.7 ± 2.9
Prompt Guard / Filter	24.4 ± 2.5	0.708 ± 0.026	0.412 ± 0.029	26.1 ± 2.6
Tool Filter (rule-based)	16.5 ± 2.2	0.674 ± 0.027	0.485 ± 0.029	18.2 ± 2.3
RightSource	8.2 ± 1.6	0.704 ± 0.026	0.564 ± 0.029	3.8 ± 1.1

Attack outcomes. RightSource records 24 attack successes in 291 trajectories, compared with 48 for the rule-based tool filter: a reduction of 8.2 percentage points, computed before rounding. Clean and attacked utility are higher by 3.1 and 7.9 points, respectively, although clean utility is 2.1 points below the unprotected agent. Trajectories containing an executed call with arguments outside the user's authorization scope fall from 53 to 11 (18.2% to 3.8%). The controller checks registered category-level credentials before dispatch, but does not certify open-ended arguments (Appendix E).

5.6 ADDITIONAL VALIDATION

Finite replay checks policy invariants; signed-activation tests reject permuted or incomplete observation bindings (Appendix D.3.1). Recorded scaling studies reduce materialization without runtime or state-count dominance; model-guided ordering reduces one verifier-work counter but increases end-to-end time. The original campaigns are not rerun by the present artifact; Appendix G distinguishes executable studies from retained records.

6 RELATED WORK

Enforcement and authorization. For tool agents (Yao et al., 2023), AgentSpec enforces runtime policy (Wang et al., 2026); FAVA tracks evidence-backed permissions and temporal obligations

(Zhang et al., 2026); CAGE certifies a fixed action under typed-return uncertainty (Delattre et al., 2026). RightSource optimizes source acquisition; Appendix B.1 compares the guarantees.

Information acquisition. HALT stops retrieval when learned verification judges expected claims covered (Roh & Han, 2026). EC^2 identifies one hypothesis equivalence class; hyperedge cutting (HEC) permits overlapping decision regions (Golovin et al., 2010; Javdani et al., 2014). Adaptive submodularity supports greedy acquisition (Golovin & Krause, 2011). POMDP and experimental-design views model belief-space problems (Kaelbling et al., 1998; Chaloner & Verdinelli, 1995). General belief-state planning subsumes our finite (B, K, U) formulation; DRD/HEC can represent it with confirmation-augmented regions. World-only stopping cannot distinguish equal beliefs with different confirmations.

Verification and deferral. LLM-Modulo and neuro-symbolic synthesis separate model proposals from verification (Kambhampati et al., 2024; Jha et al., 2023), as does our optional refinement path. Counterexample-guided abstraction refinement (CEGAR) and proof-carrying execution provide related checking patterns (Clarke et al., 2000; Necula, 1997). ToolEmu, AgentDojo, and AgentHarm test agent hazards (Ruan et al., 2024; Debenedetti et al., 2024; Andriushchenko et al., 2024); selective prediction and learning-to-defer study abstention or expert referral (Geifman & El-Yaniv, 2017; Madras et al., 2018).

7 Discussion and Conclusion

RightSource jointly plans factual evidence and scoped confirmations. Acquired evidence can change eligibility without changing factual support. Simple controls attain the optimum: conditional indexing in its InformationRouting-1 settings, L2 throughout SourceGround-8, and best-fixed+skip throughout constructed loss. Fixed-order selection explains most of SourceGround-8's machine-first deficit; residual reordering gains are localized. With a shared Retail gate, EX reduces acquisition interactions by 1.93% relative to Dev-fixed, but its advantage over L2 is unresolved and latency increases. AgentDojo attack success is lower than for the tested filters; argument-scope violations remain.

Limitations. Guarantees require a complete finite registry, correct deterministic source contracts, and valid initial confirmations; key provisioning and observation authenticity are external. SourceGround-8 uses selected configurations, scripted authority, and uniform priors; InformationRouting-1 specifies descriptor–availability dependence rather than measuring deployment. Finite-study costs and losses are normative; human responses, annotation cost, and open-world behavior are unmeasured. Paired inference is limited to the shared-gate Retail study; AgentDojo remains a system-level comparison. Cost estimates depend on reference prices (Appendix E). Pair-SAT scaling covers singleton actions with $R(w, a) = R_0(w)$; the finite compiler supports general $R(w, a)$. Noisy observations and broader augmented-space HEC comparisons remain future work.

GENERATIVE AI DISCLOSURE

Generative AI tools were used to polish the writing, retrieve and discover related work, support research ideation and execution, and draft sections of the paper. Writing assistance included sentence-level revision and section drafts; retrieval assistance identified candidate references. Research assistance included discussing study designs, benchmark and source annotation, and code development. AI tools also assisted figure layout. The authors remain responsible for checking the references, methods, claims, and results and for the final manuscript. Models evaluated in the experiments are identified separately from these assistance uses.

REFERENCES

Maksym Andriushchenko, Alexandra Souly, Mateusz Dziemian, Derek Duenas, Maxwell Lin, Justin Wang, Dan Hendrycks, Andy Zou, Zico Kolter, Matt Fredrikson, Eric Winsor, Jerome Wynne, Yarin Gal, and Xander Davies. AgentHarm: A benchmark for measuring harmfulness of LLM agents. *arXiv preprint arXiv:2410.09024*, 2024. URL https://arxiv.org/abs/2410.09024.

Kathryn Chaloner and Isabella Verdinelli. Bayesian experimental design: A review. *Statistical Science*, 10(3):273–304, 1995. doi: 10.1214/ss/1177009939.

Edmund Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. Counterexample-guided abstraction refinement. In *Computer Aided Verification*, pp. 154–169, 2000. doi: 10.1007/10722167_15.

Edoardo Debenedetti, Jie Zhang, Mislav Balunović, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. *arXiv preprint arXiv:2406.13352*, 2024. URL https://arxiv.org/abs/2406.13352.

Blaise Delattre, Cong Wang, and Yang Cao. CAGE: Certified authorization under typed-return uncertainty for tool-using agents. *arXiv preprint arXiv:2607.29190*, 2026. URL https://arxiv.org/abs/2607.29190.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems*, 2017. URL https://arxiv.org/abs/1705.08500.

Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011. doi: 10.1613/jair.3278. URL https://arxiv.org/abs/1003.3967.

Daniel Golovin, Andreas Krause, and Debajyoti Ray. Near-optimal bayesian active learning with noisy observations. In *Advances in Neural Information Processing Systems*, volume 23, 2010.

Shervin Javdani, Yuxin Chen, Amin Karbasi, Andreas Krause, J. Andrew Bagnell, and Siddhartha S. Srinivasa. Near optimal bayesian active learning for decision making. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, volume 33 of *Proceedings of Machine Learning Research*, pp. 430–438, 2014. URL https://proceedings.mlr.press/v33/javdani14.html.

Sumit Kumar Jha, Susmit Jha, Patrick Lincoln, Nathaniel D. Bastian, Alvaro Velasquez, Rickard Ewetz, and Sandeep Neema. Neuro symbolic reasoning for planning: Counterexample guided inductive synthesis using large language models and satisfiability solving. *arXiv preprint arXiv:2309.16436*, 2023.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998. doi: 10.1016/S0004-3702(98)00023-X.

Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Mudit Verma, Kaya Stechly, Siddhant Bhambri, Lucas Saldyt, and Anil Murthy. LLMs can't plan, but can help planning in LLM-modulo frameworks. In *International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2402.01817.

David Madras, Toniann Pitassi, and Richard Zemel. Predict responsibly: Improving fairness and accuracy by learning to defer. In *Advances in Neural Information Processing Systems*, 2018. URL https://arxiv.org/abs/1711.06664.

George C. Necula. Proof-carrying code. In *Proceedings of the 24th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pp. 106–119, 1997. doi: 10.1145/263699.263712.

Daeyoung Roh and Donghee Han. HALT: Verification-aware stopping for retrieval-augmented search agents. *arXiv preprint arXiv:2608.02009*, 2026. URL https://arxiv.org/abs/2608.02009.

Yangjun Ruan, Honghua Dong, Andrew Wang, Silviu Pitis, Yongchao Zhou, Jimmy Ba, Yann Dubois, Chris J. Maddison, and Tatsunori Hashimoto. Identifying the risks of LM agents with an LM-emulated sandbox. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2309.15817.

Santiago Torres-Arias, Hammad Afzali, Trishank Karthik Kuppusamy, Reza Curtmola, and Justin Cappos. in-toto: Providing farm-to-table guarantees for bits and bytes. In *28th USENIX Security Symposium*, pp. 1393–1410. USENIX Association, 2019. URL https://www.usenix.org/conference/usenixsecurity19/presentation/torres-arias.

Haoyu Wang, Christopher M. Poskitt, and Jun Sun. AgentSpec: Customizable runtime enforcement for safe and reliable LLM agents. In *Proceedings of the IEEE/ACM International Conference on Software Engineering*, 2026. URL https://arxiv.org/abs/2503.18666.

Martin L. Weitzman. Optimal search for the best alternative. *Econometrica*, 47(3):641–654, 1979. doi: 10.2307/1910412.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.03629.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024. URL https://arxiv.org/abs/2406.12045.

Yifan Zhang, Xinkui Zhao, Sai Liu, Hengxuan Lou, Guanjie Cheng, and Chang Liu. FAVA: Formal authorization for verified agents with evidence-backed permission graphs. *arXiv preprint arXiv:2607.27267*, 2026. URL https://arxiv.org/abs/2607.27267.

APPENDIX ROADMAP

Table 5: Supplement organization.

Section	Contents
A	Definitions, full proofs, complexity, and assumptions
B	Finite replay, exact compilation, and checked execution
C	Shared baselines and the four primary finite studies
D	Contract controls, mapping integrity, recovery, and receipt replay
E	Agent-level metrics, cost accounting, and interpretation
F	Implicit-support scaling, model ordering, and external controls
G	Reproducibility and evidence inventory

A FORMAL DEFINITIONS AND FULL PROOFS

A.1 NOTATION AND CERTIFIED STATES

Table 6: Notation for the formal model and proofs. All guarantees are relative to the registered problem, not to unmodeled real-world states.

Symbol	Meaning
$\Omega = \mathcal{W}$; $\mathcal{R}, \mathcal{A}, \mathcal{Q}$	registered world support; requirements, actions, and queries
$R(w,a), A^\star(w)$	prerequisites of action a; registered-safe actions in world w
$g_q(w), O_q$	deterministic outcome of query q in w and its public outcome set
$\mathrm{Auth}(q), K_q(o)$	query authority scope and requirements confirmed by outcome o
$\rho = (q,o,\eta), B_{q,o}$	accepted machine receipt and its receipt-consistent registered support
B, K, U	remaining world cell, confirmed requirements, and unused queries
$\mathrm{Safe}(B), \mathrm{Req}_a(B)$; $\mathrm{CAct}(B,K)$	common safe actions; action-specific requirements; certifiable actions
$\mathrm{RelReq}(B)$	requirements of every safe action in every world of B
$\mathrm{Adm}(B,K,U)$	implementation-admissible queries in Equation 27
$\mathcal{D}_a(K)$	authority-lifted decision region for action a
$\sigma(w) = (a(w), R(w,a(w)))$	exact signature in singleton-action experiments
$F = \{x : \Phi(x)\}$; $n_{\mathrm{role}}, n_{\mathrm{req}}, n_{\mathrm{act}}$	Boolean encoding of registered support; role, requirement, and action counts

We reserve *witness* for an outcome, receipt, or evidence object. The state component $K \subseteq \mathcal{R}$ stores the requirement identifiers that those objects have confirmed.

An outcome may confirm only requirements covered by its query's declared authority:

$$K_q(o) \subseteq \mathrm{Auth}(q). \tag{10}$$

The full registered support Ω is fixed before a probe is read. Its prior satisfies $p(w) > 0$ for every $w \in \Omega$ and $\sum_{w\in\Omega} p(w) = 1$. The metadata η binds the input manifest, extractor, and visibility checks. It records what was grounded; semantic correctness and registry completeness remain authored assumptions. In the Boolean scaling studies, $F = \{x : \Phi(x)\}$ denotes this support in assignment coordinates; $\mathcal{W}$ and Ω denote the corresponding registered worlds.

Definition 1 (Receipt-conditioned registered support)**.** *Let $\rho = (q,o,\eta)$ be an accepted receipt whose query and outcome are registered and whose deterministic outcome is reachable. For any nonempty current registered cell $B \subseteq \Omega$, define*

$$B_{q,o} = \{w \in B : g_q(w) = o\}, \qquad p(o \mid B, q) = \frac{\sum_{w\in B_{q,o}} p(w)}{\sum_{w\in B} p(w)}. \tag{11}$$

Strict positivity makes every nonempty cell have positive mass. The posterior associated with any current cell is

$$p_B(w) = \mathbf{1}\{w \in B\}\frac{p(w)}{\sum_{u\in B} p(u)}. \tag{12}$$

At the probe boundary $B = \Omega$, an accepted receipt induces residual support $B_{q,o}$ and posterior $p_\rho := p_{B_{q,o}}$. At later histories, the posterior is indexed by the current cell rather than by the latest receipt alone. The implementation rejects an empty $B_{q,o}$. Conditioning may remove registered worlds inconsistent with the observation; it cannot add worlds or repair an incomplete registry.

A state is *certified* when $\mathrm{CAct}(B, K) \neq \varnothing$, where

$$\begin{aligned} \mathrm{Safe}(B) &:= \bigcap_{w \in B} A^\star(w), \\ \mathrm{Req}_a(B) &:= \bigcup_{w \in B} R(w, a), \\ \mathrm{CAct}(B, K) &:= \{a \in \mathrm{Safe}(B) : \mathrm{Req}_a(B) \subseteq K\}. \end{aligned} \tag{13}$$

The registry fixes $K_0 \subseteq \mathcal{R}$ before execution. Each member is assumed to be a valid initial confirmation from an authorized source for the current task. When no such evidence is assumed, $K_0 = \varnothing$. Grounded transitions establish query provenance for $K(h) \setminus K_0$, not for K_0 itself. For grounded history $h = ((q_1, o_1), \ldots, (q_T, o_T))$, the complete compiler state is

$$\begin{aligned} B(h) &= \{w \in \Omega : \forall t,\ g_{q_t}(w) = o_t\}, \\ K(h) &= K_0 \cup \bigcup_{t=1}^{T} K_{q_t}(o_t), \\ U(h) &= \mathcal{Q} \setminus \{q_1, \ldots, q_T\}. \end{aligned} \tag{14}$$

Let $\Pi^{\mathrm{nr}}_{\mathrm{cert}}$ be the deterministic policy trees over states (B, K, U) that never repeat a query. An internal node selects $q \in U$, has one child for every reachable outcome, and transitions to $(B_{q,o}, K \cup K_q(o), U \setminus \{q\})$. Every terminal action belongs to $\mathrm{CAct}(B, K)$. The objective is

$$\pi^\star \in \arg\min_{\pi \in \Pi^{\mathrm{nr}}_{\mathrm{cert}}} \mathbb{E}_\pi \left[\sum_t c(q_t) \right]. \tag{15}$$

The base objective assigns equal terminal loss to all registered-safe actions, while exposure still requires membership in $\mathrm{CAct}(B, K)$. Query cost alone may therefore prefer a confirmation-free defer action over a costly repair.

Human-query model. A human source uses the same query interface, with fixed cost and a deterministic outcome per registered world. Each world fixes the source's response for the episode, while K records the confirmations actually acquired. For nonempty B, Equation 11 gives $B_{q,o} = B$ exactly when $g_q(w) = o$ for all $w \in B$. A belief-preserving query therefore has a known outcome on that cell; an uncertain response can leave machine facts unchanged while reducing the full world cell. History-dependent responses and authorization withdrawal require a different transition model. We do not estimate real human or service availability, latency, error rates, or strategic behavior. InformationRouting-1 uses a specified proof-availability distribution; scripted LiveProbe answers and abnormal responses are interventions, not frequency estimates.

A.2 Decision regions and monotonicity

Source effects in the running example.

Source query	Descriptive effect	Authority effect
Regression test: bug reproduced	Rules out worlds where the bug does not reproduce	Adds no owner approval
Compatibility check: passed	Rules out incompatible-patch worlds	Certifies compatibility only
Owner answer	May leave machine facts unchanged	Can confirm current approval
Policy location	Identifies who is entitled to answer	Not current approval

Definition 2 (Authority-lifted decision region)**.** *For action a and confirmed-requirement set K, define*

$$\mathcal{D}_a(K) := \{w \in \mathcal{W} : a \in A^\star(w) \ \wedge\ R(w, a) \subseteq K\}. \tag{16}$$

Lemma 3 (Decision-region equivalence). *State (B, K) is certified if and only if there exists $a \in \mathcal{A}$ such that $B \subseteq \mathcal{D}_a(K)$.*

Proof of Lemma 3. Suppose (B, K) is certified. Choose any $a \in \mathrm{CAct}(B, K)$. For every $w \in B$, $R(w, a) \subseteq \mathrm{Req}_a(B) \subseteq K$ and $a \in A^\star(w)$, hence $w \in \mathcal{D}_a(K)$ and $B \subseteq \mathcal{D}_a(K)$.

Conversely, if $B \subseteq \mathcal{D}_a(K)$, every $w \in B$ satisfies $R(w, a) \subseteq K$ and $a \in A^\star(w)$. Hence $\mathrm{Req}_a(B) \subseteq K$ and $a \in \mathrm{Safe}(B)$, so $a \in \mathrm{CAct}(B, K)$. □

Proposition 4 (Strict separation from ordinary decision-region stopping). *Let ordinary region completion be $\mathrm{DR}(B) \iff \mathrm{Safe}(B) \neq \varnothing$. Then*

$$\mathrm{Cert}(B, K) \iff \bigvee_{a \in \mathrm{Safe}(B)} [\mathrm{Req}_a(B) \subseteq K], \qquad \mathrm{Cert}(B, K) \implies \mathrm{DR}(B). \tag{17}$$

For any ordinary decision-region policy with terminal action a_ℓ at each leaf $\ell \in \mathcal{L}$, its unauthorized-action mass is

$$M_{\mathrm{unauth}}(\pi) = \sum_{\ell \in \mathcal{L}} P_\pi(\ell) \mathbf{1}\{a_\ell \notin \mathrm{CAct}(B_\ell, K_\ell)\}. \tag{18}$$

There exists a two-world instance with ordinary region completion at the root but $M_{\mathrm{unauth}} = 1$.

Proof. Equation 17 expands the definition of CAct. Every certifiable action is common and safe, proving the implication. Equation 18 sums the leaves whose selected action lacks safety or action-specific confirmation coverage. For strictness, take two positive-probability worlds w_0, w_1 with $A^\star(w_0) = A^\star(w_1) = \{a\}$, $R(w_0, a) = R(w_1, a) = \{r\}$, and initial $K = \varnothing$. Ordinary decision-region stopping accepts the shared action a at the root, while certified stopping rejects because $r \notin K$. The sole ordinary terminal leaf therefore has unauthorized mass one. □

Proposition 5 (World-belief-only stopping is insufficient). *If two reachable histories h, h' satisfy $B(h) = B(h')$ but $\mathrm{Cert}(B(h), K(h)) \neq \mathrm{Cert}(B(h'), K(h'))$, no predicate of B alone agrees with authority-complete stopping on every reachable history. Such an instance exists even with one world and one action.*

Proof. A predicate of B must return the same value on h and h', while certification returns different values. For existence, take one world w, one action a with $A^\star(w) = \{a\}$ and $R(w, a) = \{r\}$, and initial $K_0 = \varnothing$. A registered authority query has one constant outcome and confirms r. Before and after that query the belief is $\{w\}$; certification changes from false to true. □

Proposition 6 (Exact augmented-state reduction). *Under the finite deterministic assumptions, (B, K, U) is a sufficient Markov information state. The query transitions in Equation 14, terminal actions $a \in \mathrm{CAct}(B, K)$, and query costs define an equivalent finite stochastic-shortest-path problem. With*

$$\widehat{\mathcal{D}}_a := \{(w, K) : a \in A^\star(w),\ R(w, a) \subseteq K\}, \tag{19}$$

we have $a \in \mathrm{CAct}(B, K)$ if and only if $B \times \{K\} \subseteq \widehat{\mathcal{D}}_a$. Directly materializing the world–confirmation pairs uses at most $|\Omega| 2^{|\mathcal{R}|}$ pairs.

Proof. Given (B, K, U) and a selected query, the registered outcome maps and prior determine every successor and its probability; earlier history contributes no additional information. The terminal condition and cost also depend only on this state. Standard state aggregation therefore maps every history-dependent policy to an equal-cost Markov policy and conversely. The region equivalence follows directly from Equation 13; counting all (w, K) pairs gives the stated representation upper bound. A DRD/HEC method that carries this controlled confirmation state can simulate the formulation. Ordinary world-only regions cannot do so by Proposition 5. □

Recovering confirmations on reachable states. For a fixed registry and initial K_0, any nonempty reachable (B, K, U) satisfies

$$K = K_0 \cup \bigcup_{q \in \mathcal{Q} \setminus U} K_q(g_q(w)) \qquad \text{for any } w \in B.$$

Every used query has the same outcome on all worlds still in B, so this union is independent of the chosen w and equals Equation 14. Thus (B, U) can recover K in this model. Explicit K supports authorization accounting; Proposition 5 concerns stopping from B alone, not minimality of the augmented representation. In particular, two histories with the same reachable (B, U) cannot have different K under these fixed contracts.

Lemma 7 (Certification monotonicity)**.** *If $B' \subseteq B$ and $K \subseteq K'$, then $\mathrm{CAct}(B, K) \subseteq \mathrm{CAct}(B', K')$.*

Proof. Take $a \in \mathrm{CAct}(B, K)$. From $B' \subseteq B$, $\mathrm{Req}_a(B') \subseteq \mathrm{Req}_a(B)$ and $\mathrm{Safe}(B) \subseteq \mathrm{Safe}(B')$. Hence $\mathrm{Req}_a(B') \subseteq K \subseteq K'$ and $a \in \mathrm{Safe}(B')$, so $a \in \mathrm{CAct}(B', K')$. □

Exact-signature instantiation. The general formulation permits set-valued registered-safe action sets $A^\star(w)$. The implicit-support scaling experiments use the restricted singleton-action instantiation

$$A^\star(w) = \{a(w)\} \qquad \text{for every } w \in F, \tag{20}$$

where $a(w)$ is the unique registered settlement action. The compiler-relevant signature is

$$\sigma(w) := (a(w), R(w, a(w))). \tag{21}$$

The scalability registries store one prerequisite set $R_0(w)$ per world and instantiate $R(w, a) = R_0(w)$ for all $a \in A^\star(w)$. The finite compiler and its specification suite exercise the general action-indexed case; the Pair-SAT scaling results use this shared-prerequisite subfamily. We reserve j for a Boolean refinement coordinate and q for a registered source query. In the scaling registries, every eligible coordinate j has a declared query q_j; the proposer orders coordinates, while the compiled policy decides which q_j to issue at each runtime state. For an implicit Boolean support formula $\Phi(x)$ and selected coordinates S, the two-world oracle uses PySAT/Glucose3 to check the satisfiability of

$$\Phi(x) \wedge \Phi(x') \wedge \bigwedge_{j \in S} (x_j \leftrightarrow x'_j) \wedge [\sigma(x) \neq \sigma(x')]. \tag{22}$$

Let d be the number of support variables, b the signature-bit count, r the number of role variables, and $|\Phi|$ the number of support clauses. For our truth-table signature encoder, the pair CNF has $2d + 3b$ variables and $2|\Phi| + 2^{r+1}b + 2|S| + 4b + 1$ clauses: two support copies, two truth-table signature encodings, selected-coordinate equalities, XOR definitions, and one disagreement clause. SAT returns a conflicting pair; UNSAT proves exactness. Formula construction is exponential in r for this encoder, and each SAT call remains worst-case exponential. Refinement accepts at most one proposal per coordinate, but this does not bound solver work within a round. Proposals affect refinement work, not runtime query selection or action exposure. For selected coordinates $S \subseteq [d]$ and projection z, let $C_S(z) := \{w \in F : w_S = z\}$. We call S *signature adequate* when σ is constant on every nonempty $C_S(z)$. Define

$$\mathcal{H}_F := \{\Delta(w, w') : w, w' \in F,\ \sigma(w) \neq \sigma(w')\}, \qquad \Delta(w, w') := \{j : w_j \neq w'_j\}. \tag{23}$$

Theorem 8 (Exact-signature conflict hitting and proposer bound)**.** *Under Equation 20, the following are equivalent: (i) S is signature adequate; (ii) S hits every edge in $\mathcal{H}_F$; and (iii) every nonempty S-cell C has a unique common safe action a_C with $\mathrm{Safe}(C) = \{a_C\}$ and a common requirement set R_C with $R(w, a_C) = R_C$ for every $w \in C$. Repeatedly adding any unselected coordinate from a returned edge terminates after at most d counterexamples. If a sufficient set $S^\star$ is fixed and the proposer always chooses from $\Delta(w, w') \cap S^\star$, termination takes at most $|S^\star|$ counterexamples.*

Proof of Theorem 8. By definition, S is signature inadequate exactly when there exist $w, w' \in F$ such that $w_S = w'_S$ but $\sigma(w) \neq \sigma(w')$. The projection equality is equivalent to $S \cap \Delta(w, w') = \varnothing$, so signature adequacy is exactly the hitting-set condition for $\mathcal{H}_F$.

Now consider any nonempty S-cell C. If S is signature adequate, there exist a_C and R_C such that $\sigma(w) = (a_C, R_C)$ for every $w \in C$. By Equation 20,

$$\mathrm{Safe}(C) = \bigcap_{w \in C} \{a(w)\} = \{a_C\}, \qquad \mathrm{Req}_{a_C}(C) = \bigcup_{w \in C} R(w, a_C) = R_C.$$

Conversely, if every nonempty S-cell has a unique common safe action and a common requirement set, singleton registered-safe action sets imply that all worlds in the cell have the same $a(w)$ and $R(w, a(w))$. Hence σ is constant on every cell and S is signature adequate.

Whenever the oracle returns a conflict, its edge is nonempty and disjoint from the current S. Adding any returned coordinate strictly increases $|S|$, hence the loop terminates after at most d conflicts. If $S^\star$ is sufficient, every returned edge intersects $S^\star$; choosing an unselected coordinate from that intersection strictly increases $|S \cap S^\star|$. After at most $|S^\star|$ such choices, S contains a sufficient set and the oracle must accept. □

Signature adequacy does not imply cost-preserving query selection. Two queries can induce the same partition while costing 100 and 1; retaining only the first preserves the signature but can raise optimal acquisition cost. Finite replay therefore checks retained-query and quotient policies against the full residual policy separately (Appendix D.6). This additional equivalence check is not a consequence of Theorem 8. With the full query registry fixed, refinement order does not change the Bellman problem. If retained coordinates also restrict the available queries, the compiler is optimal only within that restriction unless the finite checks establish $V_Q = V_S = V_F$.

Lemma 9 (Confirmation conservation)**.** *For grounded history h, the confirmed-requirement set is exactly Equation 14. If every contract satisfies Equation 10, then each $r \in K(h) \setminus K_0$ is supported by an observed query whose declared authority contains r.*

Proof. The root contains exactly K_0. A transition on (q, o) replaces K by $K \cup K_q(o)$ and introduces no other confirmation. Induction on history length gives Equation 14. Contract validation enforces $K_q(o) \subseteq \mathrm{Auth}(q)$, establishing the authority statement. □

Corollary 10 (Mandatory human authority)**.** *Let $r \notin K_0$ satisfy $r \in R(w, a)$ for every initially reachable w and every $a \in A^\star(w)$. If no reachable machine outcome witnesses r, then every certified root-to-action path contains a human query outcome that witnesses r.*

Proof. Every certified leaf must contain r in its confirmed-requirement set. By Lemma 9, some queried outcome adds r. The premise rules out every machine outcome, so the witnessing query is human. □

A.3 Soundness and compositional failure handling

Theorem 11 (Authority-complete safe execution)**.** *Under the trusted-initial-confirmation premise, if grounding and verification accept a certificate, every emitted execution leaf uses only initial confirmations in K_0 or confirmations supplied by observed outcomes within the queried sources' authority scopes. Its action is safe and its requirements are confirmed in every reachable world.*

Proposition 12 (Compositional fail closed)**.** *Along the declared execution path, a parse, grounding, binding, compilation, evaluation, or projection error; an invalid terminal candidate; or the absence of a continuation that certifies every reachable branch yields no public autonomous action.*

Proof of Theorem 11. Consider an emitted leaf with history h, belief B_h, confirmed requirements $K(h)$, and action a. Grounding accepts only registered outcomes, and compilation checks $a \in \mathrm{CAct}(B_h, K(h))$. Thus a is safe in every reachable world. Each required confirmation either belongs to K_0, whose validity is assumed, or belongs to $K(h) \setminus K_0$, whose queried authority support follows from Lemma 9. □

Corollary 13 (Receipt soundness on the registered residual support)**.** *Under the same initial-confirmation premise, let $\rho = (q, o, \eta)$ be accepted against the full registered support Ω, and let $B_{q,o}$ be Definition 1. If the frozen compiler and verifier accept from $(B_{q,o}, K_0 \cup K_q(o), U_0 \setminus \{q\})$,*

then every reachable leaf ℓ with full history h_ℓ uses only trusted initial confirmations and authority-scoped queried confirmations and satisfies

$$a_\ell \in \bigcap_{w \in B_\ell} A^\star(w), \qquad \bigcup_{w \in B_\ell} R(w, a_\ell) \subseteq K_\ell, \tag{24}$$

where $B_\ell = B(h_\ell) \subseteq B_{q,o}$ and $K_\ell = K(h_\ell)$. A failed receipt check or an empty residual support exposes no autonomous action. The statement is only about receipt-consistent registered worlds; it does not cover $w \notin \Omega$ or an incorrect extractor, registry, authority scope, or safe-action label.

Proof. Receipt validation maps the observed value to a registered (q, o) and rejects an empty branch. The observed outcome contributes exactly $K_q(o) \subseteq \mathrm{Auth}(q)$, and all subsequent transitions are ordinary grounded compiler transitions on $B_{q,o} \subseteq \Omega$. At each reachable leaf, Theorem 11 applies to the worlds consistent with that leaf's full history. Equivalently, each initial residual world reaches a leaf whose action is safe and confirmed for that world. The fail-closed clause follows from Proposition 12. □

Proof of Proposition 12. The public projection is downstream of strict parsing, contract binding, registered-world coverage, exact compilation, per-world evaluation, and projection. Each stage either returns a complete typed object or raises an error; the runner maps detected provider and validation errors to a cell with no prediction. The projection rejects unresolved roots and any evaluation in which a registered world lacks a selected action. Therefore an error or an uncertifiable reachable branch yields no public autonomous action. If a prediction is non-null, evaluation has traversed every registered world and Theorem 11 applies to every exposed leaf. The guarantee is scoped to the declared pipeline and registered semantics. Signed activation blocks fixture-covered map corruption; it cannot establish the meaning of raw observations outside the registered fixtures and ontology (Appendix D.3). □

A.4 Bellman optimality for grounded non-repeating policies

For belief B, confirmed requirements K, and unused queries U, let $B_{q,o} := \{w \in B : g_q(w) = o\}$ and

$$\mathrm{RelReq}(B) := \bigcup_{w \in B} \bigcup_{a \in A^\star(w)} R(w, a). \tag{25}$$

Define

$$\mathrm{Split}(q; B) \iff \exists w, w' \in B : \; g_q(w) \neq g_q(w'). \tag{26}$$

The admissible set retains queries that split B or add an action-relevant confirmation:

$$\begin{aligned} \mathrm{Adm}(B, K, U) := \{q \in U : \; & \mathrm{Split}(q; B) \\ & \vee \; \exists o \text{ reachable} : \; K_q(o) \cap (\mathrm{RelReq}(B) \setminus K) \neq \varnothing\}. \end{aligned} \tag{27}$$

Thus descriptive queries may narrow the world cell without confirming any requirement. The union in $\mathrm{RelReq}(B)$ also retains a confirmation-bearing query for an action that may become common only after a later split.

For a reachable outcome, define the query continuation

$$\mathcal{C}_q(B, K, U) := c(q) + \sum_{o : B_{q,o} \neq \varnothing} p(o \mid B, q) V(B_{q,o}, K \cup K_q(o), U \setminus \{q\}). \tag{28}$$

The exact recursion is

$$V(B, K, U) = \begin{cases} 0, & \mathrm{CAct}(B, K) \neq \varnothing, \\ \min\limits_{q \in \mathrm{Adm}(B,K,U)} \mathcal{C}_q(B, K, U), & \mathrm{CAct}(B, K) = \varnothing, \; \mathrm{Adm}(B, K, U) \neq \varnothing, \\ +\infty, & \text{otherwise}. \end{cases} \tag{29}$$

Lemma 14 (No-progress query deletion)**.** *If $q \in U \setminus \mathrm{Adm}(B, K, U)$, then g_q is constant on B with a unique reachable outcome $o^\star$, and for every $B' \subseteq B$ and $a \in \mathrm{Safe}(B')$,*

$$\mathrm{Req}_a(B') \setminus \big(K \cup K_q(o^\star)\big) = \mathrm{Req}_a(B') \setminus K. \tag{30}$$

Any grounded non-repeating policy beginning with q can therefore delete that first step, preserve every terminal action, certification decision, and registered terminal loss, and reduce expected cost by $c(q) \geq 0$.

Proof. Because $\mathrm{Split}(q; B)$ is false, every $w \in B$ has the same outcome $o^\star$. Because the confirmation clause in Equation 27 is also false, $K_q(o^\star) \cap (\mathrm{RelReq}(B) \setminus K) = \varnothing$. For every $B' \subseteq B$ and $a \in \mathrm{Safe}(B')$, $\mathrm{Req}_a(B') \subseteq \mathrm{RelReq}(B)$. Adding $K_q(o^\star)$ therefore removes no unresolved prerequisite of any action that can be common in a descendant cell, proving Equation 30. Copy the unique reachable child subtree to the parent state and never select q again. The extra unused query does not invalidate any copied choice; the displayed equality preserves every certifiable-action set, while beliefs, posterior terminal losses, and actions are unchanged. The copied policy removes exactly the nonnegative cost $c(q)$. □

Theorem 15 (Optimality over grounded non-repeating certified policies)**.** *Fix the complete registered problem and an initial state (B, K, U). Assume finite registered support and query set, strictly positive registered priors, deterministic total outcome maps, exact conditional branch probabilities, and nonnegative query costs. Then Equation 29 minimizes $\mathbb{E}_\pi[\sum_t c(q_t)]$ over all $\pi \in \Pi^{\mathrm{nr}}_{\mathrm{cert}}$ starting from that state.*

Proof. Proceed by induction on $|U|$. When $|U| = 0$, a certified state has zero remaining cost; otherwise no admissible certified continuation exists and the value is $+\infty$.

Assume the claim for states with fewer than m unused queries and consider a nonterminal state with $|U| = m$. Lemma 14 removes any first query outside $\mathrm{Adm}(B, K, U)$ without increasing cost, so some optimum begins with an admitted query. Such a query pays $c(q)$ and supplies a continuation for every reachable outcome. Each successor has $m - 1$ unused queries. By induction, its recursive value is the minimum continuation cost. Equation 28 is therefore the minimum among policies whose first query is q; minimizing over admitted q yields the global optimum in $\Pi^{\mathrm{nr}}_{\mathrm{cert}}$. □

Optional terminal-action loss. Let $L(w, a) \in \mathbb{Q}_{\geq 0}$ be fixed before compilation for every registered safe world–action pair, and define the posterior terminal risk

$$\bar{L}_B(a) := \sum_{w \in B} p(w \mid B) L(w, a), \qquad T_L(B, K) := \min_{a \in \mathrm{CAct}(B,K)} \bar{L}_B(a), \tag{31}$$

where the minimum of an empty set is $+\infty$. Substituting V_L for V in Equation 28 gives $\mathcal{C}^L_q$. The loss-aware recursion is

$$V_L(B, K, U) = \min \left\{ T_L(B, K), \min_{q \in \mathrm{Adm}(B,K,U)} \mathcal{C}^L_q(B, K, U) \right\}. \tag{32}$$

At a certified state, the compiler stops when $T_L(B, K)$ is no larger than every admissible query continuation; otherwise it gathers more evidence. Stop is selected on a value tie, and equal-loss actions retain the lexicographic tie break.

Corollary 16 (Loss-aware soundness and optimality)**.** *If L is finite, nonnegative, fixed, and depends only on (w, a), then Equation 32 exposes only certified actions and minimizes*

$$J_L(\pi) = \mathbb{E}_\pi \left[\sum_t c(q_t) + L(W, a_\ell) \right] \tag{33}$$

over $\Pi^{\mathrm{nr}}_{\mathrm{cert}}$. Setting $L \equiv 0$ recovers Equation 29 and its stop-on-certification policy.

Proof. The stop arm is restricted to $\mathrm{CAct}(B, K)$, preserving soundness. A query outside $\mathrm{Adm}(B, K, U)$ changes neither B, the certifiable-action sets, nor posterior terminal risk; Lemma 14 therefore removes it without increasing J_L. Induction on $|U|$, comparing the certified stop arm with every admitted query arm, proves optimality. The loss function ranks certified actions and query continuations but does not change the certifiable-action set. □

Descriptive-query example. Two equally likely worlds may have different safe actions, no unresolved confirmations, and a zero-cost runtime trace that distinguishes them without confirming any requirement. Equation 27 admits this trace through its descriptive disjunct, so the compiler can query it and act safely in both branches.

Proposition 17 (Query-set monotonicity)**.** *For fixed (B, K) and $U \subseteq U'$,*

$$V(B, K, U') \leq V(B, K, U), \tag{34}$$

where $+\infty$ denotes the absence of a certified policy.

Proof. Every admissible policy using only queries in U remains available when the unused set is U'. Enlarging the optimization domain cannot increase its minimum. In particular, removing human queries cannot create a new certified policy; a lower observed cost after removal can merely reflect early unresolved termination. □

Theorem 18 (Termination and state-space bound)**.** *Let $n = |\mathcal{W}|$, $r = |\mathcal{R}|$, $k = |\mathcal{A}|$, and $m = |\mathcal{Q}|$. Recursion depth is at most m, and the number of possible memoized states satisfies*

$$S \leq (2^n - 1)2^r 2^m. \tag{35}$$

Ignoring rational-arithmetic bit complexity, direct memoized evaluation takes $O(S[m(n+r)+nkr])$ time and $O(S)$ state space in addition to the serialized policy tree.

Proof. Every recursive transition removes one query from U, so a root-to-leaf path has depth at most m. A state is identified by a nonempty subset of n worlds, a subset of r requirements, and a subset of m queries, yielding $(2^n - 1)2^r 2^m$ possible triples. Memoization evaluates each at most once. At a state, at most m queries are tested; partitioning scans at most n worlds and confirmation operations touch at most r requirements. Constructing CAct and RelReq naively checks at most nkr world–action–requirement entries. □

The historical finite-input experiments used a 64-world, 12-query compiler configuration. The earlier registered-policy suite has at most 27 worlds, three queries, and three requirements per case. These experimental limits are not rejection rules in the released general finite reference compiler. They also differ from latent Boolean support size: the cardinality–support interaction reaches 131,072 worlds, while the separate shape study reaches 4,330,260. Scaling records describe compilation on bounded role assignments or representatives. They do not establish a full-reference, per-latent-world equivalence audit at those support sizes. The exponential bound above concerns materialized compiler inputs; the finite replay checks in Algorithm A1 do not supply a symbolic optimality certificate for the large-support studies.

A.5 Uniform reduction and proposal invariance

Corollary 19 (Uniform reduction)**.** *If the complete finite support is already registered and inexpensive to enumerate, direct enumeration (*Uniform-GAC*) supplies all information required by the compiler without a stochastic proposal. In this regime, model-guided refinement-coordinate selection is not necessary for correctness or optimization within the same declared admissible class.*

Proof. When the complete finite support and normalized prior are already registered, direct enumeration supplies every candidate assignment required by the binder. The subsequent grounding, admissibility predicate, Bellman recursion, evaluation, and projection are identical to the checked compiler path. A stochastic proposal is therefore unnecessary for coverage, certified compilation, or optimization within the same declared class. □

Proposition 20 (Proposal invariance under assignment-preserving renaming)**.** *Suppose two accepted proposals enumerate the same registered assignments with the same normalized probabilities and differ only in world identifiers, ordering, or a common positive scaling of prior weights. Under the declared deterministic tie breaking, they induce the same public query policy.*

Proof. Match worlds by registered assignment. This bijection preserves registered-safe action sets, the complete action–requirement relation, query outcomes, outcome witnesses, authority scopes, and normalized probabilities. Hence it preserves each reachable partition, conditional branch probability, terminal predicate, and Bellman value. Induction on $|U|$ gives the same minimizing query at corresponding states. Tie breakers depend on cost, worst-case query count, query kind, and query identifier, not on internal world names or ordering. Public projection removes those internal names. □

Table 7: Guarantee assumptions and implementation boundary.

Assumption	Implemented check	Scope boundary
Artifact binding	Signed manifest and map activation	Meaning of the bound contents
Outcome grounding	Public IDs and source-map bindings	Truth and authenticity of raw observations
Settlement authority	$K_q(o) \subseteq \mathrm{Auth}(q)$	Correctness of authored source scopes
Initial confirmations	Registry and identifier checks for K_0	Validity of the initial authorization
World support	Registered-world coverage	Unregistered real worlds
Action exposure	$\mathrm{CAct}(B, K)$; action-indexed tests	Correctness of authored safe-action labels
Exact refinement	Signature encoders and SAT solver	Encoder/SAT correctness; general action-indexed implicit scaling
Optimization scope	Bellman recursion and Lemma 14	Policies outside $\Pi^{\mathrm{nr}}_{\mathrm{cert}}$ or the finite model
Terminal objective	Zero-loss reference; separate loss-aware finite planner	Calibration of terminal losses
Prior	Exact rational branch probabilities	Real-world calibration
Deployment budgets	Query counts and costs computed	Configurable caps and constrained optimization not implemented
Refinement resources	At most d accepted coordinates	Per-call solver cost and deployment limits

B Compiler and Verifier Algorithms

Algorithm A1 describes finite offline replay, including optional receipt conditioning and quotient comparison. Algorithm A3 is the finite action-exposure interface: it begins at the registered root and reconstructs every reached state from checked outcomes. Both use the zero-loss reference routine in Algorithm A2; a replay result does not itself grant permission to execute. These algorithms do not specify the adapters used in the agent evaluations.

Algorithm A1: Finite offline replay: activate, condition, refine, compile, and audit.

1. Authenticate the deployment manifest and execute the map-binding checks in Algorithm A3 before accepting any mapped observation. Parse and bind the problem and any provider proposal under their declared schemas. Reject unknown, duplicate, or malformed worlds, sources, outcomes, requirements, and actions.
2. Establish the complete registered support Ω and validate exact assignment coverage plus $K_q(o) \subseteq \mathrm{Auth}(q)$ for every outcome contract *before* applying an optional receipt.
3. Without a receipt, set $(B_0, K, U) = (\Omega, K_0, U_0)$. For a receipt $\rho = (q, o, \eta)$, verify repository containment, extractor identity, strict JSON projection, manifest-bound source metadata, H0/H1 invariance, and the exclusion of gold/action/post-treatment fields; use the author-provided outcome map to obtain a registered machine (q, o) and require q to equal the deterministic registry policy's root query.
4. If receipt checks fail, return unresolved. For an accepted receipt, form $B_0 = B_{q,o}$ by filtering Ω only, reject $B_0 = \varnothing$, set $K = K_0 \cup K_q(o)$ and $U = U_0 \setminus \{q\}$, and account for $c(q)$. With no receipt, retain the root state from step 3. Receipt conditioning may shrink the guarantee universe from Ω to B_0 but never defines a new support.
5. When quotienting is requested, run Pair-SAT refinement on B_0 until every cell is exact-signature homogeneous under Equation 21, then minimize the retained coordinate set under the same exact adequacy check.
6. Lower each retained coordinate j through its registered binding $j \mapsto q_j$. Compile the full residual reference, the retained-query full-world problem, and the aggregated quotient with the same `Solve` routine. Add or delete retained queries only when complete per-world trace and exact rational-cost equivalence passes.
7. Evaluate the accepted policy separately in every $w \in B_0$. Reject any missing branch, repeated query, unsafe action, authority-incomplete leaf, or requested quotient-equivalence failure. Query counts and costs are reported; deployment caps require separate enforcement.
8. Record a replay prediction only when every world in the declared residual support resolves; otherwise record unresolved. This output is an offline evaluation result, not an action-exposure token.

The full registry and authenticated map activation precede receipt conditioning. All full-reference comparisons and per-world audits in this algorithm operate on finite materialized replay inputs; they are not steps of the large-support scaling protocol. Receipt-conditioned guarantees range only over $B_{q,o} \subseteq \Omega$. Evaluations defined directly over registered outcomes begin from the authenticated finite problem; raw-observation executions additionally exercise map activation. Fixture semantics remain part of the trusted boundary.

Algorithm A2: Memoized exact compilation `Solve`(B, K, U).

1. If $\mathrm{CAct}(B, K) \neq \varnothing$, emit the lexicographically first action in $\mathrm{CAct}(B, K)$ with value zero.
2. Construct $\mathrm{Adm}(B, K, U)$. For each q in this set, partition B by g_q.
3. Recursively solve every nonempty branch $(B_{q,o}, K \cup K_q(o), U \setminus \{q\})$. Discard q if any branch is infeasible.
4. Compute the exact rational continuation cost using Equation 6, with branch probabilities from Equation 11.
5. Select the candidate ordered by expected cost, worst-case query count, machine-before-human kind, and query identifier. If none remains, return unresolved.

Each transition removes one query, giving an acyclic dynamic program.

Algorithm A2 is the implemented $L \equiv 0$ specialization. An always certifiable Defer permits immediate stopping under this objective. The separate loss-aware planner used in SourceGround-8 and the constructed-loss study compares the certified stop value in Equation 31 with every admitted query continuation.

Algorithm A3: Signed activation and action exposure.

1. Authenticate the registered problem and source-query contracts. Check that every observation map is bound to its source and satisfies the fixture-conformance predicate. Reject incomplete or inconsistent bindings before compiler construction.
2. Compile only from the full registered root (Ω, K_0, U_0) after activation. A caller-supplied conditioned belief or confirmation set cannot authorize execution.
3. Map machine observations through the activated map; their raw-source authenticity is an external premise. Verify human receipts against the authorized signer and current execution binding. Accept only registered outcomes, then update B, K, and U.
4. Recompute the terminal state and expose an action only if it matches the reached node and belongs to $\mathrm{CAct}(B, K)$. Any failed check returns unresolved.

Activation precedes compilation and is checked again during execution. Valid initial confirmations, trusted deployment credentials, execution freshness, and raw-observation authentication remain external premises.

Table 20 summarizes the finite implementation checks. The Pair-SAT scaling path instead uses Equation 21's singleton-action signature.

B.1 Comparison with Related Systems

Table 8 distinguishes acquisition planning from adjacent checking tasks.

Table 8: Closest-system boundary. These systems share trusted checking but expose different inputs and outputs, so the table reports capabilities rather than a common score.

System	Verified object	Authority-specific acquisition
AgentSpec / FAVA	Runtime action under policy or permission evidence	No cost-sensitive next-source acquisition policy
CAGE	Fixed action under typed-return uncertainty	No next-source acquisition policy
HALT / ordinary world-only HEC	Retrieval evidence or ordinary decision region	No requirement-indexed confirmation state
RightSource	Decision region and acquisition policy	Selects machine/human authorities and certifies a common safe action

C Primary Finite Studies

C.1 Fixed Orders with Adaptive Skipping and Stopping

Machine-first (MF), MF+skip, best-fixed+skip (BF), receding two-step lookahead (L2), and exact planning (EX) use identical observations, action requirements, costs, and losses. MF orders admissible queries by machine-before-human kind, then cost and identifier, and optimizes stopping along that continuation. MF+skip uses that same order but also permits permanent skips. BF considers every complete permutation and selects one minimizing the objective at the initial prior. Both skip policies adapt stopping and skipping to observations; they do not adapt the relative order of acquired queries.

For a permutation $\sigma = (q_1, \ldots, q_m)$, let $F_\sigma(h, i)$ be the minimum remaining loss after acquired history h, with positions before i no longer available. Let $T(h)$ be the minimum conditional loss of a currently certified action. The fixed-order recursion is

$$F_\sigma(h,i) = \min\Big\{T(h),\ F_\sigma(h,i+1),\\ c(q_i) + \sum_{o:\,P(o|h,q_i)>0} P(o \mid h, q_i) F_\sigma(h \cdot (q_i, o), i+1)\Big\}. \tag{36}$$

The acquisition option is included only when q_i is unused and admissible; $F_\sigma(h, m+1) = T(h)$. A skip leaves h, the belief, confirmations, and the history-defined unused set $U(h)$ unchanged, but

advances i. The skipped query cannot be revisited by this policy. The cursor is therefore part of the memoized state. Equal values prefer stop, then skip, then acquisition; tied actions and full permutations use lexical order. MF, L2, and EX use stop-first and lexical tie rules.

For expected query cost plus terminal loss J_π, the MF deficit decomposes as

$$J_{\mathrm{MF}} - J_{\mathrm{EX}} = (J_{\mathrm{MF}} - J_{\mathrm{MF+skip}}) + (J_{\mathrm{MF+skip}} - J_{\mathrm{BF}}) + (J_{\mathrm{BF}} - J_{\mathrm{EX}}). \tag{37}$$

These terms respectively measure the effect of allowing skips, choosing a better fixed order, and removing the order restriction. A positive last term is established by comparison with every full order, not inferred from a pair of visually different paths.

Independent checks. The five-method comparison covers 72 SourceGround and 1,512 constructed-loss conditions. A separately implemented receipt-history recursion checks EX at every evaluated legal history. A cursor recursion checks every fixed permutation, including skips and admissible acquisition branches not selected by the policy. Across both studies, the outputs contain 7,920 method rows, 56,340 world-policy traces, and 7,380 complete order comparisons. A clean repeat matches all non-timing results. These checks establish finite-model consistency; repetitions do not add independent tasks.

C.2 Verified Evidence at Fixed Factual Support

AuthorityWitness-1 asks whether acquiring a qualified evidence object can change action eligibility when its descriptive content is already known. The protocol uses the unmodified in-toto 3.1.0 verification API (Torres-Arias et al., 2019), independently of RightSource's state labels. It has two stages: native-verifier controls and a finite acquisition decision.

Verification context and controls. The caller fixes an owner-signed layout, its trusted public key, one authorized `attest-artifact` step and the current artifact file. Test keys are synthetic. The step requires that product and disallows other products. An inspection requires the current artifact to match the signed step product and rejects other recorded files. Each invocation starts in a fresh directory containing only the ordinary artifact file, except in the missing/extra-file controls. A fixed no-op inspection does not execute artifact content. Input metadata is stored elsewhere. The API disables inspection persistence, whereas the CLI cross-check can write an inspection record in its work directory. Concurrent modification is excluded.

Table 9 includes every registered control. The full API and CLI agree on the acceptance or rejection of all 17 inputs: 2 positive controls and 4 boundary probes are accepted; 11 negative controls are rejected by native authentication, expiry, missing-link or artifact-rule checks. No execution error is counted as a rejection. The command-mismatch probe is accepted with a warning.

Table 9: AuthorityWitness-1 native controls. These are 17 modifications of one protocol, not independent deployments. *Boundary* identifies accepted behavior outside a proposed stronger guarantee; it is not a failed negative control.

Control	Modification	Native verification result
Positive	Valid signed evidence	Accept
Positive	Same evidence via relay	Accept
Negative	Unsigned payload	Reject: link authentication
Negative	Missing required link	Reject: missing link
Negative	Unauthorized signer	Reject: link authentication
Negative	Payload altered after signing	Reject: link authentication
Negative	Current artifact changed	Reject: artifact rules
Negative	Current artifact missing	Reject: artifact rules
Negative	Extra recorded artifact	Reject: artifact rules
Negative	Untrusted layout signer	Reject: layout authentication
Negative	Layout altered after signing	Reject: layout authentication
Negative	Expired signed layout	Reject: expiry
Negative	Required step filename absent	Reject: missing link
Boundary	Different link name, validly re-signed	Accept
Boundary	Command differs from expectation	Accept; command warning
Boundary	Alternative trusted layout	Accept
Boundary	Different matching product, validly signed	Accept

The accepted relay carries the same complete proof; transport need not be a signer. A correctly re-signed link name is not required to equal the step name. A different trusted layout may accept the same proof, and a signer may attest different artifact bytes that match the current file. None of these results proves functional correctness. The default file-recording exclusions remain active, so the rules constrain recorded artifacts rather than every filesystem entry.

Acquisition instance and adapter. The registered model has one world w of probability one, $K_0 = \varnothing$, and actions `AcceptCurrentGate` and Defer. Both are safe within this bounded verification decision. The former requires r_{gate}, denoting successful verification in the fixed caller context; Defer requires nothing. The only selectable query retrieves the existing proof. Its registered outcome is deterministic. The unsigned descriptive payload is already known; a query returning it would neither shrink B nor add a confirmation and is therefore not in Adm.

For each actual query, the adapter invokes the complete upstream verifier with the caller-selected layout and trust root. Only native acceptance with the unchanged caller context adds r_{gate}. A rejection, execution error or context mismatch produces no confirmation or terminal action. Context matching is an outer binding check, not a native unique-layout guarantee; the adapter does not compare an offered link with a golden proof. In the successful replays,

$$(\{w\}, \varnothing, \{q_{\mathrm{witness}}\}) \longrightarrow (\{w\}, \{r_{\mathrm{gate}}\}, \varnothing).$$

Thus CAct changes from $\{\mathrm{Defer}\}$ to $\{\texttt{AcceptCurrentGate}, \mathrm{Defer}\}$ while B is unchanged. Cert is true at both states, and U changes. This does not establish irreducibility of K relative to (B, U).

Planning results and scope. All three methods reuse the original loss-aware planner and common gate, admissibility and stop-first tie rule. Acceptance has zero terminal loss. Primary witness cost is 1, with Defer losses $\{0.5, 1, 2\}$; the predeclared $\pm 20\%$ cost sensitivity gives the complete grid in Table 10. These costs are normative weights, not measured verification latency. Every method attains $\min(c, L_{\mathrm{defer}})$, with Defer chosen on ties.

Table 10: Complete AuthorityWitness-1 grid. Each row reports the shared result of exact, two-step, and machine-first stopping; cost 1 is the primary setting.

Witness cost	Defer loss	Decision (all three methods)	Queries	Objective
0.8	0.5	Defer	0	0.5
0.8	1	Verify, then accept	1	0.8
0.8	2	Verify, then accept	1	0.8
1	0.5	Defer	0	0.5
1	1	Defer	0	1
1	2	Verify, then accept	1	1
1.2	0.5	Defer	0	0.5
1.2	1	Defer	0	1
1.2	2	Verify, then accept	1	1.2

The 27 method rows contain 12 actual upstream calls and 15 immediate Defer decisions with no call. An independent evaluator reconstructs every policy, state transition, gate and exact objective without invoking the planner. A separate replay reproduces all scientific fields exactly. The unit is one constructed protocol instance ($n = 1$), with a deterministic world and zero between-world variance by construction. Sensitivity settings are not independent samples; there is no population estimate, confidence interval or significance test. The result supports evidence-dependent action eligibility under the declared gate, not planning superiority, dynamic human consent or end-to-end supply-chain security.

C.3 Descriptive Information for Evidence Acquisition

InformationRouting-1 is one constructed artifact-admission task. Candidates A and B have distinct artifact bytes and fixed caller-selected verification contexts. The unmodified in-toto verifier checks each acquired proof in that context. Only successful verification supplies the candidate-specific confirmation k_A or k_B. Admitting A, admitting B, and Defer have losses 0, 1, and 8 and requirements $\{k_A\}$, $\{k_B\}$, and $\varnothing$. Initial confirmations are empty. Each candidate has two complete-proof routes, costing 1 and 2; a registered missing proof is also charged. Proofs from different routes are not pooled.

Matched structures. Let x contain four route-availability bits and $z \in \{A, B\}$. Put $p_q(z) = 1/2 + \delta$ for a route matching z and $1/2 - \delta$ otherwise, and set

$$P_\delta(z, x) = \tfrac{1}{2} \prod_q p_q(z)^{x_q} (1 - p_q(z))^{1 - x_q}, \qquad M_\delta(x) = \sum_z P_\delta(z, x). \tag{38}$$

The independent control I has 16 equally likely availability vectors. C0 and C1 both use the 32-world distribution P_δ; only C1 can query z. The null N uses $P_{\text{null}}(z, x) = \frac{1}{2} M_\delta(x)$ and permits the same descriptor query. Thus N preserves the entire route joint distribution while making the descriptor independent; it does not replace correlated routes with independent marginals.

Query q_Z reads a pre-existing A/B descriptor at cost d and adds no confirmation. It changes neither file availability nor proof validity. The full descriptor is returned to the controller. Source availability and the descriptor correlation are stipulated model inputs, not guarantees of the native verifier or estimates from operational services. Only the harness holds the realized descriptor and availability vector; the controller receives a queried response after acquisition is charged. This is a cooperative access contract, not a host-level sandbox.

Primary settings are $\delta = 1/4$ and $d = 1/10$. Six sensitivity conditions vary C1's descriptor cost to $2/25$ or $3/25$, or pair C0 and C1 at $\delta = 1/5$ or $3/10$ with $d = 1/10$. These ten conditions and their interpretation rules were fixed before evaluation. They are not ten tasks.

Methods and analytic controls. The five common methods use identical contracts. BF enumerates all 24 orders in I/C0 and all 120 in C1/N, including every position of q_Z. I additionally uses the classical independent reservation-index rule (Weitzman, 1979). With reward $R_q = 8 - L_q$ for a verified proof and zero otherwise, its index z_q solves $\mathbb{E}[(X_q - z_q)^+] = c_q$, where $X_q = R_q$ with probability p_q. For these binary rewards, $z_q = R_q - c_q/p_q$ when $c_q \le p_q R_q$, and $z_q = p_q R_q - c_q$

otherwise. Search follows descending indices and stops when the best acquired reward is at least every remaining index. The contextual control always pays for q_Z first and then uses the same rule with conditional probabilities. It is compared with EX, not assumed optimal before the comparison.

Every proof success is a stopping point here: the maximum improvement after B is one, while further proof queries cost at least one and success is uncertain. Only the all-missing branch continues in C0, so its policy fits a fixed order despite correlated availability. BF and EX must therefore agree in I and C0. An independent positive-cost descriptor has no value in N. If F denotes the weighted conditional optimum after free observation of z, then the descriptor-first control costs $d + F$ and $F \le J_{\mathrm{EX}}(C1) \le \min\{J_{\mathrm{EX}}(C0), d + F\}$. These are analytic checks, not empirical discoveries.

Table 11: Primary InformationRouting objectives. Index denotes the independent control in I and the descriptor-first contextual control in C1. All objectives include acquisition cost and terminal loss.

Structure	MF	MF+skip	BF	L2	EX	Index
I	3.062500	3.062500	3.062500	3.062500	3.062500	3.062500
C0	2.714844	2.714844	2.714844	2.714844	2.714844	–
C1	2.808984	2.568750	2.537500	2.651562	2.514063	2.514063
N	2.814844	2.714844	2.714844	2.714844	2.714844	–

Table 12: All five matched descriptor conditions. Information is $J_{\mathrm{EX}}(C0) - J_{\mathrm{EX}}(C1)$, including the cost of reading the descriptor. The contextual control matches EX in every declared setting.

δ	d	$J_{\mathrm{EX}}(C0)$	$J_{\mathrm{EX}}(C1)$	Information	BF–EX	Context–EX
1/4	1/10	2.714844	2.514063	0.200781	0.023438	0.000000
1/4	2/25	2.714844	2.494063	0.220781	0.023438	0.000000
1/4	3/25	2.714844	2.534063	0.180781	0.023438	0.000000
1/5	1/10	2.833700	2.730250	0.103450	0.052650	0.000000
3/10	1/10	2.579200	2.280000	0.299200	0.008000	0.000000

Information and eligibility. At the primary setting, EX first reads q_Z. The unique optimal next query is q_{A1} after A and q_{B1} after B, each on a branch of probability $1/2$. After A, the conditional query values in order $(q_{A1}, q_{A2}, q_{B1}, q_{B2})$ are $(125/64, 173/64, 11/4, 951/256)$; after B they are $(205/64, 133/32, 23/8, 29/8)$. Immediate Defer costs 8 in both branches. The minima are strict, so the change is not an arbitrary tie. Both descriptor outcomes leave K empty and only Defer eligible. Admission becomes eligible only after the requisite proof is acquired and natively verified. In N, EX and BF never read the descriptor and both equal C0's objective $695/256$. Although reading the independent descriptor splits the full world cell and is admissible, its positive cost buys no decision value.

Table 13: Primary C1 cost decomposition and descriptor-selection probability. Components are rounded independently; exact fractions are supplied in the records.

Method	Proof cost	Descriptor cost	Terminal loss	Objective	$P(q_Z)$
MF	2.015625	0.100000	0.693359	2.808984	1.000000
MF+skip	1.500000	0.100000	0.968750	2.568750	1.000000
BF	1.531250	0.100000	0.906250	2.537500	1.000000
L2	1.875000	0.018750	0.757812	2.651562	0.187500
EX	1.562500	0.100000	0.851562	2.514063	1.000000
Contextual	1.562500	0.100000	0.851562	2.514063	1.000000

The remaining fixed-order gap. The selected primary BF order is $(q_Z, q_{B1}, q_{B2}, q_{A1}, q_{A2})$. After A it skips the B routes before trying A. If both A proofs are missing, EX can still query q_{B1}, but BF cannot revisit it. This history has mass $1/32$. The additional query costs one and, with probability $1/4$, reduces terminal loss from 8 to 1. Its conditional value is $7/4 - 1 = 3/4$; weighted by $1/32$, this

accounts for the full BF–EX gap $3/128$. EX pays more on missing-B paths and gains on verified-B paths; it does not dominate on every world. All 120 orders are compared before identifying this explanation. The contextual control matches EX, so the example does not establish a need for full-horizon search.

Execution and scope. The complete run has 56 method rows and 1,696 positive-probability world-policy traces, with 3,038 actual native proof checks and 864 descriptor reads. There are 1,519 verified and 1,519 normally missing proofs and no execution failures. Every requested proof invokes the original verifier; every descriptor response is decoded from the returned bytes before comparison with the model. Access, parser, context, and environment errors are failures, not missing responses or favorable Defer outcomes. Complete traces, losses, exact state values, order comparisons, inputs, and native records are supplied. Independent checks validate every stored receipt-history and cursor value, actual observation, eligibility transition, and acquisition record. A clean repeat matches all non-timing results and native semantics; only explicit execution times and task-independent runtime locations are normalized. The independently specified availability distribution and normative costs limit the conclusion to this controlled model.

C.4 SourceGround-8: Public-Source-Grounded Planning

Sources and task construction. SourceGround-8 contains eight software tasks, 132 registered worlds, and 21 queries. Table 14 gives their source-supported behavior and scope. The tasks and configurations were specified before method comparison. The supplied inputs include pinned source code, source locations, complete outcome maps, and action-specific requirements.

Factual rules are annotated from source code; the benchmark does not execute the full repositories. Source authority and scripted current-owner responses are specified separately, as are configurations, priors, and losses. Historical project decisions do not supply current approval.

Table 14: SourceGround-8 task scope. Linked snapshots and selected source files are preserved with their upstream licenses. Current-owner contracts are authored.

Case	Source-supported observations	Bounded task
GTasks	Invocation priority and available lists	Resolve A/B target selection under invocation precedence and current-owner agreement.
Poetry	Package mode and metadata in one configuration record	Evaluate mode/metadata build eligibility, with other packaging prerequisites fixed valid.
Positron	Workspace newline rule and opt-in setting guidance	Resolve local-setting requests and refer product-default changes outside the local contract.
PowerShell	Bound switch presence, values, and type-header rule	Determine the header form for one non-null object in a fresh invocation, or reject conflicting switches.
Prettier	Explicit all/es5/none option and syntactic position	Check trailing-comma permission for multiline function calls and object literals.
Rich	Environment and explicit Console argument precedence	Evaluate ANSI-color permission for a non-TTY stream with `TERM=xterm` and automatic color detection.
SVGO	Plugin scheduling and SVG geometry/nesting	Certify an isolated viewBox removal step from geometry, nesting, and current permission.
urllib3	Origin comparison and redirect-header configuration	Decide Cookie forwarding under the registered redirect and header-removal policy.

Objective and grid. Each case uses a uniform prior and empty initial confirmations. A query observes a complete registered source record; deterministic consequences of observed inputs are computed without another query. Defer is always permitted. The primary condition assigns machine cost 1, owner cost 4, Defer loss 8, and loss 0 to a certified concrete resolution, including rejection or owner cancellation. These weights are normative, not measured latency or deployment harm. The complete grid crosses owner cost $\{1, 4, 16\}$ and Defer loss $\{2, 8, 32\}$, keeping machine cost 1 and

uniform priors: eight primary and 64 sensitivity conditions. Every method uses the same CAct, Adm, and terminal loss.

Table 15: SourceGround-8 primary objectives and gap decomposition. All methods have zero Defer mass. A concrete resolution need not be a software modification.

Case	MF	MF+skip	BF	L2	EX	Skip	Order	Residual
GTasks	5.111	5.111	5.111	5.000	5.000	0.000	0.000	0.111
Poetry	2.333	2.333	2.333	2.333	2.333	0.000	0.000	0.000
Positron	1.889	1.889	1.889	1.889	1.889	0.000	0.000	0.000
PowerShell	3.857	3.857	3.857	3.857	3.857	0.000	0.000	0.000
Prettier	4.000	4.000	3.333	3.333	3.333	0.000	0.667	0.000
Rich	2.667	2.667	2.500	2.500	2.500	0.000	0.167	0.000
SVGO	2.167	2.167	2.000	2.000	2.000	0.000	0.167	0.000
urllib3	3.000	3.000	3.000	3.000	3.000	0.000	0.000	0.000

Results and sensitivity. Equal-case mean objectives are $1051/336$ for MF and MF+skip, $1009/336$ for BF, and $3013/1008$ for EX and L2. The primary MF–EX gap is $5/36$, split into $0 + 1/8 + 1/72$ by Equation 37. Thus fixed-order selection explains 90% of the original gap, not a 90% reduction in total cost. Mean human queries are $277/672$ for all five methods.

Across all 72 conditions, BF matches EX in 66 and is worse in six. MF+skip improves MF only for SVGO at Defer loss 2 with owner cost 4 or 16, by $1/6$ in each condition. L2 matches EX throughout. Full action distributions and all zero gaps are retained in the result data.

Table 16: Every strict BF–EX gap in SourceGround-8. The other 66 conditions have zero gap; only GTasks at owner cost 4 and Defer loss 8 is a primary condition.

Case	Owner cost	Defer loss	BF–EX	Setting
GTasks	16	32	1/9	Sensitivity
GTasks	4	32	1/9	Sensitivity
GTasks	4	8	1/9	Primary
Rich	1	2	1/24	Sensitivity
Rich	1	32	1/24	Sensitivity
Rich	1	8	1/24	Sensitivity

Path explanation. In GTasks, write I for inspection of selection parameters, L for listing available targets, and A for the owner's answer. EX can follow I, A, L after no flag and no configured target, but I, L, A under flag/configuration conflict. The selected BF order is I, L, A. In the primary condition, EX saves one list query in three cancellation worlds of mass $1/27$ each, with all other path costs unchanged; this accounts for the full $1/9$ difference. In Rich, environment inspection can lead EX to ask before checking the Console argument or to check it first. Its $1/24$ gap is a prior-weighted result with both cheaper and more expensive EX paths, not pointwise dominance. The artifact retains all six strict conditions and their complete paths; the first lexical condition in each task supplies the sensitivity witness.

Validation and scope. The per-world and all-order checks in Appendix C.1 pass throughout SourceGround-8. Identifier-free typed-graph invariants distinguish the eight registered structures. Small query sets, authored contracts and selected configurations limit the conclusions to this benchmark. The cases are not a random repository sample; sensitivity settings are not additional repositories.

C.5 Multi-Action Planning with Terminal Loss

This finite study evaluates Equation 8 when repair, reject, and defer are available. Repair requires a passing diagnosis and the relevant approvals; reject requires evidence of a failed diagnosis; defer is always safe and requires no confirmation. Three contexts use one test and one approval, one test and

two approvals, or two tests and one approval. Repairable and rejectable worlds both have positive prior mass, as do approval and refusal outcomes.

Declared grid and common semantics. The repair prior and a common approval probability each take values in $\{0.25, 0.75\}$; approval outcomes are independent of the diagnoses and of each other, with the same probability for both owners when present. In the two-test context, both diagnoses pass with the repair prior p; each of the other three joint outcomes has probability $(1-p)/3$. Human query cost ranges over $\{0.05, 0.1, 0.25, 0.5, 1, 2\}$, and defer loss over $\{0.05, 0.1, 0.25, 0.5, 1, 2, 4\}$. Machine cost is 0.25; repair and reject losses are 0.05 and 0.02. The 504 primary conditions are supplemented by 1,008 conditions at 0.8 and 1.2 times the machine cost, with no retuning. All five methods defined in Appendix C.1 share CAct, Adm, outcome maps, priors, and losses. L2 uses the current certified stop loss at the two-step horizon boundary and replans after each outcome. These are not HEC implementations: with defer already certified, the original AA-HEC-F obstruction gain is zero and does not supply a loss-aware objective.

Table 17: Mean expected objective over all 504 primary constructed conditions. Objective is query cost plus terminal loss. Ties use exact rational values; displayed components are rounded independently. Regret is objective minus the exact optimum.

Method	Objective	Query cost	Terminal loss	Regret	Ties
MF	0.628698	0.173378	0.455321	0.000618	490/504
MF+skip	0.628698	0.173378	0.455321	0.000618	490/504
BF	0.628080	0.173144	0.454936	0.000000	504/504
L2	0.631340	0.163329	0.468011	0.003260	491/504
EX	0.628080	0.173144	0.454936	0.000000	504/504

BF matches EX on all 504 primary and 1,008 sensitivity conditions. MF+skip matches MF throughout. In the primary grid, L2 has higher objective than EX in 13 conditions and ties in 491; MF and MF+skip are worse in 14 and tie in 490. Their mean regrets are 0.003260 and 0.000618, with maxima 0.330156 and 0.074792, respectively. Across the full 1,512 conditions, L2 is worse in 30 and MF in 41; BF reaches the optimum even in these conditions. This identifies a limitation of the two-step heuristic without establishing a need for outcome-dependent reordering.

EX starts with a test in 218 primary conditions, asks in 8, and defers in 278. Its mean number of queries is 0.6753, compared with 0.6230 for L2. BF and EX also ask more human queries on average than MF. Lower objective therefore need not mean fewer queries. At machine-cost multipliers 0.8 and 1.2, mean regrets are 0.002436 and 0.002170 for L2, and 0.000418 and 0.000949 for MF and MF+skip; BF's regret remains zero.

An interpretable decision transition. For one test and one approval, with both priors 0.75, machine cost 0.25, and human cost 0.05, defer losses 0.1, 0.5, and 2 select Defer, Ask, and Test at the root, with expected objectives 0.100000, 0.394375, and 0.695625. The study outputs include the full phase map.

Validation and computational scope. The per-world and all-order checks in Appendix C.1 pass throughout the grid. BF's root-time enumeration is not a cheap model-free heuristic, and these small problems do not establish a scaling advantage for any method.

This complete deterministic grid of constructed inputs is separate from SourceGround-8 (Appendix C.4) and is not a repository sample. The protocol was recorded before the sweep. Complete inputs, outcomes, losses, and per-world traces are included in the artifact.

D Authority and Execution Checks

D.1 Scope and Confirmation Controls

Consider widening legal source scopes while holding every outcome confirmation set $K_q(o)$ fixed. Contract legality is preserved, and every reachable (B, K, U), admissible query set, stopping deci-

sion, and Bellman value is unchanged. Deterministic tie breaking therefore returns the same policy. Altered-contract results in Appendix D.2 cannot be attributed to scope metadata alone; an acquisition change requires a change to the confirmations an outcome can supply.

D.1.1 FIELD-LEVEL CONTRACT CONTROLS

To make the intervention explicit, we construct a separate one-world problem with a repair action requiring owner approval and an always-safe defer action. A human confirmation costs 1; a machine record costs 0.1. Repair and defer have terminal losses 0.05 and 2. We retain candidate confirmations $P_q(o)$ as an additional input field and initially ground $K_q(o) = P_q(o) \cap \mathrm{Auth}(q)$. This grounding rule defines the constructed intervention.

In the scope control, the machine record contains an owner-approval candidate but lacks the authority to confer it. Widening its scope while holding $K_q(o)$ fixed leaves the first query human and the value 1.05. Widening the scope *and re-grounding* $K_q(o)$ instead produces a machine-first policy of value 0.15. In the location control, the machine source has the relevant scope, but its stored policy initially supplies no current approval. Adding the approval candidate to $P_q(o)$ and re-grounding $K_q(o)$, with scope fixed, also changes the first query from human to machine and the value from 1.05 to 0.15. Worlds, actions, requirements, priors, costs, and outcome maps are identical within each pair.

Every policy is valid under its supplied contract. Both confirmation-changing policies fail when replayed under the original contract; the scope-only policy passes. Thus changing metadata alone and changing the meaning of an observation are different interventions. These are constructed semantic controls, not runtime attacks or reconstructed historical cases. Complete inputs, intervention fields, and traces are included in the artifact.

D.2 MATCHED ALTERED-CONTRACT AUDITS

The retained targeted suite contains 48 matched cases and evaluates 144 policies under a frozen full-semantics oracle. It includes the earlier two-world controls and two four-world, non-uniform-prior adaptive-stopping families. `scope_adaptive` makes a cheap ticket sufficient only when its source has wide authority; `locus_adaptive` allows a policy location to expose a default value without constituting a current commitment. Ablated policies are always judged under the frozen full semantics rather than under their own weakened contracts.

Table 18: Matched altered-contract policy audits. "Fails audit" means the compiled policy fails the frozen full-semantics audit, not that runtime action exposure has already blocked it. FA mass is the unauthorized-action probability mass of that policy under full semantics.

Condition	N	Changed	Fails audit	FA mass
Full semantics	48	0/48	0/48	0.00
Narrow scope + Authority-All, adaptive	6	6/6	6/6	0.70
Wide-scope control + Authority-All	6	0/6	0/6	0.00
Absent commitment + Locus=Present, adaptive	6	6/6	6/6	0.50
Present-commitment control + Locus=Present	6	0/6	0/6	0.00

The full-semantics policies pass in all 48 cases. The recorded `authority_all` and `locus_present` comparisons evaluate altered effective contracts and do not isolate changes to $\mathrm{Auth}(q)$ from changes to $K_q(o)$. In particular, the changed `authority_all` policies cannot result from legal scope widening with fixed confirmations, by the invariance in Appendix D.1. The audit identifies failures under altered contracts, without attributing them to a single field. The matched wide-scope and present-commitment controls remain valid under the recorded audit.

D.3 OBSERVATION-MAP INTEGRITY

Threat model. Artifact binding establishes which input was read; it does not establish that an observation was mapped to the correct registered public outcome. Before execution, we froze 11 sources for which typed counterfactual observations cover all 30 registered root outcomes. Each variant was run with the correct full map, a map omitting the current value, and a deterministic

within-source label permutation. Deno and OpenTelemetry were excluded before observing results because their available projections do not identify every registered outcome.

For source s, let $C_s = \{(x_i, y_i)\}_{i=1}^{n_s}$ be a labeled fixture suite, let $X_s = \{x_i\}$, and let Y_s be the registered outcome set. A candidate observation map m_s passes the fixture-conformance check exactly when

$$\begin{aligned} \mathrm{Conf}(m_s; C_s) = 1 \iff & \ \mathrm{dom}(m_s) = X_s, \\ & \forall (x_i, y_i) \in C_s : \ m_s(x_i) = y_i, \\ & \{m_s(x) : x \in X_s\} = Y_s. \end{aligned} \tag{39}$$

The offline fixture-conformance replay defines the exposure interface

$$\mathrm{Expose}(m_s, x) = \begin{cases} \mathrm{Compile}(m_s(x)), & \mathrm{Conf}(m_s; C_s) = 1 \ \wedge \ x \in X_s, \\ \bot, & \text{otherwise}, \end{cases} \tag{40}$$

so a rejected registry map cannot expose an autonomous action through that interface. Algorithm A3 implements this conformance condition as a signed activation step and is tested end to end on valid, permuted, and incomplete bindings. The certificate binds the map, fixtures, outcome ontology, extractor implementation, and query contract before compilation.

Table 19: Ungated mapping challenge and fixture-conformance replay. "Exposed" means the evaluation interface returned an action; "blocked" means it returned unresolved before action exposure.

Arm	Units	Action exposed / blocked	False autonomy	Benign
Correct full map	30 variants	30 exposed	0	0
Missing current value	30 variants	30 blocked	0	0
Permuted map, ungated path	30 variants	30 exposed	28	2
Permuted map, fixture-conformance replay	30 variants	30 blocked	0	0

All correct labels, certificates, designated registered-world checks, and reference traces pass in 30/30 variants. Missing current values fail closed in 30/30. In contrast, the permuted map passes the ungated archived path in every variant and returns unauthorized actions in 28/30 variants; the remaining two are Cloudstic policy equivalences. At the mapping level, Equation 39 accepts 11/11 correct maps, rejects 11/11 permuted maps, and rejects all 30 single-value omissions. In the fixture-conformance replay, gating changes corrupted false autonomy from 28/30 to 0/30 while preserving 30/30 correct-map safety.

These offline rates are relative to frozen labels. Checking a map is linear in the fixture suite size $|C_s|$. In this study each of the 11 suites contains 2–4 fixtures and jointly covers 30 root outcomes, but keeping suite size proportional to registered outcomes is an authored assumption. Raw-observation diversity, annotation time, and refresh cost are unmeasured; annotation remains inside the trusted boundary.

D.3.1 Reference Implementation Validation

The packaged reference implementation passes 55 tests. The suite covers signed activation, action-indexed compilation, checked exposure, and the finite prior–cost stress runner.

Our same-terminal AA-HEC-F obstruction-cut comparator. At state (B, K, U), let the inclusion-minimal authority obstructions be

$$\mathcal{E}(B, K) := \{E \subseteq B : E \neq \varnothing, \ \mathrm{CAct}(E, K) = \varnothing, \ \forall \varnothing \neq E' \subsetneq E, \ \mathrm{CAct}(E', K) \neq \varnothing\}. \tag{41}$$

For each $q \in \mathrm{Adm}(B, K, U)$, AA-HEC-F assigns the removed-obstruction gain

$$G(q; B, K) := \sum_{E \in \mathcal{E}(B,K)} \left(\prod_{w \in E} p(w \mid B) \right) \mathbf{1}[\forall o : \ E \not\subseteq B_{q,o} \ \vee \ \mathrm{CAct}(E, K \cup K_q(o)) \neq \varnothing]. \tag{42}$$

Among admissible queries with feasible children, AA-HEC-F maximizes $G(q; B, K)/c(q)$; positive-gain zero-cost queries take priority, and ties are deterministic. This custom obstruction-cut score is not the published augmented-space HEC objective and inherits no HEC approximation guarantee.

Table 20: Reference implementation checks and their scientific scope.

Component	Executed check	Boundary
Action-indexed compiler	15 finite specification cases for $R(w, a)$, CAct, RelReq, reachable branches, memoization, and ties	Pair-SAT scaling uses a singleton-action signature
Signed-activation path	Correct-map action exposure; permutation blocked before compiler construction; missing binding and unseen value unresolved	Fixture coverage does not establish the semantics of unseen raw observations
AA-HEC-F	Shares $\mathrm{CAct} \neq \varnothing$, Adm, and an exact feasibility shield on six finite instances	Custom obstruction-cut score, not published augmented HEC; no approximation guarantee
Prior/cost runner	Five prior families by seven human/machine cost ratios; 210 condition cells and 420 method rows	Six constructed problems; repeated conditions, no statistical or generalization claim

Every resolved row passes exact per-world replay. The exact solver's 55/155/0 record against AA-HEC-F is enforced as a regression invariant and is reported as implementation validation, not as a population estimate. Trusted deployment credentials, execution freshness, and raw machine observation authentication remain in the trusted computing base.

D.4 Tool-Backed Path Integrity

LiveProbe replaced in-memory outcomes with filesystem- and tool-backed receipts from static, build/test, solver, registry, and sealed human-answer probes. Accepted receipts were cached; only public typed outcomes reached the verifier and compiler. The 12 executions crossed three domain templates (repository release, access control, incident recovery) with four shared configurations. They are path-integrity tests, not 12 independent deployments.

All 12 runs were safe, exact, and certified. They produced 9,216 receipts, 164 verifier calls, 4,559 states, and 70 counterexample rounds. Summed uncached probe time was 678.76 s; cached full replay took 12.29 s. Cold and warm deterministic wall times were 422.24 s and 12.94 s, respectively; these timers measure different operations.

A separate one-task-per-domain integrity subset rejected all three world-tree identity changes despite preloaded cached receipts. All three solver-file mutations changed the receipt and the counterexample or adequacy decision. This three-run subset does not meet the main suite's 12-execution gate. Online probe selection and unscripted human behavior were not tested.

D.5 Abnormal Source-Response Recovery Matrix

The formal matrix contains six response classes, tool and human sources, and four matched arms per class/source pair: a normal control, an abnormal primary source without fallback, an abnormal primary source with a distinct authority-legal fallback, and the same fallback blocked by the hard query budget. This yields 48 rows in 12 matched sets. Receipt admission checks source identity, query-contract identity, chain identity, outcome vocabulary, deadline, contradiction, and revocation before any outcome is exposed to the compiler.

Table 21: Fallback-available abnormal arms. Counts aggregate tool and human sources.

Primary response	Recovered	Terminal	Terminal reason
Refusal	2/2	0/2	–
Timeout	2/2	0/2	–
Unavailable	2/2	0/2	–
Contradiction	0/2	2/2	conflicting active receipts
Revocation	0/2	2/2	no active commitment
Malformed/adversarial	0/2	2/2	receipt integrity failure

Refusal, timeout, and unavailability recover through a fresh authorized fallback in 6/6 tool/human arms. Contradiction, revocation, and malformed evidence remain fail-closed in 6/6 arms. The 12 no-fallback arms and 12 budget-blocked arms also terminate before a second query. Across all 48

rows, 18 expose certified actions, 30 terminate intentionally, with zero false autonomy across all 48 rows. This establishes deterministic recovery and budget enforcement, not robustness to colluding or strategically deceptive participants.

D.6 Post-hoc Receipt-Conditioned Replay

Protocol and scope. This is a deterministic post-hoc replay over frozen repository-history artifacts, not a preregistered endpoint and not an online deployment. The language model does not choose the probe. For each case, the full registry compiler first selects its minimum-cost root query; a machine receipt is executed only when its registered query is that root. The receipt projection must be derived from an H0/H1-invariant upstream observation and is rejected if it uses an expected answer, action/world label, or revision-dependent post-treatment field. No live human query is executed.

The 14 accepted rows are historical finite replay records, separate from SourceGround-8. Shared repository names do not imply identical registries or a conditioning relation between the studies.

The post-hoc semantic audit retains 14 machine receipts, rejects four machine roots, and records one human-first defer and one terminal policy. Three rejected observations omit facts required by their registered outcomes: stream validity for Deno, explicit-setting provenance for AWS, and provider enablement for OpenTelemetry. Crossplane is rejected because its cluster-usage observation is revision-dependent post-treatment state. The rejected roots are not replaced by downstream observations. Each accepted receipt captures only facts within its source contract.

Receipt validation and residual construction. Receipts must pass Algorithm A1's validation checks and map through the frozen public outcome map to a reachable registered outcome of the deterministic full-registry root query. Only then is $B_{q,o}$ constructed from Ω.

Quotient exactness and two distinct regrets. This finite replay is separate from the large-support scaling procedure. Let $S \subseteq U_0 \setminus \{q\}$ be the queries retained after Pair-SAT refinement and exact deletion. Both π_F and π_S start with $B_{q,o}$ and $K_0 \cup K_q(o)$. They use $U_0 \setminus \{q\}$ and S, respectively; neither can repeat the observed query.

For $w, w' \in B_{q,o}$, define $w \sim_S w'$ iff $g_{q'}(w) = g_{q'}(w')$ for every $q' \in S$. Each cell C must have one implementation settlement signature: required action, required requirements, and registered-safe actions. Any mixed-signature cell invalidates the quotient. Otherwise, π_Q is compiled on these cells with masses $p_Q(C) = \sum_{w \in C} p(w \mid B_{q,o})$.

For residual world w, let $\tau_F(w)$ and $\tau_S(w)$ be the complete execution traces, and let $\tau_Q([w]_S)$ be the quotient trace lifted back to w. A trace contains query identifiers and kinds, outcomes, confirmation snapshots, selected action, authority-completeness status, and exact rational path cost. World abstraction is exact when

$$\tau_Q([w]_S) = \tau_S(w) \quad \forall w \in B_{q,o}, \qquad V_Q = V_S, \tag{43}$$

whereas retained-query pruning is exact when

$$\tau_S(w) = \tau_F(w) \quad \forall w \in B_{q,o}, \qquad V_S = V_F. \tag{44}$$

Here V_F, V_S, V_Q are exact remaining expected query costs under the renormalized receipt-conditioned prior; the already-paid receipt cost is common and cancels. The reported decomposition is

$$\begin{aligned} \mathrm{Reg}_{\mathrm{abs}} &:= V_Q - V_S, \\ \mathrm{Reg}_{\mathrm{prune}} &:= V_S - V_F, \\ \mathrm{Reg}_{\mathrm{total}} &= \mathrm{Reg}_{\mathrm{abs}} + \mathrm{Reg}_{\mathrm{prune}} = V_Q - V_F. \end{aligned} \tag{45}$$

Thus quotient exactness is a semantic lifting claim relative to $B_{q,o}$, while strict world-count compression is a separate empirical event. Exactness does not imply that a case must compress.

The extended-results supplement lists all 14 case-level records.

Results. Support shrinks in 14/14 executed cases, with median reduction 56.0%. The first Pair-SAT counterexample changes or disappears in 13/14, and refinement work improves in every case. All 14 conditioned quotients recompile exactly with zero abstraction, pruning, and realized cost regret; their action leaves pass registered-world safety and authority-completeness checks. Four cases reduce quotient-world count, with median compression 0%, so these data do not estimate repository-wide compression frequency.

E End-to-End Evaluation Details

E.1 Original system comparisons

Table 22: Original Retail system comparison over 345 trajectories (115 tasks, three repeats), separate from the shared-gate ablation in Table 3. Values are means ± task-level SEM. Queries count dispatched external tool calls, excluding user dialogue; they are not the acquisition-interaction measure I of the shared-gate ablation. Success is a proportion; costs are estimated USD per task under reference pricing. Amortized cost assumes policy reuse $M = 50$; it is calculated, not measured over repeated deployment.

Method	Success ↑	Queries ↓	Cold cost ↓	Amortized cost ($M = 50$) ↓
Standard Tool-Agent	0.704 ± 0.025	4.82 ± 0.17	0.058 ± 0.002	0.058 ± 0.002
Machine-first	0.719 ± 0.024	4.41 ± 0.13	0.055 ± 0.002	0.055 ± 0.002
Dev-fixed	0.713 ± 0.024	4.25 ± 0.12	0.053 ± 0.002	0.053 ± 0.002
RightSource	0.716 ± 0.024	4.16 ± 0.11	0.054 ± 0.002	0.049 ± 0.002

Tasks and aggregation. The base tool-using agent is `Qwen/Qwen3.6-35B-A3B`. Retail uses 115 tasks, each evaluated with seeds 42, 1337, and 2026, for 345 trajectories per method. AgentDojo covers all 97 user tasks across Workspace (40), Banking (16), Slack (21), and Travel (20). We use an experiment-defined fixed mapping from each user task to one compatible injection task in the same suite, with an objective conflicting with the legitimate request. Each pair is evaluated under three random seeds. We use AgentDojo's `important_instructions` fixed-template attack in benchmark-defined, reachable untrusted tool-response fields. Pairings and payloads are shared across all methods, without method-specific attack tuning. This covers the user-task set, not the full 629-case security matrix of the original benchmark. Each method has 291 clean and 291 attacked trajectories (582 in total). Clean and attacked utility are binary task-success rates in their respective conditions; ASR and argument-scope violations use attacked trajectories. In Travel, searches gather information while reservations change external state; untrusted search results do not themselves supply confirmation credentials.

For a metric y_{ir} on task i and repeat r, let

$$\bar{y}_i = \frac{1}{3}\sum_{r=1}^{3} y_{ir}, \qquad \bar{y} = \frac{1}{N}\sum_{i=1}^{N} \bar{y}_i, \qquad \mathrm{SEM}(\bar{y}) = \frac{s(\bar{y}_1, \ldots, \bar{y}_N)}{\sqrt{N}}, \tag{46}$$

where s is the sample standard deviation and N is the number of tasks, not the number of trajectories. Equal repeat counts make the task-averaged mean identical to the pooled trajectory mean. SEM is not a confidence interval. Original Retail success counts are 243, 248, 246, and 247 for Standard Tool-Agent, Machine-first, Dev-fixed, and RightSource, respectively. Table 23 gives the AgentDojo counts underlying Table 4.

Comparators and outcomes. The retail comparison includes a standard tool agent, Machine-first, Dev-fixed, and RightSource. Dev-fixed is the fixed-order comparator reported for retail; it is distinct from the exhaustive best-fixed oracle in the finite studies. AgentDojo includes an unprotected agent, Prompt-Guard-86M with threshold 0.85, a rule-based tool filter, and RightSource. The rule-based filter uses a static regular-expression deny list (version 2.1) for write paths, external IP addresses, and high-risk tool arguments. These system-level comparisons do not isolate the effect of adaptive query ordering.

Table 23: AgentDojo outcome counts. Each column has denominator 291 per method; clean utility uses clean trajectories and the other columns use attacked trajectories. Attack success and argument-scope violations need not coincide.

Method	Attack success	Clean task success	Attacked task success	Argument-scope violation
Unprotected Agent	158	211	76	165
Prompt Guard / Filter	71	206	120	76
Tool Filter (rule-based)	48	196	141	53
RightSource	24	205	164	11

Retail success and AgentDojo utility concern the legitimate task; ASR concerns the attacker's objective. AgentDojo attack success is scored by the corresponding official injection-task evaluator, using model output, environment changes, or execution traces as required by that task. Task failure or deferral alone does not imply attack success. For every method, $M^{\mathrm{arg}}_{\mathrm{unauth}}$ is the fraction of attacked trajectories containing at least one dispatched and executed tool call whose arguments exceed the user's authorization scope. Proposals blocked before dispatch do not count. This trajectory-level measure differs from the finite-world terminal mass in Equation 4. Binary rate differences are computed from the integer counts before rounding and are expressed in percentage points. Original system-level cost reductions use the displayed means, $100(\bar{x}_b - \bar{x}_{\mathrm{RS}})/\bar{x}_b$.

Costs and authorization. Retail costs estimate agent-side USD expenditure using assumed reference rates of \$0.40 per million input tokens and \$1.20 per million output tokens. The user simulator is excluded for all methods; SourceGround query weights are not USD prices. Amortization assumes reuse over $M = 50$ requests, rather than measuring 50 successive deployments. For RightSource, the estimated task and per-trajectory checking costs are \$0.0481 and \$0.0008; retries are included in task cost. An assumed \$0.0050 policy-construction fee gives \$0.0539 before amortization and \$0.0490 after amortization. This fee is not a measured CPU-time charge. Treating it as constant leaves task-level SEM unchanged by amortization. The single success column does not compare separate cold and reused-policy runs.

The finite guarantee is conditional on registered action semantics and the checked execution path. In this agent integration, certification checks action categories, not the semantic scope of open-ended arguments. The observed 11/291 argument-scope violations therefore concern a boundary not certified by the category-level controller.

Reporting scope. The original system tables report descriptive means and task-level SEM; no paired non-inferiority test is reported for these comparisons. Cost comparisons are conditional on the stated pricing and reuse assumptions. Attack successes and argument-scope violations are separate outcomes; their marginal counts do not establish an overlap or a causal failure breakdown.

E.2 Retail planning under a shared gate

This separate comparison evaluates Dev-fixed, L2, and EX on the same 115 Retail tasks with three seeds per task (345 trajectories per planner; 1,035 total). The base model, registered task models, argument checks, and authorization gate are shared; the comparison changes the planning strategy. Dev-fixed retains a fixed relative query order with skipping and stopping, L2 uses two-step lookahead, and EX uses full-horizon planning. The counts below belong to this comparison, not Table 22.

Counts and uncertainty. Acquisition interactions are $I = Q_{\mathrm{obs}} + H_{\mathrm{confirm}} + H_{\mathrm{info}}$: read-only tool observations, confirmation requests, and factual follow-up questions. A single dual-purpose user request is counted once. Total tool calls Q_{tool} also include state-changing calls; user questions are not tool calls. Equation 46 defines the task-level SEM in Table 3. Defer rates are 14.8%, 14.2%, and 13.9% for Dev-fixed, L2, and EX. Argument-scope violation rates are $3.8 \pm 1.1\%$, $3.8 \pm 1.1\%$, and $3.5 \pm 1.0\%$, respectively (SEM in percentage points). These rates count trajectories with an executed out-of-scope call, not blocked proposals.

Table 24: Shared-gate Retail counts. Success, Defer, and argument violations are trajectory counts out of 345 per planner. $\sum I$ and $\sum Q_{\text{tool}}$ are event totals; dividing by 345 gives their trajectory means.

Planner	Success	$\sum I$	$\sum Q_{\text{tool}}$	Defer	Arg. violations
Dev-fixed	247	1,454	1,673	51	13
L2	248	1,431	1,648	49	13
EX	249	1,426	1,642	48	12

Paired comparisons. We average the three seeds within each task, then form paired method differences. Task-cluster bootstrap resampling keeps all seeds and methods together (10,000 resamples, seed 41003). EX minus Dev-fixed is $-28/345 = -0.081159\ldots$ interactions per trajectory, with a two-sided 95% interval of $[-0.150, -0.012]$. The success difference is $2/345$, or 0.58 percentage points, with a one-sided 95% lower bound of -1.78 points. This exceeds the prespecified -3-point margin; it does not establish equal or superior success rates. For EX minus L2, the interaction difference is $-5/345 = -0.014493\ldots$ (95% CI $[-0.061, 0.032]$), and the success difference is 0.29 percentage points (95% CI $[-1.62, 2.18]$ points). These exploratory intervals leave the difference unresolved, rather than establishing equivalence.

Table 25: Outcome-stratified accounting of EX minus Dev-fixed interactions. The signed sums add to -28 across all 345 pairs. Groups are defined by observed completion outcomes; the decomposition is descriptive, not a causal failure analysis or a replacement for the all-task comparison.

Paired outcome	Pairs	$\sum(I_{\text{EX}} - I_{\text{fixed}})$	Mean difference
Both succeed	234	-24	-0.1026
Only EX succeeds	15	$+5$	$+0.3333$
Only Dev-fixed succeeds	13	-6	-0.4615
Neither succeeds	83	-3	-0.0361

Both-success pairs contribute $24/28 = 85.7\%$ of the net interaction reduction. There are also 13 pairs where Dev-fixed succeeds and EX fails; their six fewer interactions cannot be interpreted as equally successful work at lower cost. Completion outcomes alone do not identify why either planner stopped.

Computation and reference-priced costs. Planning time has mean/p95 values 0.12/0.35, 1.85/4.60, and 3.40/8.90 seconds for Dev-fixed, L2, and EX. End-to-end latency is 18.4/42.0, 19.8/45.2, and 20.9/48.5 seconds. Relative to Dev-fixed, EX increases mean latency by 13.6% and p95 by 15.5%.

Cold-start costs use the same reference token prices as the system comparison ($0.40/$1.20 per million input/output tokens), plus a reference compute rate of $0.17 per hour. The user simulator is excluded. Mean input/output usage is 52,400/17,200 tokens for Dev-fixed and 51,600/16,850 for EX. Pricing their respective 0.12 and 3.40 seconds of computation at that hourly rate gives

$$\widehat{C}_{\text{fixed}} = 0.041600 + 0.12(0.17/3600) = \$0.04160567,$$
$$\widehat{C}_{\text{EX}} = 0.040860 + 3.40(0.17/3600) = \$0.04102056.$$

The difference is approximately $0.000585 per trajectory (1.41%, computed before rounding). These are reference-price estimates for model usage and the reported computation, not actual expenditure or measurements of policy reuse. The cost comparison is descriptive and covers Dev-fixed and EX only. The reduction in acquisition interactions therefore comes with a small estimated monetary saving and higher latency.

F Optional Implicit-Support Studies

These retained studies examine support construction, proposal ordering, and representation under a shared exact certifier and authority compiler. They are separate from the finite planning results. The

extended-results supplement contains the full per-task matrices, generator records, and diagnostic plots. Solver-state reductions below refer to the instrumented verifier counter; explicit-search states count different operations, and provider time must be included when assessing end-to-end speed.

F.1 Support access and semantic cardinality

The multi-axis study has 10 cases and 40 method rows. Full enumeration, deterministic and random refinement, and Exact-DEL-SAT use the same certifier and compiler. All completed policies are exact and authority complete. Enumeration is censored at the frozen 8,192-world construction cap before producing a policy; these rows are not unsafe policies.

Table 26: Nuisance-support crossover with five fixed authority roles and six actions. Enumeration cells give materialized worlds/search states; lazy cells give materialized worlds/verifier calls/verifier states. DEL denotes Exact-DEL-SAT.

Support	Enumeration	Deterministic	Random	DEL
32	32/64	43/12/1,704	45/12/1,464	41/6/843
512	512/1,039	54/14/2,212	57/20/3,130	42/14/2,697
8,192	8,192/16,639	66/26/4,797	63/24/3,974	42/22/4,721
32,768	8,192/16,656 cap	70/30/5,309	72/32/4,546	42/26/5,451

Explicit construction is preferable at the smallest support. At 32,768 worlds, all three lazy methods return the same certified five-query policy at total query cost nine while enumeration reaches the cap. The other axes vary fixed-area grid shape (2×12, 3×8, 4×6) or constraints controlled per role (1, 2, 4). Neither shows a monotone difficulty trend, and neither varies semantic cardinality. Exact-DEL-SAT is the strongest same-oracle deterministic control; a white-box ROBDD that receives the support CNF remains faster on the earlier low-width product family. These results support reduced materialization under the stated access model, not a general runtime advantage or a scaling law based on support size alone.

F.1.1 Fixed-support cardinality

A separate generator fixes support at $2^{10} = 1{,}024$ worlds and varies roles $n_{\mathrm{role}} \in \{3, 4, 5, 6\}$, requirements $n_{\mathrm{req}} \in \{2, 4, 6, 8, 10\}$, or actions $n_{\mathrm{act}} \in \{2, 4, 6, 8\}$, holding the other quantities fixed and using seeds $\{3, 7, 11\}$. Its signature is $\sigma(w) = (a(w), M_{\mathrm{req}}(w))$, so requirements affect the Pair-SAT disagreement predicate. Every action and requirement is reachable, every requirement has an authorized outcome witness, and autonomous actions have explicit guards. The 64-assignment compiler bound excludes $n_{\mathrm{role}} = 7$. All 195 rows across 39 cases are safe, exact, authority complete, budget compliant, and compiler certified.

Uniform-GAC uses the least construction work at this support; Exact-DEL-SAT is generally the strongest lazy method. Its median verifier states rise from 2,199 to 13,335 ($6.06\times$) over roles $3 \to 6$, from 4,248 to 8,291 ($1.95\times$) over requirements $2 \to 10$, and from 6,060 to 8,529 ($1.41\times$) over actions $2 \to 8$. Requirement and action interiors are non-monotone. Given the complete public decision signature and authority contract, Qwen completes all nine maximum-cardinality cells with zero invalid proposals. It beats deterministic refinement in 8/9 and random refinement in 7/9, but the strongest same-case lazy method in only 1/9. No support, threshold, seed, or baseline was changed after this result.

F.1.2 Preregistered cardinality–support interaction

The interaction fixes $(n_{\mathrm{role}}, n_{\mathrm{req}}, n_{\mathrm{act}}) = (6, 10, 8)$ and varies support $2^{10}, 2^{13}, 2^{15}, 2^{17}$ over seeds $\{3, 7, 11\}$. The paired-XOR family, one human-cost role, five machine-cost roles, 64-assignment compiler bound, construction cap, model packet, and outcome gates were frozen. All seven methods share the task, exact oracle, minimizer, certifier, compiler, budget, and cost rule.

Of 84 rows, 78 pass and six enumeration rows are construction-censored; every lazy row is available and safe. Uniform-GAC is best at the two smaller supports. Across the six larger-support pairs, Qwen records 6/0/0 wins/losses/ties against the strongest same-case non-model lazy method: median total states are 22,210 versus 29,739.5, median paired reduction is 22.21%, and the two-sided

Table 27: Cardinality boundaries. (a) Median verifier states at maximum cardinality and fixed 1,024-world support. (b) Median total exact-solver states over three seeds; "best lazy" selects the strongest same-case non-model method. Enumeration and verifier counters measure different operations.

(a) Fixed support

Maximum cell	Deterministic	Random	DEL	Qwen
$n_{\mathrm{role}} = 6$	23,925	20,667	13,335	19,929
$n_{\mathrm{req}} = 10$	10,785	10,207	8,291	7,828
$n_{\mathrm{act}} = 8$	13,361	13,871	8,529	14,244

(b) Support interaction at $(6, 10, 8)$

Support	UNIFORM-GAC	Qwen	Best lazy	Preferred regime
2^{10}	2,717	21,215	17,905	explicit
2^{13}	17,360	20,750	24,494	explicit
2^{15}	censored	20,902	28,921	Qwen-guided lazy
2^{17}	censored	25,005	30,426	Qwen-guided lazy

exact sign test gives $p = 0.03125$. All 12 Qwen rows pass exactness, authority, budget, and compiler checks, with zero invalid proposals or provider failures. The intention-to-treat account is 106 calls, 154,559 tokens, and 192.08 seconds of proposer wall time. Provider time is larger despite the reduced verifier-state count.

F.2 CROSS-MODEL CONFIRMATION AND EARLIER PILOTS

The frozen confirmation suite crosses three generators and two support scales with four disjoint seeds per stratum (24 tasks). Cardinality tasks use six roles, ten requirements, eight actions, and supports 32,768 and 131,072. Coloring supports are 30,901–31,986 and 215,373–218,453; role-coupled supports are 34,706–41,270 and 207,841–225,975. These latter generators retain the five-role Locus–Present schema: settlement locus, present commitment, API compatibility, failure reproduction, and rollback safety. Authority schema is assigned by generator, not crossed independently with it.

For task i, B_i is the minimum total exact-solver states among deterministic refinement, the median of random seeds $\{17, 41, 137\}$, Exact-DEL-SAT, PSEC-1, and PSEC-16; the effect is $r_i = (B_i - M_i)/B_i$. The task is the statistical unit. For 10,000 bootstrap draws, four tasks are resampled within each of the six strata and their 24 effects pooled to obtain a median. Holm–Bonferroni correction covers only the two frozen Qwen/GPT exact sign tests. Provider failure, timeout, invalid JSON, and unsafe or uncertified output count as losses with reduction -1. These tasks are not IID deployment samples.

Table 28: Model ordering on the frozen 24-task suite. Reductions measure exact-verifier states. Primary rows use 24 assigned tasks; replication rows use 24 retained tasks.

Proposer	Protocol	Win/loss	Median reduction	95% interval
Qwen36-35B-A3B	Primary	21/3	27.3%	[22.3, 30.3]%
GPT-5.6-Sol	Primary	23/1	32.8%	[31.0, 37.0]%
Claude-Sonnet-4.6	Replication	23/1	30.1%	[25.7, 34.0]%
DeepSeek-V3.2	Replication	23/1	32.8%	[29.5, 37.5]%
DeepSeek-V4-Flash	Replication	23/1	29.7%	[25.6, 35.2]%
Kimi-K3	Replication	23/1	34.2%	[30.5, 35.9]%

Qwen/GPT Holm-adjusted sign-test values are 2.77×10^{-4} and 5.96×10^{-6}. Exact-DEL-SAT is the strongest baseline in 20/24 tasks, deterministic refinement in three, and PSEC-1 in one. All 48 primary model rows are exact, budget compliant, and registry-certified. The campaign uses 280 provider calls, 428,172 tokens, and 995.5 summed guided-row seconds, below its frozen stopping limits. Small role-coupled tasks have the weakest median reductions (9.2% Qwen, 9.5% GPT), including losses of -17.4% and -7.6% on the same task.

The additional families were selected before their panel outputs. Each replication has 24/24 verified retained rows, uses the same stratified bootstrap units, and reports an uncorrected sign $p = 2.98 \times 10^{-6}$. Claude and DeepSeek-V3.2 were single-model replications. DeepSeek-V4-Flash and Kimi-K3 excluded timeout and duplicate-resume records; exclusion counts, units, replacement rules, and resource costs are unspecified. Their effects remain post-confirmatory and are not pooled with the primary intention-to-treat comparison or its Holm correction.

F.2.1 EARLIER PROPOSER PILOTS

On a 12-task Pair-SAT matrix, Qwen has 11 wins and one tie against lexical selection, but its 16.7% median round reduction misses the preregistered 20% gate. On a 14-case repository matrix it has three wins and 11 ties with zero median relative reduction; blinded deterministic history/cost selection beats Qwen on nine of 14 cases. Accepted traces are exact, budget compliant, and compiler-certified. These pilots are not pooled with the later comparisons and do not establish model necessity or generalization.

F.3 ACCESS-MATCHED AND FULL-SUPPORT BASELINES

PSEC-κ exposes at most κ live Pair-SAT conflicts per round and selects the public coordinate with the largest empirical cross-signature edge cut per unit cost. It cannot inspect support clauses, hidden role relevance, or the role map. Ordered and swapped pairs are blocked; every sampling call, decision, propagation, state, and materialized world is charged. This is an EC^2-inspired empirical control. PSEC-1 is the strongest non-model comparator in one of the six large-support interaction cells; PSEC-16 never wins after full accounting. A separate nine-cell stress test keeps the original Qwen tasks and other baseline rows fixed. Median total states are 2,496 for Qwen, 5,502 for deterministic, 5,094 for random, and 5,993 for Exact-DEL-SAT. PSEC-κ uses 5,416, 12,597, 40,339, and 130,755 states for $\kappa = 1, 4, 16, 64$. Qwen wins all nine cases against each method. Increasing the sample budget does not recover the model-guided ordering result on these cases; symbolic summaries or learned public heuristics remain possible alternatives.

F.3.1 FULL-SUPPORT AUTHORITY-SIGNATURE CONTROL

This control materializes exact support, groups worlds by the frozen singleton action–shared-prerequisite signature, and greedily maximizes expected cross-signature edge cut per unit cost. It applies the same authority-legality mask, certifier, and compiler. Its `FULL_SUPPORT_ACCESS` interface has the 8,192-world cap and provides more information than the lazy oracle. All 90 uncensored rows are safe, exact, and certified; six 32,768-world rows are censored. Completed methods have compiler expected query cost 3.9375. The control uses fewer post-selection verifier states, but materializes $195.81\times$ as many worlds as the best lazy method at 8,192 worlds.

Table 29: Full-support controls. (a) Worlds include construction and certificate representatives; states exclude support construction. (b) Released HEC metric with our recursive adapters. Times exclude exporting/materializing the hypothesis matrix; TO is the fixed 600-second timeout.

(a) Authority-signature control versus best lazy method

Support	Full worlds	Lazy worlds	Full states	Lazy states
32	64	39	150	360
512	544	42	246	905
8,192	8,224	42	374	1,427
32,768	censored	42	–	1,566

(b) Released HEC metric: unit-cost and cost-aware adapters

Support	HEC cost	Nodes	Time (s)	Cost-HEC cost	Nodes	HEC+Auth cost
1,024	10	127	0.005	17	2,047	10
8,192	10	127	0.041	21	16,383	10
32,768	10	127	0.176	24	65,535	10
131,072	10	127	0.871	TO	TO	10

F.3.2 Released HEC metric and CAGE

The retained HEC evaluation compiles the released metric of Javdani et al. (2014) unchanged under BSD-3-Clause. Our recursive adapters supply hypotheses, outcomes, costs, and signature regions with complete explicit support. The unit-cost adapter completes all 12 cardinality–support tasks at expected query cost 10. Cost normalization favors cheap nuisance tests, and all three 131,072-world runs time out. A diagnostic filter restricted to confirmation-bearing queries (HEC+Auth) restores cost 10 and 127 nodes.

On 24 matched adaptive authority cases (six per condition), all 72 HEC-adapter policies identify a region. Under the fail-closed terminal authority audit, certified mass is 0.30 for every adapter under narrow scope and 1.00 under wide scope. In Locus–Present cases, it is 0.20 except HEC+Auth with absent commitment, which reaches 0.00. RightSource reaches 1.00 with zero false autonomy in all four conditions. Filtering queries does not make region completion an authority-complete stopping rule.

The retained official CAGE reproduction, under MIT, passes 10/10 oracle tests, emits 1,332 records with zero C-witness invariant violations, and reproduces the published joint discrete/continuous non-composition examples. CAGE certifies a fixed action over a typed-return perturbation neighborhood; it does not select the next source or acquire human settlement authority. These external campaigns are not rerun by the present artifact, and their different interfaces preclude a shared solver-state ranking.

F.4 Same-compiler interface comparison

The interface study rejects malformed DAG entries, missing branches, wrong action labels, incomplete world sets, invalid scope/confirmation pairs, and exhausted repair budgets. It separately measures proposal construction, adapter work, native verification, shared repair, compiler verification, and certification. Action and authority omissions use one call/state/byte budget. All four interfaces lower to the same canonical input and have identical expected and worst-case query costs, selected-query count, policy signature, and authority completeness.

Table 30: Interface medians at $d = 20$ over five seeds. All 240 runs compile to the same canonical input and authority-complete policy.

Interface	Bytes	Adapter calls	Act. states	Auth. states	Repair calls
Query subset	141	0	26	25	2
Authority certificate	859	0	42	41	3
Ordinary worlds	2,541	48	80	80	2
Complete DAG	5,451	48	42	41	3

The query subset is the smallest serialized interface. Authority certificates can repair an omitted authority atom natively, but do not reduce total verifier work. The study therefore gives no efficiency advantage to certificate syntax; end-to-end model generation quality across representations remains untested.

G Artifact and Reproducibility

Table 31 distinguishes executable finite studies from recorded results. The finite artifact contains the compiler, observation-map checks, and study runners. The manuscript-source archive reproduces the document and result tables; it is not an executable evaluation package. Agent-level summaries support arithmetic checks, not trajectory-level replay or independent recomputation of task-level SEM and paired confidence intervals.

Table 31: Reproduction scope of the supplied evidence.

Evidence	Supplied material	Reproduction scope
Finite compiler	55 tests; six cases; 210 conditions and 420 method rows	Executable compilation, safety, authority, and cost checks
SourceGround-8	Eight cases; 72 conditions, 360 five-method rows, and 5,940 traces	Executable planning comparison and independent replay
AuthorityWitness-1	17 native controls; 9 conditions and 27 method traces; signed fixtures	Executable verifier controls, evidence admission, and planning replay
Constructed loss	1,512 conditions, 7,560 five-method rows, and 50,400 traces	Executable loss-aware planning and replay
InformationRouting-1	10 conditions; 56 method rows; 1,696 traces; 3,038 native proof checks and 864 descriptor reads per run	Executable matched source-selection and proof-acquisition comparison
Retail agent evaluation	Aggregate means and SEM, success counts, and evaluation and cost definitions	Table reconstruction and count-derived checks; not an agent rerun
Shared-gate Retail	Three-planner aggregates, paired intervals, event totals, joint completion counts, and reference-cost inputs	Count and cost checks; no task-level replay or interval recomputation
AgentDojo agent evaluation	Aggregate means and SEM, outcome counts, and attack and scoring protocol	Table reconstruction and count-derived checks; not trajectory replay
Mapping integrity	Activation tests and the 30-variant challenge records	Executable activation checks; recorded challenge results
Contract controls	Before/after fields and traces; 48 matched historical cases	Executable constructed controls; recorded historical audit
Scaling and model ordering	Scaling grids, 24-task primary state counts, and supplemental model summaries	Recorded comparisons and figure regeneration
External references	HEC comparison records and CAGE validation counts	Recorded results and capability comparison
LiveProbe and abnormal responses	Aggregate receipt, fault, and fallback results	Recorded tool-backed runs; no original runner supplied
Finite quotient and interface studies	Equivalence summaries and interface tables	Recorded replay results; not a rerun entry point