BEYOND CURRENT-TASK SUCCESS: REVISION SUFFICIENCY FOR LEARNING AGENTS*

Anonymous authors
Paper under double-blind review

ABSTRACT

Agents compress experience into reusable strategies for the objective at hand. A strategy that improves current execution may nevertheless become inadequate when the success criterion changes. We study *revision sufficiency*: whether a retained representation supports behavior under a specified revised criterion, without additional adaptation episodes before held-out evaluation. On procedurally generated tool-use tasks, we compare a trajectory archive with a strategy built from the same evidence for the original objective. Planning and execution are matched across representations within each stage. In a preregistered study of 600 seeds with 507 complete pairs, the strategy succeeds more often before revision (99.2% versus 89.2%) but 47.7 percentage points less often afterward. A separate control constructs the strategy for the revised objective under the same length cap; it then outperforms both the original-objective strategy and the archive after revision. These results distinguish the immediate usefulness of goal-specific preprocessing from the ability to reuse experience under a changed objective. Current-task acceptance can favor the former without establishing the latter. This paper identifies an overlooked factor in long-running self-evolving agent systems.

1 INTRODUCTION

Long-running agents reuse experience across model contexts. Persistent runtimes carry memory and skills between sessions (OpenClaw Foundation, 2026; Nous Research, 2026); in Claude Plays Pokémon, memory supports tens of thousands of interactions (Anthropic, 2025). Over such a run, the information and procedures needed for progress can change as later stages introduce new requirements. Agents therefore need more than a way to retain useful procedures: they need to reconsider those procedures when the objective changes. The question is whether the experience retained for an earlier objective still supports the later requirements.

Existing self-improving agents turn execution experience into prompts, workflows, and structured context, accepting an artifact when it improves current execution (Agrawal et al., 2025; Zhang et al., 2025; Wang et al., 2024; Lee et al., 2026). Such artifacts can help a planner by organizing experience into instructions for the objective it currently serves. However, a current-task acceptance test measures performance only under the criterion it evaluates. It can favor a strategy that makes the current procedure easy to use without testing distinctions that become relevant under a different objective. Good current performance is therefore not, by itself, evidence that the retained representation will support later requirements.

We call adequacy after a specified objective change *revision sufficiency*. It is relative to the revision, the model reading the representation, and the execution budget. We operationalize this question by expanding the set of state variables checked for success. The original facts and verdicts remain correct; the revised criterion simply requires more than the original one did. This separates the usefulness of retained experience under a new objective from the correction of an erroneous historical judgment.

Changing objectives already motivate reward-free exploration, which collects trajectories before a reward is specified (Jin et al., 2020), and successor features, which support transfer across rewards relative to a feature basis (Barreto et al., 2017). Our evaluation concerns a different object:

*Project homepage: `https://ledger-revision.github.io`

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.

Table 1: Acceptance, evidence links, and revision paths described in seven systems. "Not specified" means that the cited description does not give an explicit path for the indicated revision; it is not a claim that the system cannot be extended to support one.

System	Artifact acceptance	Evidence link	Revision path
GEPA	reflective prompt edits retained by score	candidate and score; no trajectory dependency	rescore only; no targeted revision
ACE	numbered playbook items merged incrementally	addressable item; no trajectory or verdict dependency	not specified
AWM	workflow from successful trajectories	none recorded	not specified
AHE	edit with failure evidence and predicted effect	evidence-to-edit pointer	reversible edit; corrected-verdict path not specified
Self-Harness	bounded edit checked for held-in/out regression	failure modes cite trajectories	not specified; verdict presumed
Meta-Harness	optimizer reads filesystem history	readable history; no dependency record	not specified
Heuristic Learning	policy-code edit tested and recorded	tests and failure directions	not specified for a faulty test

a language strategy constructed from experience for an earlier objective. Agent memory methods preserve detailed playbooks (Zhang et al., 2025), improve compression using current-task failures (Kang et al., 2025), or retain an experience pool alongside extracted insights (Zhao et al., 2023). We ask what happens when the criterion changes after the strategy has been constructed. Table 1 compares the acceptance and revision paths of seven systems; Appendix A situates this comparison among related evaluation settings.

To isolate this question, we hold the source experience fixed and compare two ways of retaining it: a trajectory archive T and a short strategy C_{g_0} constructed for the original objective g_0. The archive already contains evidence relevant to the revised objective g_1. At each stage, a planner reads one representation with the current objective and writes fresh execution guidance for a separate worker. Both representations are evaluated under matched conditions within that stage. This controlled handoff tests whether evidence that was available before compression can support behavior after revision, without requiring an end-to-end lifelong learning loop.

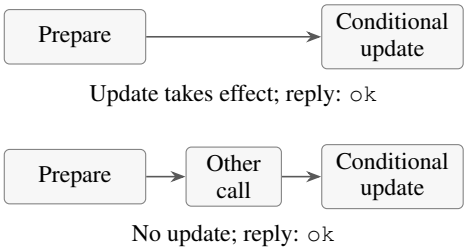
The paper makes three contributions.

- We define revision sufficiency alongside factual and behavioral sufficiency, and develop a paired evaluation of retained experience before and after an announced objective revision (Section 2).
- We report a ranking reversal: the original-objective strategy outperforms the archive before revision and underperforms it afterward. A control that changes the builder's objective shows that the same length cap can support a strategy that performs well under the revised criterion (Sections 3.1–3.2).
- We analyze the distinction between goal-specific preprocessing and revision-time reuse, using execution audits and controls to delimit the explanation. The results motivate retaining source evidence and recording the scope under which a strategy was evaluated (Sections 3.4–5).

2 Evaluating retained experience under objective revision

A failure after revision can arise because relevant evidence was never acquired, or because the retained experience no longer supports its use. We study the second possibility by starting from an archive that already demonstrates a revision-relevant rule. The original objective does not require that rule, whereas the revised objective does. Comparing the archive with a strategy built for the original objective then tests how the two representations support reuse of the same source experience. The task makes this distinction observable through a hidden tool-ordering constraint and an explicit expansion of the success criterion.

Call order determines the third update



Revision changes what success checks

	Original g_0	Revised g_1
Direct-update variables	checked	checked
Conditional-update variable	ignored	checked
Tool-call budget	14	26

The archive contains both call sequences. Only the revised objective requires the variable controlled by their ordering.

Figure 1: Why the task tests revision-time reuse. The archive demonstrates a history-dependent operation that the original objective does not require. Revision expands the evaluated state to include that operation's target variable. The operation labels shown here are explanatory; the agent sees opaque names and undocumented effects. "Other call" includes state reporting.

2.1 Procedurally generated tool-use tasks

We use tools with undocumented effects so that prior procedural evidence can inform execution. Each task instance exposes six operations with opaque names and argument schemas. The agent must infer their effects from observations and reach target values for three numerical state variables within a tool-call budget. The runtime calls these variables *slots*. Operation-role names below describe the task for the reader; they are not disclosed to the agent.

Two direct-update operations, called movers, each change one variable. A conditional-update operation, called the combiner, changes the third variable only when it immediately follows a preparation operation. Any intervening dispatched call, including a state-reporting call, cancels the preparation. This is the *ordering constraint*. A hidden flag records preparation but is absent from the reported state variables. A well-formed conditional-update call returns `ok` even when it has no effect, so the response alone does not reveal a violated constraint. State values wrap around a modulus, and another operation resets the state; Appendix C specifies the runtime and which calls count against the budget.

The original objective g_0 checks the two directly updated variables within 14 budgeted tool calls. The announced revision g_1 checks all three variables within 26 calls (Figure 1). The announcement specifies the new success criterion but does not reveal the ordering constraint. An external verifier checks the terminal state and call count, rather than the model's written answer. The expanded evaluation scope makes the ordering evidence relevant; the hidden preparation state and silent failure require the agent to infer the rule from call sequences rather than an explicit error message.

Each source seed defines a source instance and its held-out instances. Held-out instances rename operations and state variables and change constants while preserving the operation rules. The agent can reuse a procedure for identifying tool roles, but cannot solve the task by copying source names or constants. A reference solver succeeds on $200/200$ instances spanning 50 seeds and four kinds of instance change (Table 14). The primary experiment uses renamed instances and one announced expansion of the evaluated variables; no stored fact or historical verdict is corrected.

2.2 Source experience and retained representations

The source archive is constructed to contain the evidence needed for both objectives. It holds three episodes from a reference solver and a fixed probe policy: a complete solution, a solution that sets only the two directly updated variables, and a probing episode. The probe includes both an effective prepare–combine sequence and an interrupted sequence with no effect. Thus evidence of the ordering constraint is present before either representation is evaluated.

Archive T records the ordered calls and observations, source-instance metadata, and terminal state. Historical evaluations V_{g_0} accompany the observations and specify each episode's verdict and the variables checked. Source metadata includes the variable names, modulus, and target. The archive

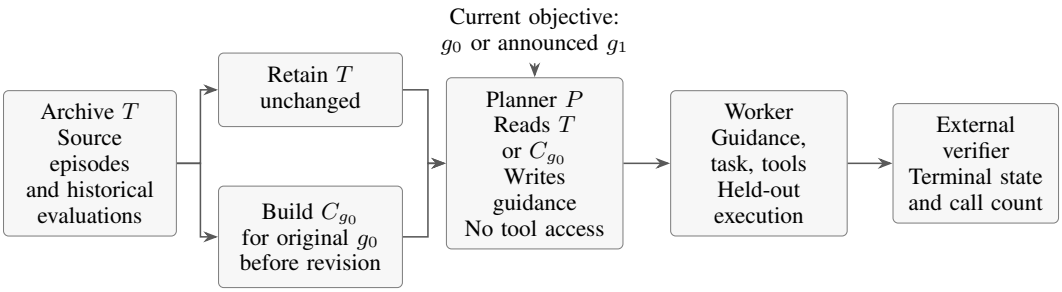


Figure 2: Controlled information handoff. The planner's retained input changes between the archive and strategy paths. Both receive the current objective and produce fresh execution guidance through the same pipeline. No new adaptation episode precedes either evaluation.

excludes the hidden preparation flag, effective-update counts, and held-out instance parameters (Appendix C).

A builder model M reads T with g_0 and writes the original-objective strategy $C_{g_0} = M(T, g_0)$ in one call, with a 220-word cap. The primary comparison is between retaining T and retaining C_{g_0}. The archive is deliberately small: the pilot reports approximately $1{,}800$ characters for the archive and $1{,}000$ for the strategy (Table 16). The experiment therefore concerns the objective used to construct a representation, rather than pressure from a context-length limit. Additional representation conditions are specified in Appendix C.3.

2.3 Matched planning and execution

To compare the retained representations rather than different execution systems, we use the same planner and worker for both paths (Figure 2). At each stage, planner P reads one representation, the current objective, the tool-call budget, and any announcement. It has no environment access and writes fresh *execution guidance* of at most 220 words. The worker receives this guidance, the task statement, and the tools, but not the retained representation. It then acts on the held-out instance. The strategy is therefore a persistent input to planning; the guidance is a newly generated instruction text for one episode.

Within each stage, the two paths share the tool interface, model settings, execution limits, and external scoring. The original and revised stages are separate episodes with fresh model contexts. There are no additional adaptation episodes before evaluation, although the worker can explore within the evaluated episode, without verifier feedback. Across stages, the primary design changes the objective, announcement, budget, and held-out instance together; separate controls hold selected factors fixed (Section 3.3). All primary roles use `ds-v4.1-flash` at low reasoning effort (Table 2). A separate worker-harness guard limits episodes to 24 model turns. This guard was not declared at registration; its unequal impact across paths and the associated sensitivity analysis are reported with the primary result.

2.4 Evaluation criteria and paired design

Following the comparison-of-experiments view (Blackwell, 1953), we treat sufficiency as adequacy for a stated use. *Behavioral sufficiency* asks whether a representation supports held-out success relative to the archive under the original objective. *Revision sufficiency* applies the same comparison under the revised criterion, with planning and execution matched across representations. These are behavioral comparisons for a specified reader and budget, rather than claims that a representation preserves every fact in its source.

Factual sufficiency is an auxiliary diagnostic: preregistered questions test access to recorded states, action ordering, counterexamples, evaluation scope, and unknowns. Each factual probe uses a separate model context, so its answers never enter behavioral evaluation (Appendix C). The three criteria

distinguish answering questions about experience, using it for the original task, and using it after revision.

The primary experiment assigns $N = 600$ source seeds. For each seed, both T and C_{g_0} are used once under each objective, giving a paired 2×2 design with one episode per seed–representation–stage condition. The source seed is the statistical unit. For source seed i, let Y_{isa} be binary success for representation $a \in \{T, C\}$ under objective condition $s \in \{0, 1\}$, where $C = C_{g_0}$. If n_{pair} seeds have valid outcomes in all four conditions, their success rate is $B_s(a) = n_{\text{pair}}^{-1} \sum_i Y_{isa}$. Define the original-objective strategy advantage as $\delta_0 = B_0(C) - B_0(T)$ and the revised-objective archive advantage as $\Delta_1 = B_1(T) - B_1(C)$. These signs favor the strategy before revision and the archive afterward. For each paired seed, the change in relative advantage is $d_i = (Y_{i1T} - Y_{i1C}) - (Y_{i0T} - Y_{i0C})$. Averaging over the same seeds yields

$$D_{\text{rev}} = \frac{1}{n_{\text{pair}}} \sum_i d_i = \big[B_1(T) - B_1(C)\big] + \big[B_0(C) - B_0(T)\big] = \Delta_1 + \delta_0. \tag{1}$$

Two preregistered criteria test original-task non-inferiority, through δ_0, and an increase in the archive's relative advantage after revision, through D_{rev}. Appendix E defines the decision rules and paired intervals; Appendix O defines valid outcomes and transport-only retries. A separate prerequisite study confirms that retained experience can change held-out behavior on the renamed-instance workload (Appendix I.1).

3 Current success does not establish revision sufficiency

The central comparison asks whether the representation favored by the original objective remains the better input to planning after revision. Because both paths receive the new criterion, a post-revision difference reflects how their retained experience supports the revised task, not a failure to announce the change.

3.1 Paired comparison and ranking reversal

Of the 600 preregistered seeds, 507 have valid outcomes in all four conditions and form the complete paired set. On these seeds, the original-objective strategy succeeds more often than the archive before revision: 99.2% versus 89.2%. After revision, their ranking reverses: C_{g_0} succeeds on 5.3% and T on 53.1%. The paired archive advantage is 47.7 percentage points, with a two-sided 95% interval of $[43.1, 52.3]$ (Figure 3a,c). Both preregistered criteria pass. The before-and-after ranking is established by the separate two-sided paired contrasts; the decision rules and non-inferiority margin are detailed in Appendix E.

The direction is robust to the missing outcomes and the undeclared turn guard. The 24-turn guard ended $95/547$ valid post-revision strategy episodes and $36/555$ archive episodes early; these remain failures rather than exclusions. Assigning every missing archive outcome to failure, every missing strategy outcome to success, and every turn-capped strategy failure to success still leaves an archive advantage of at least 19.7 percentage points over all 600 assigned seeds. This is a finite-sample sensitivity bound, not an estimate of performance without the guard (Appendix O). The gap also persists among seeds on which both paths succeeded before revision (Appendix G).

3.2 The builder's objective changes post-revision performance

The reversal could reflect the limited length of a strategy or the objective used to construct it. To distinguish these explanations, a registered control on a separate 80-seed block gives the builder g_1 instead of g_0. The archive, builder model, effort, schema, word cap, and verified build-call budget are held fixed. This produces $C_{g_1} = M(T, g_1)$, a strategy constructed once the revised objective is known.

After revision, C_{g_1} succeeds in 68.4% of its valid episodes ($52/76$), compared with 51.3% ($41/80$) for the archive, and ranks above C_{g_0} (Figure 3b,c). Thus the length cap alone does not explain the primary strategy's deficit. Under that same cap, construction for g_1 yields a strategy that supports the revised task. The control also shows why the archive is a reference for retained evidence rather

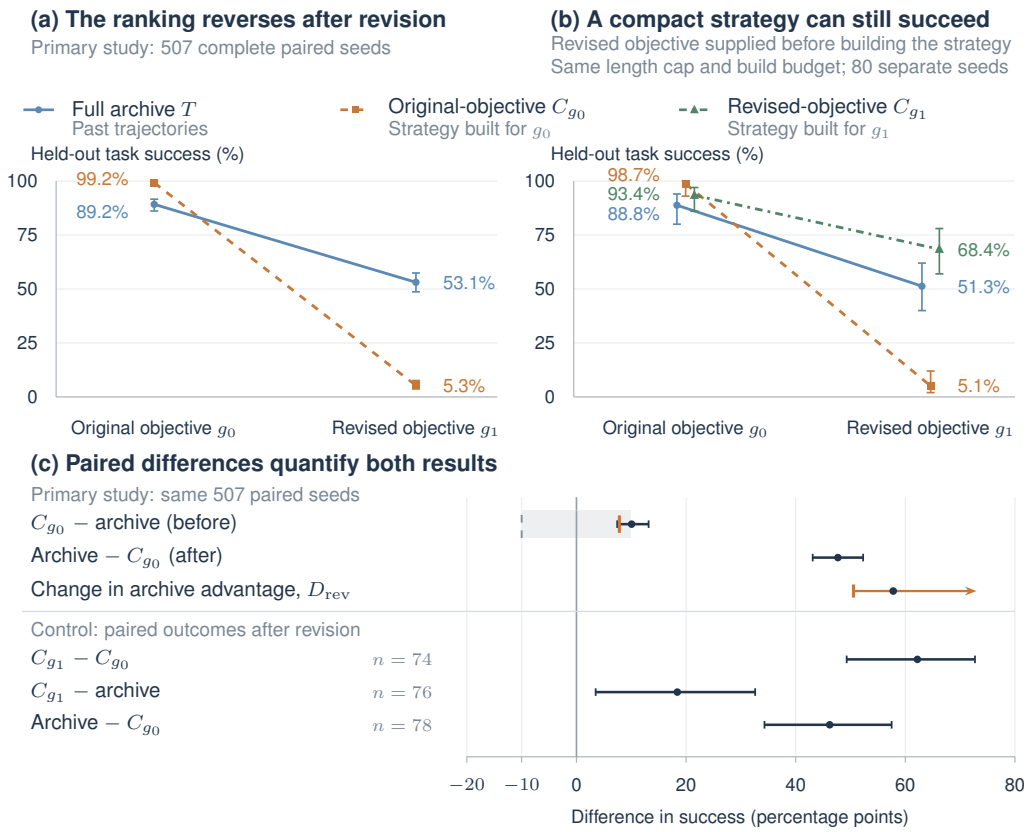


Figure 3: Original-task performance and revision-time performance rank the representations differently. (a) The primary comparison uses the same 507 seeds with valid outcomes in all four conditions. (b) A separate 80-seed control constructs a strategy for the revised objective; marginal success rates use each condition's valid denominator. (c) Paired success-rate differences; D_{rev} is the change in the archive's relative advantage across stages. Intervals in (a, b) are Wilson 95% intervals; (c) uses two-sided Tango 95% intervals. Orange marks show the one-sided bounds used by the preregistered criteria (Appendix E).

than an upper bound on the performance of a fixed planner: objective-specific preprocessing can make that evidence easier to use.

3.3 Objective revision matters with instance and budget held fixed

The primary stages change more than the success criterion, so separate controls examine the contributions of the objective, announcement, budget, and held-out instance. With instance and budget fixed, changing the objective and its announcement increases the archive's relative advantage over C_{g_0} by 52.5 percentage points $[43.2, 62.7]$ (Figure 4a). Increasing the call budget widens the advantage under g_1 but not detectably under g_0. The instance-only contrast remains unresolved. Instance change may therefore contribute to the primary result, but cannot account for the full change in relative performance (Appendix H).

The ranking reversal also appears when `MiniMax-M3` replaces the planner and worker while the retained representations remain fixed. On 109 complete paired seeds, the strategy leads by 26.6 percentage points $[15.0, 37.7]$ before revision and trails the archive by 12.8 $[5.9, 21.0]$ afterward. This second-reader comparison retains the direction of the result with a smaller post-revision gap (Appendix H.5).

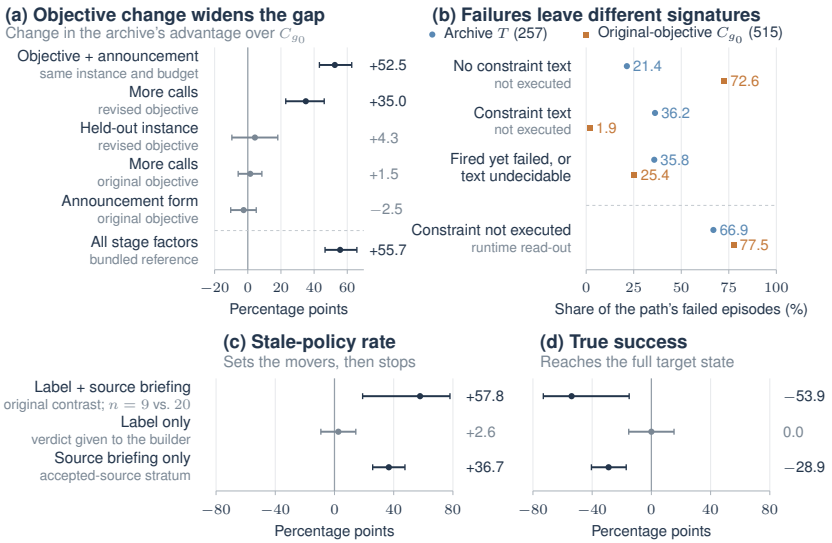


Figure 4: Controls distinguish changes in execution, explicit guidance, and source experience. (a) Changes in the archive's relative advantage; the bundled reference uses an independent block. (b) Post-revision failure signatures, as shares of each path's failed valid episodes. Rows overlap; "fired yet failed, or text undecidable" combines a runtime outcome with an undecidable text class. (c, d) Source-generation controls: faulty or unbriefed minus correct or briefed, for stale-policy rate and true success. Intervals are 95%; contrasts use separate analysis sets. Panel (a) uses paired Tango intervals. Details are in Appendices H and I.4.

3.4 Tracking the ordering constraint from strategy to execution

To understand the behavioral difference, we track whether the ordering constraint is explicitly stated in the strategy, appears in the planner's guidance, and is executed by the worker. An offline replay applies a fixed text-matching criterion to saved model outputs and reads execution from the recorded tool calls. Under that criterion, C_{g_0} states the constraint in $31/557$ built artifacts, compared with $53/76$ for C_{g_1} on its separate control block. Among the 547 valid post-revision C_{g_0} episodes, the planner carries the constraint into guidance for $24/30$ strategies with a matched statement, but adds it for only $2/517$ without one. The constraint is therefore rarely explicit at construction and is rarely introduced during planning when the strategy lacks a matched statement.

The execution records also show how the ordering can be disrupted. Report calls intervene between preparation and the conditional update in 26.4% of post-revision conditional-update calls under C_{g_0}, versus 16.2% under T; these are unpaired call-level shares. Such a report cancels preparation while the subsequent update still returns `ok`. Figure 4b summarizes the failure signatures, and Figure 5 replays one outcome-selected case (Appendix C.4). Text matching can miss paraphrases, and workers execute the constraint in $127/547$ strategy episodes without a matched statement in the guidance. The audit therefore describes the propagation of explicit instructions and observed actions, not the causal share of failures attributable to omitted evidence (Appendix H.4).

To test whether the post-revision difference is already visible at the local decision that enforces the ordering constraint, we run a separate read-out with the open-weight `gemma-4-E2B-it` model in bf16. The model reads one stored representation and acts as its own worker at a scripted point on the held-out instance. After a fixed prefix identifies the operation roles and demonstrates one successful preparation–combine pair, the preparation operation is called again and we score the next call. On $497/507$ seeds this choice cleanly separates the two actions: the combiner completes g_1, whereas a state-reporting call cancels the preparation. On the remaining ten, the prefix has already completed g_1 or the generated instance repeats a state-variable name; excluding these seeds changes no estimate by more than 0.01 nats or 0.2pp.

The reader's log-probability margin for the combiner over the report is 2.01 nats higher from T than from C_{g_0} ($[1.70, 2.32]$), and its greedy next call is the combiner on 39.3% versus 18.1% of seeds

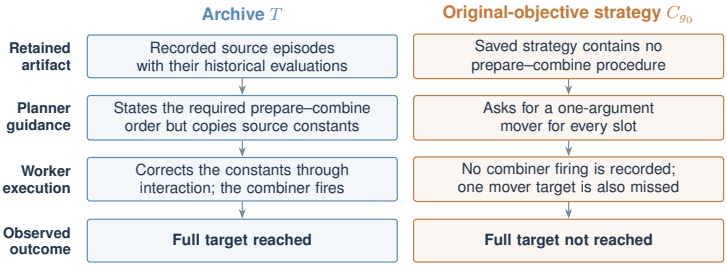


Figure 5: Replayed post-revision case, selected by outcome rather than drawn at random. Boxes restate the saved guidance and execution; they are not a verbatim tool log.

(+21.1pp $[+16.9, +25.4]$). On the builder-objective block, the same margin is 1.93 nats higher from C_{g_1} than from C_{g_0} ($[1.23, 2.65]$, $n = 74$). Reordering or deleting the archive's interrupted preparation–combine segment does not reduce the margin. This diagnostic therefore shows a corresponding representation difference at one post-revision decision, but does not identify which archive content produces it (Appendix H.6).

3.5 Goal-specific preprocessing and retained evidence

Why can the original-objective strategy outperform its source archive before revision? A strategy does more than shorten a record: it organizes the experience into instructions for a particular objective. For a planner with limited computation, this preprocessing can make the procedure easier to use (Xu et al., 2020). Appendix D.1 formalizes the distinction between available information and the computation needed to use it. For objective g, let $b_g(R)$ be expected success with the fixed planner reading representation R, $U_g(R)$ the corresponding bound for planners with unrestricted internal computation, and $\Gamma_g(R) = U_g(R) - b_g(R) \geq 0$ the fixed planner's shortfall. Under the stated assumption that an unrestricted archive reader can simulate the builder, $U_g(C_{g_0}) \leq U_g(T)$, and

$$b_g(C_{g_0}) - b_g(T) = \underbrace{U_g(C_{g_0}) - U_g(T)}_{\text{information term } \leq 0} + \underbrace{\Gamma_g(T) - \Gamma_g(C_{g_0})}_{\text{planner-shortfall reduction}}. \tag{2}$$

The strategy therefore adds no information beyond its source, yet can reduce the fixed planner's computational shortfall. The observed pre-revision advantage is consistent with this usefulness of goal-specific preprocessing.

The objective change tests a different requirement. The ordering evidence was present in the archive, unnecessary for g_0, and seldom explicit in C_{g_0}. Building for g_1 makes the constraint more frequently explicit and improves revised-task performance. These results implicate the construction objective, while leaving unresolved how much of the behavioral gap comes from information absent from the strategy and how much comes from difficulty using what remains.

A post-hoc selection analysis illustrates the consequence for current-task acceptance. A rule that chooses C_{g_0} whenever it does no worse than T before revision, and chooses T otherwise, selects the strategy on $503/507$ seeds. The representations selected by this rule succeed after revision on $30/507$ seeds (5.9%), compared with $269/507$ (53.1%) for always retaining the archive (Appendix G). This is a re-tabulation of the observed outcomes, not a selection step used in the experiment. It shows how a preference justified by current performance can select the less useful representation for a later criterion.

4 Evidence acquisition and revision-time reuse

The main comparison fixes an important prerequisite: revision-relevant evidence is already in the archive. A separate transfer study examines the earlier acquisition step by changing the instructions used to generate source experience. This asks whether a later representation receives the evidence needed for the task at all, rather than how it reuses a fixed archive.

In this study, the source operator is the model that generates the source episodes. The original comparison varies both the verifier that accepts those episodes, faulty ϕ versus correct ϕ', and the operator's briefing, a description of the task regularity with execution advice. Two registered controls separate the label supplied to the builder from the instructions supplied to the operator. Among source instances accepted under both briefing conditions, experience from unbriefed operators yields a held-out stale-policy rate 36.7 percentage points $[25.8, 47.6]$ higher than experience from briefed operators. Here the stale procedure sets the two directly updated variables without executing the conditional update. This is a total effect through the generated trajectory within the selected stratum. Changing only the verdict label supplied to the builder detects no effect on that rate ($+2.6$ percentage points $[-9.2, 14.4]$; Figure 4c,d; Appendix I.4).

The distinction is between evidence acquisition and subsequent reuse. Source-generation instructions can change what enters the archive; the primary objective-revision experiment holds that evidence fixed. Retaining an archive preserves a rebuild option only for evidence that was actually acquired.

5 Discussion and conclusion

This work introduces revision sufficiency as a distinct criterion for evaluating retained experience. Current-task acceptance establishes usefulness only under the criterion it tests; it does not establish adequacy after that criterion changes. In the primary experiment, the original-objective strategy is favored before revision and disfavored afterward, even though the source facts and historical verdicts remain correct. The C_{g_1} control further shows that this reversal is not a generic consequence of compactness: under the same length cap, a strategy built for the revised objective performs well after revision.

The broader implication is that goal-specific preprocessing and revision-time reuse are different system objectives. A strategy can make experience easier for a bounded planner to use now while failing to preserve distinctions needed later. Recording the construction objective and evaluated scope makes this boundary explicit, while retaining source evidence preserves the option to rebuild when requirements change. Appendix D develops the proposed LEDGER design and its cost model; these are design implications rather than evaluated recovery methods.

The evidence is limited to one procedurally generated task family, one announced expansion of the evaluated state, and isolated information handoffs. The fixed-representation second-reader study reproduces the ranking reversal, whereas a separate end-to-end transfer replication did not meet its registered effect criterion (Appendix N). An open-weight reader shows the same representation ordering at one scripted post-revision decision, while the registered gradient diagnostic does not qualify (Appendix H.6). The result therefore concerns support for a specified revision, not a universal preference for archives over compact strategies.

The central finding is that current-task usefulness and revision sufficiency can diverge. For long-running agents, evaluating retained experience therefore requires asking not only whether it improves the current task, but also whether it still supports behavior when the requirements change.

AI USE STATEMENT

We used generative AI tools to assist with software implementation, experimental monitoring, and manuscript preparation. In particular, AI tools were used to write and revise research code, inspect experiment execution and logs, and produce initial manuscript drafts from human-authored outlines. All AI-assisted code was reviewed and tested by the authors, experimental results and claims were checked against recorded outputs, and AI-generated manuscript text was reviewed and revised by the authors. The authors made the final decisions regarding the research questions, methodology, experimental design, interpretation, and presentation, and take responsibility for all content in this work.

REFERENCES

Lakshya A. Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J. Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Dan Klein, Alexandros G. Dimakis, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning, 2025. URL https://arxiv.org/abs/2507.19457.

Anthropic. Claude's extended thinking. https://www.anthropic.com/news/visible-extended-thinking, 2025. Section "Claude plays Pokémon".

Andre Barreto, Will Dabney, Remi Munos, Jonathan J. Hunt, Tom Schaul, Hado P. van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/hash/350db081a661525235354dd3e19b8c05-Abstract.html.

David Blackwell. Equivalent comparisons of experiments. *The Annals of Mathematical Statistics*, 24(2):265–272, 1953. doi: 10.1214/aoms/1177729032.

Chi Jin, Akshay Krishnamurthy, Max Simchowitz, and Tiancheng Yu. Reward-free exploration for reinforcement learning, 2020. URL https://arxiv.org/abs/2002.02794.

Minki Kang, Wei-Ning Chen, Dongge Han, Huseyin A. Inan, Lukas Wutschitz, Yanzhi Chen, Robert Sim, and Saravan Rajmohan. ACON: Optimizing context compression for long-horizon LLM agents, 2025. URL https://arxiv.org/abs/2510.00615.

Yoonho Lee, Roshen Nair, Qizheng Zhang, Kangwook Lee, Omar Khattab, and Chelsea Finn. Meta-Harness: End-to-end optimization of model harnesses, 2026. URL https://arxiv.org/abs/2603.28052.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474, 2020.

Jiahang Lin, Shichun Liu, Chengjun Pan, Lizhi Lin, Shihan Dou, Zhiheng Xi, Xuanjing Huang, Hang Yan, Zhenhua Han, Tao Gui, and Yu-Gang Jiang. Agentic harness engineering: Observability-driven automatic evolution of coding-agent harnesses, 2026. URL https://arxiv.org/abs/2604.25850.

Nous Research. Hermes Agent: The agent that grows with you. https://hermes-agent.nousresearch.com/, 2026.

OpenClaw Foundation. OpenClaw documentation. https://docs.openclaw.ai/, 2026.

Toshiro Tango. Equivalence test and confidence interval for the difference in proportions for the paired-sample design. *Statistics in Medicine*, 17(8):891–908, 1998.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory, 2024. URL https://arxiv.org/abs/2409.07429.

Jiayi Weng. Learning beyond gradients. Blog post, May 2026a. URL https://trinkle23897.github.io/learning-beyond-gradients/.

Lilian Weng. Harness engineering for self-improvement. Lil'Log, July 2026b. URL https://lilianweng.github.io/posts/2026-07-04-harness/.

Zidi Xiong, Yuping Lin, Wenya Xie, Pengfei He, Zirui Liu, Jiliang Tang, Himabindu Lakkaraju, and Zhen Xiang. How memory management impacts LLM agents: An empirical study of experience-following behavior, 2025. URL https://arxiv.org/abs/2505.16067.

Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. A theory of usable information under computational constraints. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1eBeyHFDH.

Nilutpaul Sarker Yash, Tirtho Roy, and Ushashi Bhattacharjee. Towards reversible forgetting: Managing obsolete knowledge in continual enterprise ai agents, 2026. URL https://arxiv.org/abs/2608.18177.

Hangfan Zhang, Shao Zhang, Kangcong Li, Chen Zhang, Yang Chen, Yiqun Zhang, Lei Bai, and Shuyue Hu. Self-Harness: Harnesses that improve themselves, 2026. URL https://arxiv.org/abs/2606.09498.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Zou, and Kunle Olukotun. Agentic context engineering: Evolving contexts for self-improving language models, 2025. URL https://arxiv.org/abs/2510.04618.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners, 2023. URL https://arxiv.org/abs/2308.10144.

Junhao Zheng, Xidi Cai, Qiuke Li, Duzhen Zhang, ZhongZhi Li, Yingying Zhang, Le Song, and Qianli Ma. LifelongAgentBench: Evaluating LLM agents as lifelong learners, 2025. URL https://arxiv.org/abs/2505.11942.

Zexuan Zhong, Zhengxuan Wu, Christopher D. Manning, Christopher Potts, and Danqi Chen. MQuAKE: Assessing knowledge editing in language models via multi-hop questions, 2023. URL https://arxiv.org/abs/2305.14795.

A EVIDENCE BEHIND THE SYSTEM COMPARISON

The comparison in Section 1 concerns retained strategies after an objective change. Related evaluation settings address other changes: knowledge editing revises a stored fact and tests dependent answers (Zhong et al., 2023), while lifelong-agent benchmarks measure accumulation and forgetting (Zheng et al., 2025). Studies of experience-following behavior also show that stored errors can influence later execution (Xiong et al., 2025). These settings motivate distinguishing changes in objectives from corrections to historical facts or verdicts.

Table 1 compares the acceptance rules, evidence links, and revision paths described in seven systems. The paragraphs below explain what each published mechanism records and which parts of a later revision path it specifies.

GEPA. GEPA uses execution trajectories and textual feedback to reflect on candidate prompts, retaining candidates by score (Agrawal et al., 2025). Its population can be rescored under another evaluator. The retained prompt, however, does not identify which rule was induced from which trajectory or verdict. Rescoring can reveal a changed aggregate ranking, but it does not locate the affected prompt content or define how to revise that content while preserving unaffected behavior.

ACE. ACE counters brevity bias and context collapse by preserving detailed knowledge through incrementally maintained playbooks (Zhang et al., 2025). It creates and merges numbered context items using execution feedback. Numbered items are addressable, which supports later editing, but the described representation does not link an item to the trajectories and evaluator judgments that licensed it. When a judgment changes, addressability alone does not determine which items require review or whether their downstream behavior remains adequate.

AWM. AWM induces reusable workflows from successful trajectories (Wang et al., 2024). The workflow is the retained artifact; the published mechanism does not preserve a dependency from its steps to the supporting trajectories or their success judgments. A later objective change can therefore alter whether the same steps are useful without supplying a system-level route from that change to workflow reconsideration.

AHE. AHE comes closest to an explicit evidence link: a harness edit ships with failure evidence and a predicted behavioral effect, and component edits are reversible (Lin et al., 2026). The pointer runs from evidence to a proposed edit and its check. The described lifecycle does not say how a corrected verdict invalidates earlier evidence, identifies every dependent edit, or triggers re-evaluation under the corrected criterion.

Self-Harness. Self-Harness bounds a proposed edit and accepts it only when held-in and held-out checks do not regress; its failure descriptions cite trajectories (Zhang et al., 2026). Those checks control acceptance under the current evaluator. If a test or failure label is later corrected, the published acceptance rule does not version that judgment or revisit edits that previously passed because of it.

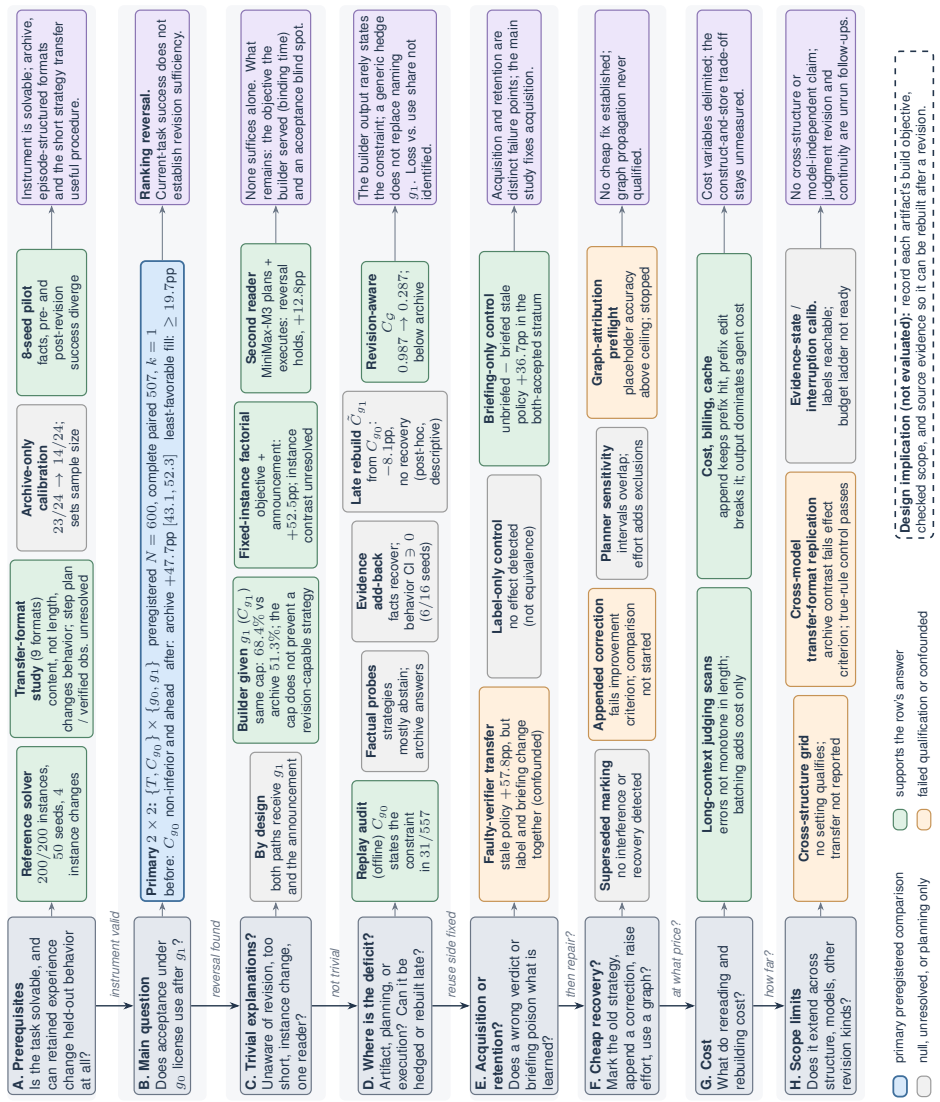
Meta-Harness. Meta-Harness lets the optimizer read filesystem history while optimizing the harness end to end (Lee et al., 2026). Readable history supplies broad context, but it is not a dependency record connecting each retained change to the evidence and judgments it used. The published mechanism therefore does not provide targeted invalidation or behavioral re-acceptance after one of those judgments changes.

Heuristic Learning. Heuristic Learning has a coding agent edit policy code, run tests, and record outcomes and written failure directions (Weng, 2026a). Tests and logs support iteration and rollback, but the procedure assumes that the tests remain authoritative. It does not specify how to reinterpret past runs, locate policy code accepted because of a faulty test, or revalidate that code after the test is corrected.

B EXPERIMENT INVENTORY

The experiments address different parts of the argument: whether prior experience affects execution, whether the primary representations change rank after revision, which controls constrain the explanation, and what evidence would be needed for a recovery method. Figure 6 organizes the main executed comparisons, audits, and prerequisite checks by these questions. An additional open-weight diagnostic probes one revision-relevant local decision: T yields a $+2.01$-nat combiner-versus-report margin relative to C_{g_0}, while archive interventions do not localize that difference. A separate gradient-direction diagnostic fails its registered qualification, so the g_0- versus g_1-accepted weight-space contrast is not interpreted (Appendix H.6). The task and information access are defined in Section 2; the full runtime follows in Appendix C.

Experiments use separate seed blocks, analysis sets, and denominators; their results are not pooled. The primary comparison and controls are distinguished from descriptive pilots and failed qualification checks. Appendix Q records follow-up designs whose intended behavioral comparisons were not run.



Additional registered open-weight diagnostics. Decision read-out (E39a): at one scripted post-revision choice, $m(T) - m(C_{g_0}) = +2.01$ nats; reordering or removing the interrupted archive segment does not localize the difference. Gradient direction (E39b): the registered $a(D_{g_1}) > 0$ qualification fails, so the g_0- versus g_1-accepted weight-space contrast is not interpreted.

Figure 6: Argument map of the executed experiments. Rows are the questions the paper asks; boxes are the experiments that bear on each, coloured by how their result reads; the right column is the answer each row licenses. The addendum records the later open-weight diagnostics so that the inventory remains complete. Numbers repeat those in Section 3 and the cited appendices, with their denominators; the map adds no evidence and pools no blocks.

C ENVIRONMENT, ARCHIVE, AND FACTUAL-PROBE DETAILS

This appendix specifies the task runtime, the archive, and the factual probes used in Section 2. The ordering constraint determines what counts as task state. The reported state variables alone are not Markov: preparation leaves them unchanged but alters the next combiner's effect. Fix θ, comprising the operation mapping, modulus, initial state, and conditional-update wiring. Physical transitions are deterministic in $x_t = (z_t, q_t)$, where z_t is the vector of numerical state variables and q_t the preparation flag: $x_{t+1} = F_\theta(x_t, a_t)$. Tool-call budget accounting and the worker's budget cutoff lie outside this physical transition. The model sees observations rather than x_t. For source episode $j = 1, \ldots, J$, with $J = 3$, the observed trajectory is $\tau_j^{\text{obs}} := \big(m_j, (a_{j,t}, o_{j,t+1})_{t=0}^{L_j-1}, r_j\big)$, where m_j gives slot names, modulus, and target, $a_{j,t}$ is a call, $o_{j,t+1}$ its return, L_j the number of calls, and r_j the terminal state-variable values. The existence of a predictively sufficient summary does not imply that a strategy written for g_0 is sufficient for another objective.

For recognized calls with valid arguments, the runtime clears the previous preparation flag before dispatch; only a preparation operation sets it again. Unknown operations and malformed integer arguments return before that reset and can consume budget without changing the physical state. Disconnected operations do not count against the tool-call budget. The free budget-control tool does not enter the task runtime. Thus "immediately after preparation" describes the order of successfully dispatched operations, not of every attempted API or tool request. The fire counter is an audit readout and is not used to update state-variable values or to score target attainment.

Table 14 defines the instance-change conditions and outcomes. The objective-revision experiment uses renamed instances with unchanged operation rules. Source exploration permits 200 budgeted tool calls; Section 2 gives the held-out budgets and objectives. Appendix O defines exclusions and retry eligibility.

C.1 NOTATION AND EXPERIMENTAL UNITS

Symbol	Meaning
T	Rendered observed trajectories with metadata and historical evaluations
g_0, g_1	Original and revised success criteria
C_{g_0}, C_{g_1}	Strategies built from T with the original or specific revised objective
$C_{\mathcal{G}}$	Strategy built with notice that objectives may change, without the specific g_1
P	Frozen planner mapping a representation and task information to execution guidance
$B_s(a)$	Held-out success rate for representation a at objective condition s
$b_g(R)$, $U_g(R)$	Expected success with the fixed planner and its unrestricted-computation bound
$\Gamma_g(R)$	Planner shortfall $U_g(R) - b_g(R)$ under matched guidance and execution limits
δ_0, Δ_1	Before-revision $C_{g_0} - T$ and after-revision $T - C_{g_0}$ success contrasts
D_{rev}	Change in the archive's relative advantage on the same paired seeds
N, n_{pair}, k	Assigned source seeds, complete paired seeds, and repeats per cell
ϵ	Frozen non-inferiority margin, justified for planning rather than at task level
ϕ, ϕ'	Faulty and corrected verifiers in the separate label-revision experiments
$\mathcal{D}$	Proposed record of evidence dependencies used to construct an artifact
M, J	Builder, as in $C_g = M(T, g)$; number of archive episodes ($J = 3$)
E_{tot}	Fixed episode budget in the planning variance identity
ω	Within-cell determinism parameter in the planning model; unidentified at $k = 1$

A source seed is the statistical unit; a cell is a seed–representation combination. Stage denotes an execution checkpoint, while objective condition denotes g_0 or g_1. Transfer-format comparisons vary episode delivery; representation comparisons hold the source bundle fixed. Method-level success

includes construction failures; success conditional on construction uses only seeds with a usable artifact.

C.2 Archive construction and behavioral checkpoints

The archive contains separate complete, movers-only, and operation-probing episodes, each starting from a fresh runtime. The complete plan sets the movers, prepares, and combines; its terminal state is appended from the runtime. The probe plan brackets operations with reports and includes a successful adjacent prepare/combine pair and a failed attempt after an intervening zero-delta mover. A failed combiner returns `ok`; one operation silently restores the initial state. The ordered contrast records evidence relevant to the revised goal.

The pre-revision budget was fixed before any compressed representation was built. A calibration showed that reducing the call budget made the task discriminate between blind and archive-guided execution; this calibration preceded the confirmatory protocol.

Separate copies of each representation receive the factual probe, pre-revision task, and post-revision task. Behavioral records contain evaluator acceptance, external ground-truth success, and terminal state. The planner has no environment access, and neither probes nor adaptation rollouts enter a behavioral copy.

C.3 Additional retained representations

Besides the original-objective strategy, the builder constructs a step plan, verified operation–effect observations, and a revision-aware strategy $C_{\mathcal{G}}$. These conditions use the same builder settings, call count, length cap, and schema. The revision-aware builder is told that objectives may change but is not given g_1. Together with the archive and a no-memory baseline, they form the six representation conditions. Only T and C_{g_0} enter the primary preregistered decision criteria.

A registered amendment limits the four secondary conditions to their own 150-seed block (Table 8). The revision-aware strategy succeeds on $147/149$ valid episodes before revision and $41/143$ afterward, below archive-level post-revision performance on that block. These secondary results retain their own denominators and do not enter the primary paired analysis.

C.4 Observed case selection

The saved case is the smallest seed among resolved primary seeds in the complete paired set for which the archive succeeds after revision, the original-objective arm fails, and the latter has no recorded combiner firing. The selection is outcome-conditioned, not a representative draw.

Figure 5 traces both paths from the retained artifact to the outcome. Replay confirms that both executions terminate normally and that only the archive path fires the combiner. The failed path also misses a mover target, so the example does not identify a unique cause.

C.5 What the factual questions can diagnose

Four of the six preregistered revision questions ask for source-instance facts: the combiner, preparation operation, reset operation, and state variables checked by the old verdicts. Those names need not transfer to a renamed instance. The other two ask whether an intervening call prevents combination and whether failed and successful combiner calls return the same response. This classification changes neither the frozen questions nor their scoring. A wrong answer can come from omission, retrieval, interpretation, or formatting, so on its own it does not prove that information was lost.

Runtime facts score probe answers automatically. Source-name recall need not predict action on a renamed instance, and the two question types behave differently. On the four name questions the compressed representations are near the floor because they abstain rather than answer wrongly: C_{g_0} answers $127/2224$ items correctly (0.057), abstains on 0.937, and errs on 0.004, against $2226/2268$ for the archive. On the two relational questions the archive answers $1134/1134$, C_{g_0} $10/1112$ on the 600-seed block and $0/156$ on the 80-seed block, $C_{\mathcal{G}}$ $10/298$, and C_{g_1} $33/152$ (0.217 $[0.159, 0.289]$), almost all from the ordering question ($32/76$). These are descriptive item counts clustered within seeds, reported per block without pooling. The factual probes measure access to the source record;

Table 2: Frozen role settings and information boundaries. The alternative planner is used only in the add-back utilization diagnostic.

Role	Model (effort)	Output cap	What it reads
Builder	`ds-v4.1-flash` (low)	32,000 tokens, 220 words	seed archive; one call per representation, matched settings for C_{g_0} and $C_{\mathcal{G}}$
Planner P	`ds-v4.1-flash` (low)	32,000 tokens, 220 words	representation, current rule, any announcement, and budget; no environment access; one call per stage
Episode worker	`ds-v4.1-flash` (low)	32,000 tokens	planner guidance and task tools; no within-episode verifier feedback
Probe copy	`ds-v4.1-flash` (low)	32,000 tokens	representation only; automatically scored JSON, no model judge
Alternative planner	`ds-v4.1-flash` (high)	48,000 tokens, 660 words	add-back utilization check only

the execution audit tracks explicit statements and observed actions (Appendix H.4). Appendix J tests whether adding evidence changes factual answers or held-out behavior.

D GOAL-SPECIFIC PREPROCESSING AND EVIDENCE-BASED REVISION

The analysis below distinguishes the information available in a representation from the computation a planner needs to use it. The later subsections develop the proposed evidence records and cost rule motivated by Section 5; these are design proposals rather than evaluated recovery methods.

D.1 INFORMATION AND COMPUTATIONAL SHORTFALL

For a fixed task distribution and objective g, let $b_g(R)$ be expected success with the fixed planner reading representation R, rather than a finite-sample rate $B_s(R)$. Let $U_g(R)$ be the supremum over planners with unrestricted internal computation but the same background knowledge, worker, tools, call budget, and guidance cap, without environment access. Define the planner's shortfall as $\Gamma_g(R) = U_g(R) - b_g(R) \geq 0$. A planner reading T that knows g_0 and can simulate the builder can first generate $C_{g_0} = M(T, g_0)$ and then emulate any strategy reader. The comparison averages over build and execution randomness; g_0 is available from the historical evaluation scope, and the primary builder and planner share a model. Under this simulation premise, $U_g(C_{g_0}) \leq U_g(T)$ (Blackwell, 1953). Substituting $b_g(R) = U_g(R) - \Gamma_g(R)$ into the expected success difference gives the decomposition in Equation 2. A positive expected strategy advantage lower-bounds the planner-shortfall reduction; preprocessing can make information more usable for a bounded reader (Xu et al., 2020). The observed advantage estimates a behavioral contrast, not either latent term separately. After revision, the same identity does not determine how much of the gap comes from information loss or computational shortfall. Here $U_g(T)$ is an idealized decision bound, not the measured archive success rate or a claim that the archive is complete for arbitrary revisions.

The bound includes whatever the fixed worker can rediscover through interaction within its call budget; it is not a property of the stored text alone. Changing that worker or budget changes the decision problem. If both planner shortfalls approach zero while this problem stays fixed, the strategy's success advantage approaches the nonpositive information term. A reader with a different prior need not satisfy the shared-information simulation premise; the second-reader comparison does not estimate this limit.

Artifact-level text checks could describe whether a condition was explicitly stated before planning, but absence of a textual match does not prove information loss. Even an explicit condition can fail through interpretation or execution, and interaction may recover an omitted condition.

The Information Bottleneck provides a complementary interpretation of goal-directed compression: it retains information predictive of a chosen target (**?**). In this task, the ordering contrast is unnecessary for g_0, so construction directed at that objective may have little incentive to state it. This

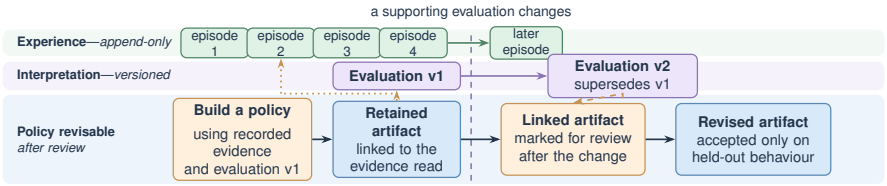


Figure 7: Proposed revision lifecycle. Experience is retained, evaluations are versioned, and links identify derived artifacts for review when an evaluation changes.

interpretation concerns how an objective can prioritize evidence; the builder audit in Appendix H.4 measures explicit text rather than an information-theoretic quantity.

D.2 Proposed evidence dependencies and versioned evaluations

LEDGER (learning with evidence dependencies under goal and evaluation revisions) is a proposed design for recording how retained artifacts were constructed. It keeps factual trajectories, versions their evaluations V, and records dependencies for learned artifacts A. The experiment's rendered archive T already includes historical evaluations alongside observations. The proposed dependency record $\mathcal{D}$ identifies the trajectories, evaluations, objectives, and upstream artifacts used to construct each artifact.

When an evaluation is corrected, $\phi \to \phi'$, the design re-evaluates the records, creates a new version in V, and identifies dependent artifacts for review and behavioral re-acceptance (Figure 7). The links identify review candidates rather than automatically invalid artifacts; evidence is retained rather than removed (Yash et al., 2026). For an expanded objective, the record must also state the scope of earlier acceptance, since newly relevant evidence may never have been used to construct the artifact.

Storage and dependency information serve different purposes. Files edited with shell tools (Weng, 2026b) or a retrieval index (Lewis et al., 2020) locate stored material; explicit links additionally record which evidence and evaluations supported its construction. A proposed graph rendering and its failed qualification check are documented in Appendix Q.4.

Acquiring a rollout, re-evaluating a trajectory, rereading evidence for temporary guidance, and revising a persistent artifact act on different objects. A label revision changes a verdict; an objective revision changes task requirements; an environment revision changes operation semantics; and a source revision changes trust in a judge. Existing observations may support the first two, while the latter two may also call those observations into question. Retention cannot restore unacquired evidence.

D.3 When to construct and store a reusable strategy

For artifact a, let G_a be construction and verification cost, N_a expected future uses, ΔC_a the saving per use, ρ_a the probability that a supporting evaluation changes, R_a re-evaluation and rebuilding cost, and L_a expected loss from the stale policy. The proposed rule constructs and stores a only when a conservative estimate of $N_a \Delta C_a - G_a - \rho_a(R_a + L_a)$ is positive. These cost terms remain unmeasured. Testing the rule requires varying reuse count and revision rate rather than only context length or model price. Precomputation motivates the per-use saving, while early binding creates exposure to rebuilding and stale-policy loss. The success-rate terms in Equation 2 are not monetary costs and cannot be substituted for these quantities. Even if a stronger planner closes a behavioral gap, reusing an artifact may still save repeated inference; this is an economic claim requiring cost measurement at matched behavioral quality. Evaluating a candidate revision family contributes to G_a, and any resulting acceptance claim remains limited to that family.

A rule-based alternative would retain temporal dependencies and uncertainty about untested interactions rather than only a procedure for the present objective. Its schema must express history-dependent effects such as preparation, and its adequacy must be tested over a stated objective family. Neither a short list of supplied true rules nor accurate answers to targeted probes establishes

Table 3: Structural floors imposed by the registered gate arithmetic. These are not power calculations.

Gate	Floor	Consequence
δ_0 non-inferiority	$n_{\min} = 24.35$ at $\epsilon = 0.10$, $\alpha = 0.05$	A fully successful $n \leq 24$ sample cannot pass by construction
D_{rev} summed bound	$b_1 + b_0 > 2z^2 = 7.683$	At least 8 favorable discordant objective-condition cells; *necessary*, not sufficient

that the builder can induce such a representation from finite evidence. An archive is also a chosen observation schema; retaining it cannot restore unobserved state or unrecorded interactions.

D.4 Evidence still required for a method claim

A recovery-method comparison would hold archive, feedback, interaction budget, and worker budget fixed across full rereading, rescore-and-continue, direct correction, and targeted revision. Its outcomes are held-out recovery, unaffected-task degradation, new rollouts, and total cost at matched behavioral quality.

The representation gates pass, but the registered prerequisite calibration does not. Appended correction fails the registered improvement criterion against full rereading, even under the missing-outcome sensitivity analysis. A high no-correction success rate leaves little behavioral deficit to recover, while factual recovery and behavioral success rank the conditions differently. The calibration therefore does not authorize a recovery-method comparison.

Artifact location and editing cost, together with degradation on unaffected tasks, also remain unmeasured. The proposed cost comparisons in Appendix Q would require these measurements rather than a comparison of provider prices alone.

E Inference details: gate floors, validity, and exclusions

This appendix defines the two preregistered decision criteria, called gates below, for the paired contrasts δ_0, Δ_1, and D_{rev} in Equation 1. The first gate requires a one-sided 95% Tango lower bound for δ_0 above $-\epsilon$, with frozen $\epsilon = 0.10$; the margin was registered for sample-size planning and has no task-level justification. The second requires the sum of one-sided Tango lower bounds at error level $\alpha/2$ for the two component contrasts to exceed zero, with $\alpha = 0.05$ (Tango, 1998). The estimands, decision criteria, sample size, and exclusion rules are unchanged from registration. Coverage checks use the planned rather than achieved sample size; no power or rejection rate is interpolated to the achieved paired set.

Here b_1 counts seeds favoring T after revision and b_0 those favoring C before it. Seven favorable discordant cells cannot pass the summed-bound gate. Eight can pass only with no reverse discordance and a feasible split across stages.

The earlier sign-flip gate on d_i was retracted because the design neither randomizes T/C labels nor supplies sign exchangeability. A preregistered coverage checker may exclude a source seed before execution if a relevant direction was never explored. After entry, archive failure is a valid zero and inclusion does not depend on T success.

Marginals use stage-specific valid denominators; d_i, Δ_1, and both gates use seeds with all four outcomes. The source seed is the statistical cluster and generates one task-family template, operation names, constants, and two held-out instances.

E.1 Approximate coverage and interpretation of the gates

Tango's matched-binary paired-proportion score interval (Tango, 1998) remains defined with zero off-diagonal cells or complete concordance. Because $d_i \in \{-2, \ldots, +2\}$, the D_{rev} gate sums one-sided Tango lower bounds at $\alpha/2$. The union bound permits arbitrary cross-stage dependence, while

Table 4: A constructed counterexample. Both frozen gates can pass at sufficient sample size, yet C is better than T before and after the revision.

Objective condition	Full trajectory T	Compressed representation C
before revision, B_0	0.80	0.95
after revision, B_1	0.50	0.55

$\delta_0 = +0.15 > -0.10$ (non-inferiority passes); $\Delta_1 = -0.05$; $D_{\text{rev}} = +0.10 > 0$ (summed gate passes at sufficient sample size). C leads at both stages throughout.

each stage's paired table still assumes independent source seeds. The resulting coverage is nominal rather than exact.

An offline null-boundary audit, frozen before effect read-out, finds a liberal minimum-discordance case only at the smallest checked sample. Every checked structure at the planned confirmatory size falls within the prespecified approximate-validity band. This conclusion is limited to the enumerated coupling family.

Passing both gates establishes current-task non-inferiority at the registered margin and greater relative degradation after revision, not current-task equivalence or post-revision archive superiority. Table 4 gives a case in which C remains better at both stages.

Post-revision superiority requires the two-sided Δ_1 interval; current-task equivalence requires a two-sided analysis at $\pm\epsilon$. These secondary read-outs use the same paired score construction without changing either gate.

E.2 Planning assumptions and confirmatory interpretation

We use $k = 1$, one planner call and episode per stage in each seed–representation cell. Nominal N, coverage-clean seeds, and complete paired n are therefore distinct (Table 6). Under the fixed-episode budget in Appendix F, the planning variance cannot decrease by replacing source seeds with repeated cells. This supports sampling more seeds under that budget but does not show that within-cell noise is small.

Sample size was frozen from the archive calibration and preregistered discordance scenarios. Table 5 reports the planning grid. The realized discordance pattern falls outside those scenarios and the achieved paired sample lies between checked grid points, so no achieved power or coverage rate is assigned.

F Unit of observation: the variance identity

This appendix explains why the design runs one episode per cell ($k = 1$). Under a fixed episode budget, repeating a cell leaves fewer source seeds available. The planning model expresses this tradeoff through the variance of the revision contrast. Let ω be within-cell determinism: 1 makes repeats exact duplicates, while 0 removes seed-level structure and most favors repeats. Let $m_{s,a}$ be the success probability for stage s and arm a, $S = \sum_{s,a} m_{s,a}(1 - m_{s,a})$, and σ^2_{arch} the seed-level archetype-contrast variance for a given cross-stage coupling. With k episodes per cell, a budget of E_{tot} episodes covers $E_{\text{tot}}/(4k)$ seeds and

$$E_{\text{tot}} \cdot \text{Var}\big(\widehat{D}_{\text{rev}}\big) = 4\big[\, k\,\omega^2\sigma^2_{\text{arch}} + (1-\omega^2)\,S\,\big], \tag{3}$$

This variance is non-decreasing in k for every coupling and ω, and increases strictly only when both $\omega > 0$ and $\sigma^2_{\text{arch}} > 0$. Under the fixed-episode budget, repeats cannot improve precision because they displace seeds, although they may reveal call-level randomness. The gate also requires favorable discordant cells (Table 3).

For $k > 1$, the compared estimator changes from Tango score inversion to bounded-mean bounds. Its structural floors differ, so the procedures cannot be compared by treating repeated episodes as additional independent seeds. Offline fixed-budget comparisons favor $k = 1$ across the checked

Table 5: Frozen sample-size calculation. Joint power is the probability that both gates pass on the same sample. Values are minima over the enumerated coupling family and four assumed discordance scenarios. Intervals reflect the replication count. Values come from deterministic offline planning simulations; no intermediate N was simulated.

N (source seeds)	minimum joint power	95% interval	replications	status
195	0.304	$[0.295, 0.313]$	10000	below target
300	0.589	$[0.579, 0.599]$	10000	below target
400	0.711	$[0.702, 0.720]$	10000	below target
600	**0.886**	$[\mathbf{0.879}, \mathbf{0.892}]$	10000	**frozen**; smallest tested grid point ≥ 0.80 (grid: 195, 300, 400, 600)

Table 6: Nominal sample size, coverage, and valid paired outcomes are different quantities. Only the first two are fixed before execution. Stage-specific marginal denominators accompany their results in Table 7.

Set	Seeds	What it counts
Frozen source list	600	Nominal N, not completed outcomes
Coverage-clean source seeds	600	Deterministic archive check, not execution validity
Both archive stages valid	553	Two valid outcomes for T
Both original-objective stages valid	547	Two valid outcomes for C_{g_0}
All four outcomes valid	507	Common paired set used by both primary gates

within-cell dependence settings, although repeats could help if source seeds rather than episodes were the limiting resource.

One Bernoulli draw per cell leaves ω unidentified, so the $k = 1$ choice is conditional on the fixed-episode budget. A repeated-planning diagnostic would measure call-level variability on fresh seeds without altering the confirmatory gates.

G Confirmatory objective-revision comparison: sample size and analysis sets

This appendix gives the sample-size calculation, the analysis sets, and supplementary re-tabulations of the primary comparison. The design fixes six representations, the two stage budgets, one episode per cell, $N = 600$, and the two gates in Section 3. Archive calibration supplies the planning marginals and within-archive pairing (Table 15).

A registered amendment caps the four non-gate arms without changing the primary estimators, gates, exclusions, or records. No memory, observations, and plan remain descriptive controls; $C_{\mathcal{G}}$ lacks multiplicity control. The primary archive–original-objective contrast keeps the frozen sample size and has no interim gate analysis. Reduced arms retain separate denominators and are never pooled with the primary pair.

The same assumed-discordance case binds every row in Table 5. The enumerated family includes extreme transportation-polytope couplings and Gaussian-latent couplings, but stage marginals do not identify cross-stage dependence and worse unenumerated couplings may exist.

The source list is the deterministic coverage-clean prefix of the screened candidate pool. Coverage applies to source evidence, not to the validity of all four behavioral cells.

Losses arise from both infrastructure and in-method failures; cells eligible for transport-only remeasurement are classified by the underlying failure rather than a generic failed-build status. A valid early-stage outcome remains in its marginal analysis if a later stage fails. Archive failures in valid episodes remain zeros, no seed is screened on archive success, and losses do not reselect the frozen source list.

Table 7: Primary-arm stage-specific marginals. Each stage retains all valid outcomes, so these denominators differ from the common paired set in Figure 3. Intervals are Wilson 95%; marginal differences are not paired estimates.

Representation	Before revision	After revision
Archive T	$503/562 = 0.895$ $[0.867, 0.918]$	$298/555 = 0.537$ $[0.495, 0.578]$
Original-objective C_{g_0}	$550/555 = 0.991$ $[0.979, 0.996]$	$32/547 = 0.059$ $[0.042, 0.081]$

Table 8: Reduced arms on the 150-seed capped block. Descriptive controls, capped by a registered amendment, each with its own valid denominator; Wilson 95% intervals; not pooled with the primary pair and not subject to multiplicity control.

Representation	Before revision	After revision
No memory	$96/150 = 0.640$ $[0.561, 0.712]$	$0/150 = 0.000$ $[0.000, 0.025]$
Step plan	$85/149 = 0.570$ $[0.490, 0.647]$	$3/143 = 0.021$ $[0.007, 0.060]$
Verified observations	$116/148 = 0.784$ $[0.711, 0.842]$	$1/143 = 0.007$ $[0.001, 0.039]$
Revision-aware $C_{\mathcal{G}}$	$147/149 = 0.987$ $[0.952, 0.996]$	$41/143 = 0.287$ $[0.219, 0.366]$

Table 8 reports the four reduced arms on their own 150-seed block; they are descriptive controls capped by a registered amendment and are never pooled with the primary pair.

Common-pass subset and either-path success. A supplementary re-tabulation of the recorded complete-paired outcomes, one execution per cell, restricts to the $448/507$ seeds on which both paths succeed before the revision. There the post-revision success is $236/448 = 0.527$ $[0.481, 0.573]$ for T and $24/448 = 0.054$ $[0.036, 0.078]$ for C_{g_0}, a paired difference of $+47.3$pp $[+42.4, +52.2]$ (Tango two-sided, $b = 219$, $c = 7$), so the gap is not carried by seeds where the strategy already failed the old task. Either-path success, $\max(Y_{1T}, Y_{1C})$, holds on $276/507 = 0.544$ seeds; of the 27 post-revision C_{g_0} successes, 20 are also archive successes and 7 are strategy-only. The complete-case primary analysis is unchanged.

Acceptance simulation. A post-hoc selection rule applied to the same recorded outcomes, not an acceptance step in any run, keeps C_{g_0} when it is not worse than the archive on the current task, breaking ties toward the shorter artifact. It retains C_{g_0} on $503/507 = 0.992$ seeds, as expected from $B_0(C) = 0.992$. The representations selected by the rule succeed after revision on $30/507 = 0.059$ $[0.042, 0.083]$, against $269/507 = 0.531$ for always choosing the archive and 0.544 for either-path success. The rule "keep C_{g_0} only if it succeeded" selects the identical set, because no seed fails both paths before revision. The comparison with always retaining T describes how selection on the original task changes post-revision performance. Either-path success is a post-hoc union of observed successes, not an executable selection policy.

Missing-outcome sensitivity bound. Let $x_{1T}/n_{1T} = 298/555$ and $x_{1C}/n_{1C} = 32/547$ be the post-revision success counts in Table 7, with $N = 600$ assigned seeds. Counting every missing archive outcome as a failure and every missing original-objective outcome as a success gives

$$\underline{\Delta}_1 = \frac{x_{1T} - \left[x_{1C} + (N - n_{1C})\right]}{N} = \frac{298 - (32 + 53)}{600} = \frac{213}{600} = 0.355. \tag{4}$$

Any completion of the 45 missing archive and 53 missing original-objective outcomes therefore leaves a post-revision difference of at least 35.5pp over the 600 assigned seeds. The bound is deterministic and finite-sample: it is not a confidence interval, supports no population inference, is stated only for the post-revision contrast, and does not replace the registered complete-case analysis.

A joint worst case also flips the 95 turn-capped C_{g_0} failures (Appendix O). They lie inside the 547 valid outcomes and are all failures, so they do not overlap the 53 missing ones, and counting them as successes gives

$$\underline{\Delta}_1^{\text{joint}} = \frac{298 - (32 + 53 + 95)}{600} = \frac{118}{600} = 0.197. \tag{5}$$

This is a finite-sample bound under a stated fill-in and flip rule, not a confidence interval and not an estimate of success without the turn guard; it is the bound quoted in Section 3.1.

Restricting the coupling family would make smaller samples appear adequate only under an unverifiable joint-distribution assumption. Cost did not determine N. Appendix L states the accounting scope; Appendix O states retry and exclusion rules.

H CONTROLS AND EXECUTION DIAGNOSTICS

The registered controls vary the construction objective or selected execution factors. The offline audit describes saved outputs and actions; the old-strategy-only reconstruction pilot is reported separately as post-hoc. Together these analyses support the interpretation of Section 3.

H.1 OBJECTIVE, BUDGET, ANNOUNCEMENT, AND INSTANCE

The primary stages change the objective, tool-call budget, announcement, and held-out instance together. Independent registered blocks hold selected factors fixed. Figure 4a plots their estimands; none changes the confirmatory sample or gates.

The fixed-instance objective contrast rules out instance change as the sole explanation. It includes the objective announcement. The instance contrast covers one pair at one objective and budget, so its interval cannot support a general no-instance-effect claim. The factorial block changes instance with objective and therefore cannot by itself attribute the bundled effect. Reference cells reproduce the frozen configurations.

H.2 CONSTRUCTION FOR THE REVISED OBJECTIVE

$C_{g_1} = M(T, g_1)$ matches C_{g_0} in archive, builder, effort, schema, length cap, and build budget, differing only in the build-prompt objective. A fresh disjoint block supplies one episode per stage and one verified equal-budget build call per compressed arm. Figure 3 reports all success rates and paired contrasts.

The registered branch passes: C_{g_1} remains adequate before revision and ranks above the archive afterward, while the archive–C_{g_0} direction reproduces without pooling. The result concerns target-conditioned building for the tested builder, cap, archive, planner, worker, and stage budgets.

A commitment re-score finds that missing fields in the compressed conditions are predominantly abstentions rather than definite errors. Because the design records no confidence signal or abstention criterion, this is not a calibration result.

Reconstruction from the old strategy alone. A post-hoc descriptive diagnostic asks whether a revised-objective rebuild can succeed without the archive. Its registration file was written after the first API call, so no preregistered decision rule is applied. On a fresh disjoint block of 40 source seeds, three arms per seed share the build-call budget, cap, model, and effort of the other controls, with one execution per cell: C_{g_0} (no rebuild, reference); $\tilde{C}_{g_1} = M(C_{g_0}, g_1)$, for which the builder receives only the stored old strategy and g_1, not the archive; and $C_{g_1} = M(T, g_1)$. Post-revision valid-episode success is $6/39 = 0.154$ $[0.07, 0.30]$, $3/38 = 0.079$ $[0.03, 0.21]$, and $25/38 = 0.658$ $[0.50, 0.79]$; all three arms lie between 0.92 and 0.98 before revision. Paired two-sided Tango intervals: $\tilde{C}_{g_1} - C_{g_0} = -0.081$ $[-0.227, +0.050]$ ($n = 37$; no detected recovery, not equivalence); $C_{g_1} - \tilde{C}_{g_1} = +0.583$ $[+0.349, +0.744]$ ($n = 36$); $C_{g_1} - C_{g_0} = +0.486$ $[+0.249, +0.665]$ ($n = 37$), reproducing the direction of the registered control on a disjoint block. Five of 120 resolved cells are excluded (four transport, one model-side truncation). One extra objective-specific rebuild from the old strategy alone shows no detected recovery at this sample size, whereas rebuilding with the archive improves post-revision performance. The builder prompt head necessarily differs between the two inputs (a stored note versus an archive), so $C_{g_1} - \tilde{C}_{g_1}$ compares material plus its description, not evidence content alone. With 40 seeds, one block, and $k = 1$, the pilot resolves only effects of roughly 20pp or more.

Table 9: Old-policy interference after revision. Each paired contrast has its own complete-case denominator and Tango 95% interval.

Contrast	Estimate	95% interval	Registered interpretation
Standing instructions minus neutral	−11.3pp	$[-23.6, +1.2]$	No detected effect
Superseded minus standing instructions	+7.0pp	$[-5.4, +19.2]$	No detected effect

Table 10: Post-revision stage-wise read-outs on valid episodes. Each column is a marginal count over the arm's own valid denominator, here the 547 valid post-revision C_{g_0} episodes ($30/547$ artifacts state the condition), not the 557 built artifacts behind the builder-output share $31/557$ in Section 3.4. Columns are not nested: workers can execute the condition without the guidance stating it. The same frozen text criterion scores artifact and guidance; execution is the replayed runtime fact that the combiner fired immediately after preparation. Descriptive; blocks are not pooled.

Path (block)	n	Artifact states	Guidance states	Worker executes	Success
T (600-seed)	555	—	382 (0.688)	379 (0.683)	298 (0.537)
C_{g_0} (600-seed)	547	30 (0.055)	26 (0.048)	143 (0.261)	32 (0.059)

H.3 Old-policy interference and historical marking

A registered diagnostic places the identical archive after neutral prose, an original-objective strategy presented as standing instructions, or the same strategy marked superseded. The superseded header is the intervention; it is not length matched, and no condition supplies the correct rule.

Neither contrast establishes an effect, equivalence, demotion, or repair. Truncations concentrate in the superseded condition, further limiting interpretation.

H.4 Utilization audit

The audit asks whether the retained artifact and the planner output state the necessary prepare–combine ordering constraint and whether the worker executes it. It replays saved outputs and executions without model calls, using one frozen text criterion with an explicit undecidable category; every replayed episode reproduces the stored call count, firing count, and stop reason. Figure 4b summarizes the failure classification. All numbers are descriptive, use each block's own valid denominators, and never pool the 600-seed confirmatory block with the 80-seed builder-objective block.

Explicit statements across construction and planning. Applied to the builder output, the criterion finds the condition in $31/557$ C_{g_0} artifacts (0.056 $[0.039, 0.078]$), $45/149$ revision-aware artifacts (0.302 $[0.234, 0.380]$), and $53/76$ revised-objective artifacts (0.697 $[0.587, 0.789]$; 80-seed block). On the 547 valid post-revision C_{g_0} episodes, artifact and guidance both state the condition in 24, the artifact alone in 6, the guidance alone in 2, and neither in 515: the planner keeps the condition in $24/30$ artifacts that state it and adds it in $2/517$ that do not. Under the text criterion, the builder output rarely states the constraint explicitly. The archive is a transcript and is not scored by this matcher. Table 10 gives the post-revision stage-wise read-outs; its columns are marginal, not nested, because workers execute the condition without the guidance stating it in $127/547$ (C_{g_0}) and $93/555$ (T) episodes, a quantity that mixes matcher misses with unaided discovery and cannot be separated further.

How the condition is broken. Most non-firing combiner calls had a preparation call earlier but another call in between. Of 1,665 post-revision combiner calls under C_{g_0}, 0.112 $[0.097, 0.128]$ immediately follow preparation, 0.264 follow preparation with only report calls in between, 0.343 follow preparation with another dispatched operation in between, and 0.281 have no prior preparation; the corresponding shares of 2,005 calls under T are 0.333, 0.162, 0.286, and 0.219. Because any dispatched call closes the window, a report issued to confirm preparation defeats the condition while the combiner still returns `ok`. The terminal states do not support early stopping as the main explanation of post-revision C_{g_0} failures: $25/515$ (0.049 $[0.033, 0.071]$) end with both movers on

Table 11: Successes over episodes by guidance text (frozen criterion) and whether the combiner fired immediately after preparation, all valid post-revision episodes. False gate = guidance states a parity condition that is false for every instance. Descriptive; blocks are not pooled.

Guidance	Executed	T ($n = 555$)	C_{g_0} ($n = 547$)
States condition	yes	$244/286 = 0.853$	$12/16 = 0.750$
States condition	no	$3/96$	$0/10$
Does not state	yes	$41/74 = 0.554$	$15/121 = 0.124$
Does not state	no	$1/56$	$5/379$
False gate	yes	$5/13$	—
False gate	no	$0/17$	—
Undecidable	yes	$4/6$	$0/6$
Undecidable	no	$0/7$	$0/15$

target and at least one unused call, $13/515$ with at least four, and $354/515$ (0.687) exhaust the budget; the median episode uses $26/26$ calls in both arms. A further $95/547$ C_{g_0} and $36/555$ T valid post-revision episodes end at the harness turn cap (Appendix O).

The criterion can miss equivalent wording, and workers may discover the condition unaided, so the audit does not estimate mechanism shares and is not a control.

Denominators behind Figure 4b. The failed-episode shares depend on the denominator: the archive row "condition text, not executed" is $93/555 = 0.168$ of valid episodes, $93/257 = 0.362$ of failures, and $93/165 = 0.564$ of failures excluding the third row. That third row ("fired, still failed") is the runtime class "combiner fired after preparation yet the episode failed" ($85/92$ for T, $116/131$ for C_{g_0}) together with 7 and 15 text-undecidable episodes; its share also differs by path (0.358 of archive failures against 0.254 of strategy failures), so failure-only shares should be read both with and without it.

Full-sample cross-table. Table 11 crosses guidance text, execution, and success on all valid post-revision episodes. Executing the condition is close to necessary for success on both paths ($4/176$ and $5/404$ successes without it, all instances whose conditionally updated variable started on target), and on the archive path, execution accompanied by a matched statement in the guidance has a higher success rate than execution without one (0.853 against 0.554). These are associations in recorded outcomes; nothing here was randomized.

Terminal-state decomposition. Replaying the terminal states of the same episodes, with Q for within the revised 26-call budget, μ for both directly updated variables on target, and K for the conditionally updated variable on target, no valid episode exceeds the budget because the runtime refuses over-budget calls. For T ($n = 555$): $Q(1-\mu)$ 152 (0.274), $Q\mu(1-K)$ 105 (0.189), success 298 (0.537). For C_{g_0} ($n = 547$): 318 (0.581), 197 (0.360), 32 (0.059). The old-objective variable subset is met within the revised budget in $403/555 = 0.726$ archive episodes against $229/547 = 0.419$ strategy episodes; this is not the original g_0 success rate, whose budget was 14, and the relative composition of failures is similar across paths, so the difference is in level, not in kind.

H.5 Cross-model revision with fixed representations

The reader-only replication reuses the primary model's compressed representation byte-identically and changes only the planner and worker to `MiniMax-M3`. Table 12 summarizes the common complete paired set.

The ranking reversal survives this reader change; the replication makes no gate claim. The provider does not honor the requested reasoning-effort setting, so matched request shape is not matched compute.

Table 12: Fixed-representation second-reader comparison. Contrasts are paired Tango 95% intervals on the same $n = 109$ seeds.

Contrast	Estimate	95% interval
Original-objective minus archive, before revision	+26.6pp	$[+15.0, +37.7]$
Archive minus original-objective, after revision	+12.8pp	$[+5.9, +21.0]$

H.6 Open-weight decision read-out and gradient diagnostic

Two registered diagnostics use the same open-weight reader, `gemma-4-E2B-it` in bf16, with its audio and vision towers removed. The registration was frozen before any call on a registered seed. After launch and before any registered row was read, an offline check identified ten E39a seeds and five E39b seeds on which the scored decision does not cleanly separate g_1 or the generated instance repeats a state-variable name. The registered analysis retains these cases; excluding them in sensitivity analyses changes no branch or estimate reported below by more than 0.01 nats or 0.2pp.

Decision-level read-out (E39a). For each seed, the reader is placed at one scripted point on the post-revision held-out instance. A fixed prefix reports the state, tests the two direct-update operations, executes one adjacent preparation–combine pair, verifies the resulting state change, sets the two directly updated state variables to target, and invokes preparation again. The reader receives the stored representation, the revised task and announcement, and this prefix as alternating turns. We define m as the log-probability of the combiner call minus that of the state-reporting call, summed over the assistant turn's content tokens; the end-of-turn marker is not scored. The reader's greedy next call provides a behavioral read-out.

Two archive interventions probe whether the interrupted preparation–combine example carries the difference. T_{swap} reorders that segment so preparation and combine are adjacent, preserving the calls and their position while replacing the interrupted example with a second successful firing example. T_{cut} removes the segment entirely. Continuous contrasts use seed-level percentile bootstrap intervals with 5000 resamples; the paired greedy-choice contrast uses a two-sided Tango interval.

Table 13: Open-weight reader at the post-revision decision (E39a). m is the combiner-versus-report log-probability margin in nats. The greedy row reports the paired difference in the fraction of seeds whose next call is the combiner.

Contrast	Estimate	95% interval	n
$m(T) - m(C_{g_0})$	+2.01	$[+1.70, +2.32]$	507
greedy combiner, $T - C_{g_0}$	+21.1pp	$[+16.9, +25.4]$	507
$m(C_{g_1}) - m(C_{g_0})$, builder-objective block	+1.93	$[+1.23, +2.65]$	74
$m(T) - m(C_{g_1})$, builder-objective block	+0.47	$[-0.26, +1.22]$	74
$m(T) - m(T_{\text{swap}})$	−0.01	$[-0.04, +0.03]$	507
$m(T) - m(T_{\text{cut}})$	−0.09	$[-0.16, -0.01]$	507
$m(T) - m$(no material)	−1.53	$[-1.87, -1.19]$	507
$m(C_{g_0}) - m$(no material)	−3.53	$[-3.89, -3.17]$	507

The first contrast is corroborating diagnostic evidence rather than a replication: both the reader and the endpoint differ from the episode-level study. The greedy combiner counts are $199/507$ from T and $92/507$ from C_{g_0}; a report is instead the greedy next call on $284/507$ and $398/507$ seeds, respectively. Neither archive intervention supports locating the difference in the interrupted example: swapping it produces essentially no change, while deleting it slightly raises the margin. A gradient-times-input attribution is consistent with this null localization but is only heuristic: normalized by token share, the complete, direct-update-only, and interrupted episodes receive 1.44, 1.69, and 0.90 times their respective token shares of attribution mass.

The no-material condition is a sanity reference rather than an operational baseline. Both T and C_{g_0} lower this particular combiner-versus-report margin relative to no material, and without material

the reader chooses neither candidate operation on $161/507$ seeds. Across the stored post-revision outcomes of the primary study, m is also descriptively higher on seeds where the corresponding path succeeded: -0.75 versus -1.05 from T, and -2.62 versus -2.91 from C_{g_0}. This association compares outcomes produced by a different reader and is not a mechanism estimate.

Gradient direction (E39b). A separate registered diagnostic uses 200 fresh seeds. A scripted source operator generates 24 episodes per seed in which the preparation–combine step is adjacent, interrupted by a state-reporting call, or omitted, with occasional off-by-one direct updates. The original and revised verifiers define the accepted episode sets. Five seeds have an empty accepted set under the registered analysis rule, leaving 195 analyzed seeds.

For each accepted set D, we compute the gradient of mean action-token negative log-likelihood within the span of rank-8 LoRA-B adapters on the query and value projections. We then measure its cosine $a(D)$ with the gradient of the held-out margin m. The registered qualification requires $a(D_{g_1}) > 0$: the first-order direction induced by experience accepted under g_1 should raise the combiner-versus-report margin on the renamed held-out instance. This qualification fails, with $a(D_{g_1}) = -0.018$ $[-0.025, -0.012]$; $1133/1160$ of the g_1-accepted source episodes contain an adjacent preparation–combine step.

Because the qualification fails, the registered comparison between g_0- and g_1-accepted experience is not interpreted. The result establishes only that, for this model and this LoRA-B subspace, the probed first-order update direction from g_1-accepted source episodes does not raise the held-out decision margin across renaming. It therefore does not test whether selecting fine-tuning data under the current objective would reproduce the behavioral acceptance blind spot. The registered fine-tuning comparison was not run.

I SUPPORTING DETAIL FOR THE RESULTS

This appendix reports the experience-transfer prerequisite, calibration studies, and controls on source generation that complement Sections 2–4.

I.1 TRANSFER-FORMAT VALIDITY, PAIRED INFERENCE, AND TRANSFER SCOPE

The primary representation comparison assumes that retained experience can influence later execution. A separate study tests this prerequisite on a fixed renamed-instance workload by varying how experience is supplied. The nine formats include no experience, neutral text, short written summaries, and recorded episodes. Unlike the primary comparison, this study changes the experience-transfer format rather than holding one source bundle fixed.

The archive outperforms no experience and neutral text on two seed blocks. Formats that retain episode structure, as well as the short strategy, transfer useful procedure (Figure 9).

Success at the method level counts failed construction against the assigned method. Success conditional on construction excludes those failed builds. Both rates exclude transport faults and model truncations under the frozen rules. Figure 9 displays both short-strategy denominators and the marginal intervals; paired archive contrasts are reported separately from those bars.

The two blocks retain separate analyses. Cross-condition exclusion spreads exceed the registered warning threshold, so pairing cannot remove all selection risk.

Table 14 separates surface, structural, and rule changes. The new-structure arms are near the floor and lack a calibrated true-rule control. Their registered recalibration varies operation count, state dimension, and call budget separately, accepting a setting only when true-rule success is at least 0.70, no-experience success at most 0.20, and the dynamic-range interval has a positive lower bound. No setting yet supports a cross-structure result. The changed-rule archive arm has six valid episodes and an interval containing zero, so it also supports no transfer conclusion.

Table 14: Different instance changes test different forms of transfer. Counts are successful/valid episodes for no experience and archive. Differences retain the originally reported 95% intervals. Mean calls are budgeted tool calls, not API calls. The primary renamed-instance result appears quantitatively in Figure 9 and descriptively in Figure 8.

Instance change	What changes and what a gain could show	Successes: none; archive	Difference; mean calls
New target only	Same family and surface names; concrete fact reuse, not a transfer claim	$0/11$; $12/12$	$+100.0$pp $[+65, +100]$; 16.1
Renamed operations	Same family and operation rules; new names and constants test role-level reuse	Primary condition: quantitative bars in Figure 9, descriptive grid in Figure 8, and Appendix I.1; mean calls 24.3	
New family structure	Same operation regularity in a new structure; abstract-rule reuse remains uninterpretable before recalibration	$0/11$; $1/9$	$+11.1$pp $[-16, +44]$; 26.0
Changed operation rules	The regularity changes; reuse of an identification procedure is unresolved	$3/11$; $4/6$	$+39.4$pp $[-8, +69]$; 22.2

Table 15: Archive-only reference calibration before and after objective revision on 24 fresh seeds. Each objective has 24 valid episodes, with no exclusions and one attempt per episode.

Objective condition	Call budget	Valid n	Successes	Rate, Wilson 95%	Exact 95%
Before revision: movers only (g_0)	14	24	23	0.958 $[0.80, 0.99]$	$[0.79, 1.00]$
After revision: full state (g_1, announced)	26	24	14	0.583 $[0.39, 0.76]$	$[0.37, 0.78]$

Within-archive pairing on the same 24 seeds, written as (before revision, after revision): (success, success) 14; (success, failure) 9; (failure, success) 0; (failure, failure) 1.

I.2 Archive-only reference calibration

Table 15 supplies the archive marginals and within-path pairing used for planning. Nine seeds succeed before revision and fail afterward, with no reverse discordance: $B_0 - B_1 = +0.38$, with exact two-sided $p = 0.004$ on discordant pairs.

I.3 Eight-seed objective-revision pilot

The pilot shows that factual recall and current behavior need not track one another. Both compressed conditions pass the nominal pre-revision screen despite low factual scores and trail the archive after revision; the revision-aware condition also trails the original-objective condition. The sample lies below the registered gate's structural floor, and its one descriptive bootstrap interval excluding zero depends on the archive reference rate.

I.4 Verifier-label and source-operator confounding

The original faulty-versus-correct comparison changes both the verifier label and the source briefing. Table 17 gives its per-condition rates; Figure 4c,d plots its contrasts together with two registered controls that isolate one factor at a time.

In the stratum of source instances accepted under both briefing conditions, the unbriefed-minus-briefed contrast in held-out stale-policy rate is $+36.7$pp $[25.8, 47.6]$. The briefing includes execution advice and changes the generated trajectory, so this is a total effect through source experience, not a comparison with T fixed. The label-only contrast is $+2.6$pp $[-9.2, 14.4]$; no effect is detected on either the stale-policy rate or true success. Rejected source sessions are outcomes rather than exclusions.

Table 16: Eight-seed objective-revision pilot. "Before" tests the movers-only objective; "after" tests the announced full-state objective. Factual columns score original-objective and revision questions. Rates use the original valid denominators ($3/48$ trials excluded). API cost uses all eight assigned trials per representation and includes construction, probes, planning, and execution under both objectives. Character counts are valid-only means.

		Factual answers		Task success			
Representation	Valid n	Original objective	Revised goal	Before B_0	After B_1	Characters	USD / assigned trial
No memory	8	0.00	0.00	0.62	0.00	0	0.0211
Full archived trajectory	8	1.00	1.00	1.00	0.50	1815	0.0285
Verified observations	7	0.21	0.00	0.86	0.00	316	0.0456
Step plan	7	0.21	0.10	0.57	0.00	212	0.0453
Original-objective strategy	7	0.04	0.07	1.00	0.29	990	0.0309
Revision-aware strategy	8	0.00	0.06	1.00	0.12	924	0.0284

The full trajectory is the reference. Original-objective and revision-aware strategies pass only the nominal pre-revision screen; the other conditions fall below its margin. Verified-observations B_0 retains the pilot's original cell-wide $6/7$ convention; under the later stage-specific convention the same records give $7/8$. Token counts are unavailable.

Table 17: Per-condition rates in the faulty-verifier transfer study. True success and the stale procedure are read from runtime state; combiner execution is counted inside the executor.

Source label	valid n	true success	95% CI	stale-policy rate	95% CI
ϕ (faulty)	9	0.11	$[0.02, 0.44]$	0.78	$[0.45, 0.94]$
ϕ' (correct)	20	0.65	$[0.43, 0.82]$	0.20	$[0.08, 0.42]$

I.5 Planner sensitivity in recovery

A separate diagnostic asks whether planner effort limits recovery from the archived trajectory. Three planner settings produce broadly overlapping recovery intervals; higher effort mainly increases exclusions rather than valid recoveries.

J Evidence add-back

This diagnostic tests whether supplying revision-relevant evidence restores factual answers or held-out behavior. The evidence block restores factual probes, but with only a handful of pairs and one binary episode per cell, no behavioral interval excludes zero.

The nearly verbatim, report-bracketed add-back is slightly larger than the archive itself, so it is not a targeted addition. Planner-budget comparisons are unresolved.

K Reading economics and scale limits

This appendix gives the cost model for rereading an archive and a descriptive view of the experience-transfer formats.

K.1 A cost model for rereading

The reconciled off-peak snapshot is \$0.003 per million cache-hit input tokens, \$0.15 per million cache-miss input tokens, and \$0.60 per million output tokens. For the following expression, p_{hit}, p_{miss}, and p_{out} use USD *per token*. If a strategy reconstruction reads H input tokens, obtains a cache-hit fraction $q \in [0, 1]$, and emits G_P billed output tokens, its API charge is

$$c_{\mathrm{reread}} := H\big[q\, p_{\mathrm{hit}} + (1 - q)p_{\mathrm{miss}}\big] + G_P p_{\mathrm{out}}. \tag{6}$$

Table 18: Add-back effects relative to token- and format-matched neutral filler, paired by seed. Of 16 registered seeds, only 6 were executed before interruption. Paired n is the actual sample for each contrast. B_1 is success after the announced full-state objective; revision probes measure factual recall, including source-instance facts.

Base representation	Read-out	paired n	difference	95% CI
original-objective	revision probes	6	$+0.97$	$[+0.92, +1.00]$
original-objective	B_1, standard planner	6	-0.17	$[-0.50, +0.00]$
original-objective	B_1, high planner	6	$+0.33$	$[+0.00, +0.67]$
revision-aware	revision probes	6	$+0.89$	$[+0.83, +0.94]$
revision-aware	B_1, standard planner	6	$+0.17$	$[-0.33, +0.67]$
revision-aware	B_1, high planner	5	$+0.40$	$[+0.00, +0.80]$

Table 19: Unscored API microbenchmarks. A judgment request reads a 213,331-token trace ($n = 1$ cold, $n = 3$ warm); the comparison request produces 20,000 output tokens from a 97-token prompt.

	USD	seconds
judgment request, cached	0.00085	3.0
judgment request, uncached	0.03206	4.5
output-only generation (2×10^4 tokens)	0.01201	64.8

The first term is reading; the second includes new reasoning and completion output. The expression does not include tools, verification, acquisition of new experience, or cache maintenance. The artifact decision in Appendix D also depends on reuse and revision risk.

One of three warm reads reaches the length limit with empty answer text. The output-only request is neither executed nor scored, and these partial bills do not order reading against recovery.

K.2 Descriptive transfer-format grid

For plotting, let x be recorded USD per assigned task and y the success rate. The descriptive score is $L = y - 0.25(\log_{10} x - \log_{10} 0.01)$, with edges at 0.20, 0.45, and 0.75. Cost differences reflect incidental worker deliberation, not a measured cost-to-success curve. For the annotated pair, billed output means provider-reported completion tokens including reasoning; the $5.9\times$ full-episode/continued-session gap is descriptive, not a causal cost attribution.

L Cost accounting and long-context judging

This appendix documents how recorded usage becomes cost and reports the long-context judging scans.

L.1 Recorded usage and denominator scope

Cost accounting joins each result to recorded usage and the price window at call start, then applies Equation 6. Completion usage includes reasoning; missing usage remains unknown rather than zero. These aggregates do not support cell-level resampling or a new paired cost comparison.

The confirmatory run has incomplete usage because a transport-level billing interruption produced unpriced attempts. Transfer-format and pilot totals include construction, probes, planning, and behavioral execution, while unrelated calibration calls are excluded. Character counts cannot substitute for missing Token measurements. In the fully measured archive calibration, output dominates cost, showing why cheap input reading alone does not make complete agent computation cheap.

L.2 Transfer-format cost and success views

The transfer-format bars hold one renamed-instance workload fixed and group conditions by what the agent receives. They show valid-episode success with marginal Wilson intervals and retain

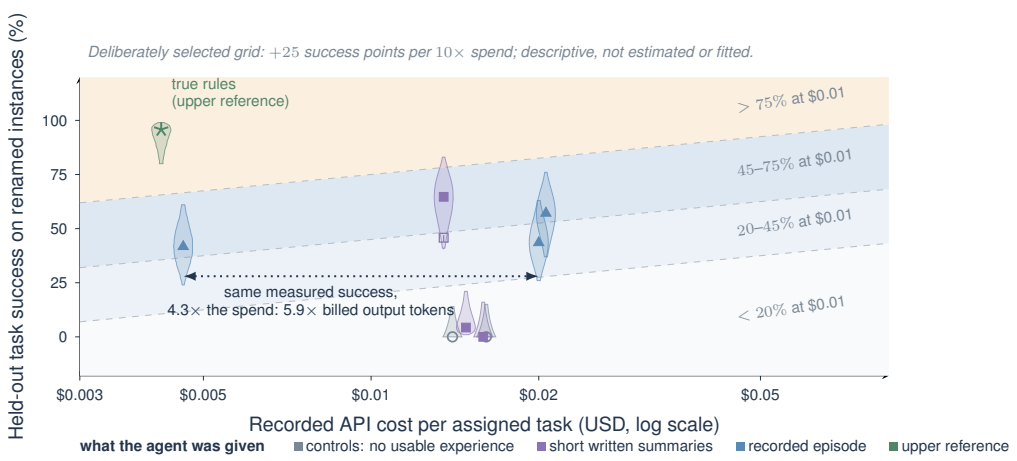


Figure 8: Nine experience-transfer formats on one fixed renamed-instance workload. Vertical extents show marginal Wilson 95% intervals, and cost divides recorded usage by all 24 assigned tasks. The diagonal bands use a chosen 0.25 success-per-decade slope to separate conditions visually; they are not fitted or validated scaling relationships. Conditions vary supplied experience, not one method's budget or transfer distance.

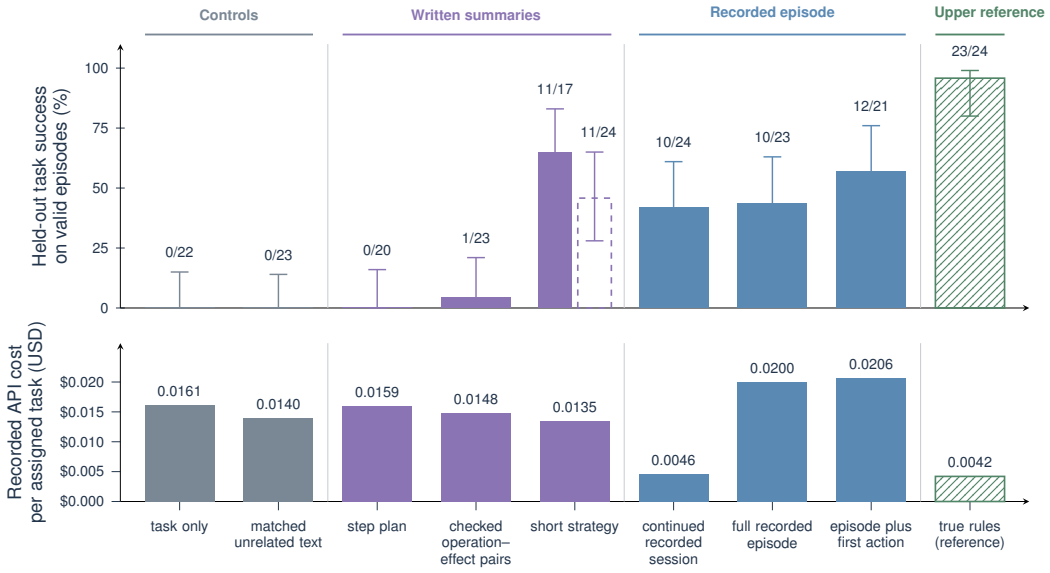


Figure 9: Success and recorded cost for nine experience-transfer formats at one fixed renamed-instance transfer workload. The upper panel gives held-out success on valid episodes with marginal Wilson 95% intervals; solid and dashed short-strategy bars retain the execution-conditional and method-level denominators. The lower panel divides recorded usage by all 24 assigned tasks per condition. What the agent receives changes success, while cost does not order success here. Intervals are marginal; the surrounding text defines the denominators and missing usage.

both method-level and construction-conditional denominators for the short strategy. Cost divides recorded usage by all assigned tasks, including construction attempts; missing usage stays unknown.

The operation–effect-pairs condition is near the floor, so neither figure reports cost per success: its reciprocal would not be a meaningful efficiency read-out. Several conditions also share the same recorded format length, preventing length from separating them.

Table 20: Errors when judging code against explicit specifications. Fixed-filler costs are medians per judgment; execution-history costs are reported per-judgment amounts. Correctness rates retain source precision. False-accept and false-reject denominators count their respective opportunities; unparseable answers are separate. The five-record batch is reported separately from the single-judgment context-length conditions. Fixed-filler rows list 62 judgments per condition; these are 31 items judged twice, so the independent unit is the item and the descriptive item-level intervals use $n = 31$.

Context / condition	Judgments	Unparsed	Accuracy	False accepts	False rejects	USD/judgment
Code judging with a fixed filler prefix						
No filler	62	0	1.000	0/22	0/40	0.000200
Approximately 4K tokens	62	0	0.984	0/22	1/40	0.000218
Approximately 50K tokens	62	0	0.968	0/22	2/40	0.000351
Approximately 200K tokens	62	0	1.000	0/22	0/40	0.000750
Code judging with recorded execution histories						
Code + specification, 1799 tokens	88	0	0.99	0/56	1/32	0.000176
Complete trace, 4325 tokens	88	0	0.99	0/56	1/32	0.000230
Target after history, 305369 tokens	88	0	0.99	0/56	1/32	0.001169
Five records per request	80	5	0.96	2/44	1/31	0.000746

L.3 Task construction and error interpretation

The fixed-filler scan places each of 31 small-function code items after a fixed prefix and judges it twice at each of four lengths with low reasoning effort. The execution-history scan uses recorded multi-attempt traces, pooling two effort settings and repeated judgments per item. Its target is always at the end; reconstructed conflict cases use genuine trace text but are not episodes that occurred intact. Neither scan tests middle-position retrieval, near-million-token behavior, or long-horizon recovery. Context construction and length vary, not dollar budget.

A false accept passes a submission that fails the corrected evaluator; a false reject rejects one that passes. Errors concentrate in one code item rather than increasing monotonically with context length, and near-ceiling accuracy precludes a general degradation claim.

The independent unit in the fixed-filler scan is the code item rather than each repeated judgment. Item-level uncertainty is wider than the naive judgment-level interval, while the history scan lacks the records needed for a new cluster-aware interval. Batching several records costs more without an observed quality gain.

M Measured billing and cache boundaries

This appendix records the measured cache and billing behavior behind the cost model in Appendix K. Cache behavior depends on prefix matching, minimum length, block granularity, invalidation, and lifetime. In the measured long-input workload, cache reuse reduces billed input price far more than time to first token. Appending preserves the prefix hit, whereas prepending revised instructions invalidates it. Concurrent identical requests do not share entries within a batch, and short prompts do not reach the cache-hit tier.

No decay appears within the measured window, but the upper lifetime is unknown and best-effort disk-cache residency is not an architectural guarantee. These observations cover one provider's off-peak tier.

N Cross-model transfer-format replication and population scope

This appendix tests whether the transfer-format result replicates with a different model. This end-to-end replication changes source producer and target reader together, unlike the fixed-representation second-reader result in Appendix H.5.

Source screening precedes arm assignment and removes a failed seed from every arm. It therefore defines the population of instances the tested model solves unaided at this operating point without differential arm selection. The screening rate is a capability measurement, not a transfer-format effect.

The registered transfer-format comparison includes no experience, archive, length-matched neutral text, and a true-rule positive control. The archive contrasts fail the registered meaningful-effect criterion while the positive control succeeds. The turn guard is non-binding; the effective limit is the held-out tool-call budget.

Four limitations remain: the provider ignores the requested reasoning-effort setting; producer and reader change together; screening restricts inference to source-solvable instances; and the replication covers renamed instances rather than the other transfer conditions.

O EXECUTION VALIDITY

This appendix states the rules that decide which outcomes are valid. The frozen protocol fixes the observation unit, sample size, conditions, exclusions, and one initial attempt per cell. Exclusions cover protocol failure, model truncation, empty artifacts, unreconstructable calls, and responses outside the frozen model condition. Execution batches cross seeds with all scheduled representations to reduce representation–time confounding.

One transport-only remeasurement is allowed only when the original attempt produces neither usage nor a model-side observation. Mixed or model-side failures are not retried, missing usage is never imputed as zero, and valid model failures remain behavioral zeros. This rule materially changes the complete-case sample but not the estimator, gate, or exclusion definition.

The worker's runaway-loop turn guard of 24 assistant turns is distinct from the verifier's tool-call budget and was not declared as a budget in the registration. Because the provider emits roughly one tool call per assistant turn, the guard binds first in some post-revision episodes: an offline replay of the stored calls finds that it ended $95/547$ (0.174) valid post-revision C_{g_0} episodes and $36/555$ (0.065) valid T episodes with one to five tool calls unused (median two); on the 80-seed builder-objective block the counts are $14/78$ (C_{g_0}), $3/80$ (T), and $1/76$ (C_{g_1}). These episodes are not exclusions; they stay in the valid denominators as failures, and the two arms are affected unequally. A deterministic bound shows that the asymmetry cannot reverse the post-revision direction: counting every turn-capped C_{g_0} episode as a success would raise its marginal to at most $(32 + 95)/547 = 0.232$, while counting every turn-capped T episode as a failure leaves $298/555 = 0.537$ unchanged, since those episodes already are failures. This is a bound on the stage marginals, not an estimate of what the unused calls would have produced. Combined with the missing-outcome fill-in, the joint worst case still leaves the post-revision difference at least $118/600 = 19.7$pp over all assigned seeds (Equation 5).

P PRE-REGISTERED STOPPING CONDITIONS

The registration named results that would stop or narrow the main conclusion. The control built for the revised objective succeeds at both stages on a fresh disjoint block, activating the registered branch that limits the conclusion to the tested builder or acceptance criterion rather than compactness (Appendix H). The other branches remain: closure under a larger planner budget would support a utilization limit, while no add-back advantage over neutral filler would withdraw the localized-information-loss explanation without erasing the behavioral result.

Q FOLLOW-UP DESIGNS AND UNMET PREREQUISITES

The following designs address revision cost, judgment, and task continuity; their intended behavioral comparisons remain unrun.

Q.1 CONTROLS MOTIVATED BY THE OBJECTIVE-REVISION RESULTS

Each of the following is unrun and has no result. (a) A scope stamp appended to C_{g_0}, stating which state variables the original acceptance checked and that the others were unevaluated, against length-matched neutral text, post-revision only. (b) The scope stamp plus a pointer to the archive episode that holds the contrast probe. (c) The archive path under a 440- or 660-word planner cap, to test whether the C_{g_1} advantage is a guidance-length effect. (d) Generator ablations of the three instrument switches in Section 2: an unarmed combine that errors instead of returning `ok`, a preparation flag exposed in the observation, and sampling which state variables the acceptance checks.

Q.2 COST AND CONTEXT COMPARISONS

Any economic comparison must include input, output, tools, behavioral verification, new rollouts, and cache maintenance, with either quality matched at equal cost or cost matched at equal quality. The behavioral archive and long-context pricing workloads differ greatly in scale, so the current small-archive result establishes no long-context economic benefit.

Proposed reconstruction arms would compare full rereading, selective exclusion, appended correction, and compressed context while holding world, objective, and operation semantics fixed. An equal-dollar model comparison would additionally require zero-context capability controls. Both require the behavioral recovery contrast specified in Appendix D and matched-quality accounting before execution.

Q.3 EVIDENCE-STATE AND INTERRUPTION TASKS

The evidence-state generator constructs histories whose revised verdict can be `PASS`, `FAIL`, or `INSUFFICIENT_EVIDENCE`. Offline audit shows all labels are reachable after balancing visible features, but the engineering pilot tests only task execution, and its permissive budget creates a ceiling.

The interruption design asks whether an agent can resume a tool-use task after unrelated inserted work while preserving the operation-role mapping. Development calibration exhausts its registered budget ladder without authorizing the comparison. A new registration and instrument, rather than more samples under the failed calibration, are required.

Q.4 GRAPH-STRUCTURED CONTEXT

Graph-update comparisons require answers to depend on the artifacts being revised (Figure 10). The preflight compares supplied artifacts with neutral placeholders on old-answer questions. Artifacts improve accuracy, but the placeholder condition exceeds the registered qualification ceiling, so propagation, selective repair, full rebuild, oracle recovery, and unaffected retention were not tested. A later comparison needs a qualifying workload and must measure verified recovery, unaffected-task degradation, and total construction, update, retrieval, and rerendering cost.

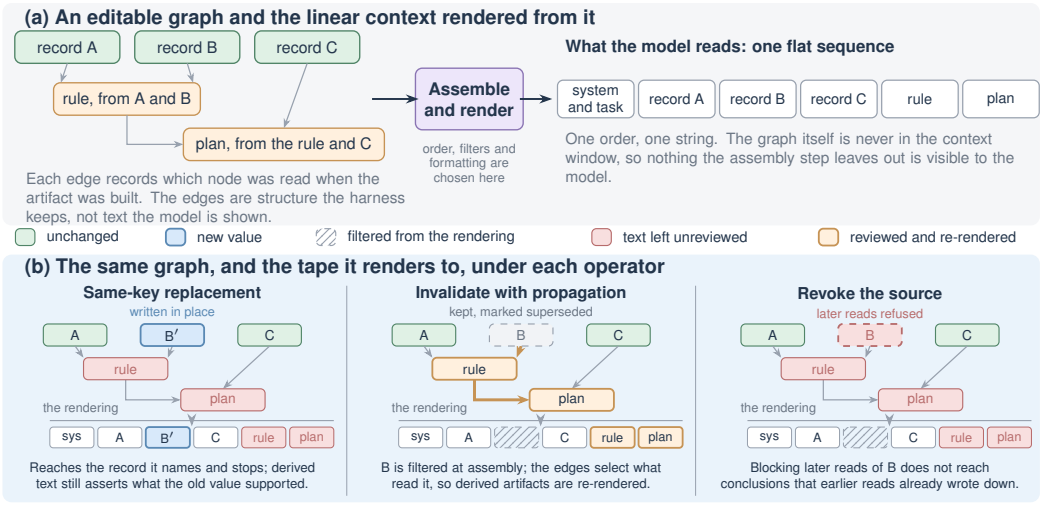


Figure 10: Proposed graph-context write-back. (a) An editable provenance graph is rendered as the single sequence the model receives. (b) Replacement, propagated invalidation, and revocation yield different renderings.