FREEZING OPPORTUNITY: MATCHED-UNIT ATTRIBUTION FOR SHARED-BUDGET AGENT MEMORY CONTROL

Anonymous authors
Paper under double-blind review

ABSTRACT

An agent-memory policy allocates a shared budget across retrieval, verification, summarization, and writing, so each decision changes which information actions remain affordable later. Policy comparisons become confounded when prompts, tools, candidate pools, admissible actions, or accounting rules also change. We introduce opportunity-matched attribution, a protocol that fixes this interface contract and matched evaluation units, then resolves a policy gain through four preassigned contrasts: exact-parity deployment, paired same-information control, shared-training-trace temporal control, and randomized feasible-action intervention. The resulting estimands distinguish gains due to controller organization from gains due to information access, temporal representation, or score ordering. LedgerGate instantiates the protocol as a support-aware five-action scheduler whose role-typed state limits aliasing, reliability particles update the value of expensive information, reserve-aware scoring allocates one token/time ledger, and deterministic fallback governs weak-support states. On LOCOMO v2, ToolBench-POMDP, and SWE-verify, LedgerGate improves tight-budget success over a prespecified MLP by $+2.2/+3.7/+2.4$ percentage points. At the medium budget, it remains within 0.5 points of Always-retrieve while reducing memory-action tokens by a 31.7% cross-benchmark median. Across 20 matched seeds, LedgerGate retains $+1.6/+1.8/+1.2$ points over the same-information controller and $+0.9/+1.1/+0.8$ over a parameter-matched GRU, with positive seed-first 95% intervals on every benchmark. Feasible-state interventions reach 82–87% sign-correctness for `RETRIEVE` and 74–79% for `VERIFY`, the actions carrying most resource pressure. Opportunity matching thereby converts bundled agent-system comparisons into controlled policy attribution and makes competing explanations empirically separable.

1 INTRODUCTION

Long-lived agents repeatedly choose whether to retrieve evidence, verify a claim, compress context, write a durable record, or continue without spending. These actions draw from the same token and tool-time budgets, and an action taken now changes the feasible continuation of the episode. Memory control is therefore a sequential allocation problem over a coupled set of information opportunities. A two-point success gain can be consequential in this regime when it is achieved under an identical resource envelope and repeated across thousands of matched episodes.

The central measurement challenge is that an agent policy is often changed together with the system it controls. Adaptive retrieval and memory systems may expose different prompt signals, retrieval candidates, tools, write paths, or feasibility rules (Jiang et al., 2023; Asai et al., 2024; Jeong et al., 2024; Du et al., 2025; Yang et al., 2026; Zhang et al., 2026a). Their end-to-end outcomes combine two effects: the quality of policy decisions and the opportunity supplied to those decisions. Matching the opportunity set identifies the source of improvement and supplies the cross-policy comparison that within-system component analysis leaves unresolved.

We introduce opportunity-matched attribution to isolate the policy effect. The protocol fixes an interface tuple containing prompts, tools, retrieval candidates, actions, budgets, feasibility rules, and

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.

evaluation harness, then pairs task instances, environment randomness, model-sampling seeds, and initial memory. Four estimands are assigned before held-out evaluation. Exact-parity deployment measures the complete policy effect; a paired same-information controller tests the role of state organization; controllers trained on the same behavior traces test temporal representation; and feasible-action randomization measures whether the deployed score orders realized action gains. The sequence converts common explanations for a controller gain into matched empirical contrasts under one opportunity contract.

LedgerGate is the scheduler used to instantiate this protocol. It groups pre-action observations by decision role, tracks retrieval and verifier reliability with lightweight particles, compares five actions through reserve-aware cost-adjusted advantages, enforces token and time feasibility at decision time, and switches to deterministic fallback under low overlap or elevated surprise. These elements form one allocation mechanism: role typing reduces state aliasing; reliability updates change the value of expensive calls; continuation reserves expose the future cost of current spend; and support monitoring keeps extrapolation from governing scarce resources. Flat-semantic, Trace, and parameter-matched GRU controllers isolate these commitments while receiving the same ex ante opportunity.

Across LOCOMO v2, ToolBench-POMDP, and SWE-verify, LedgerGate improves tight-budget success over the pre-specified MLP by $+2.2/+3.7/+2.4$ percentage points and preserves medium-budget success with a 31.7% median reduction in memory-action tokens. Over 20 matched seeds, every benchmark retains a positive seed-first interval against same-information and parameter-matched temporal controls. Removing the learned score costs $4.1/4.5/3.2$ points across the three workloads, while genericizing the role-typed state costs $2.5/2.1/1.7$. Randomized interventions independently verify action ordering where retrieval and verification exert the greatest budget pressure.

The paper makes three technical contributions:

- opportunity-matched attribution, which defines a fixed interface contract, matched evaluation unit, and four pre-assigned estimands for separating policy quality from information access, temporal representation, and action ordering;
- LedgerGate, a support-aware shared-budget scheduler that integrates role-typed state, dynamic reliability, continuation reserves, cost-adjusted action values, feasibility masks, and deterministic fallback into an inspectable allocation path; and
- a three-benchmark study combining complete-episode effects, seed-first uncertainty, same-information and temporal controls, cross-benchmark component interactions, resource accounting, and randomized action-level validation.

Together, they provide a controlled basis for comparing memory, tool-use, and information-acquisition policies that share a constrained action interface.

2 Opportunity-Matched Attribution and LedgerGate

2.1 Opportunity contract and matched estimands

Let the controlled interface be $\mathcal{I} = (P, T, R, \mathcal{M}, B, F, E)$: prompts P, tools T, retrieval pool R, action menu $\mathcal{M}$, budget ledger B, feasibility rule F, and evaluation harness E. A matched unit u fixes the task instance, environment seed, model-sampling seed, and initial memory state. Every controller receives the same observation schema, sources, pre-action timing, and matched initial condition from $\mathcal{I}$; later observations may differ only through the controller's preceding actions, which are part of the policy intervention. For outcome Y, the protocol assigns four estimands:

$$\begin{aligned}
\Delta_{\text{dep}} &= \mathbb{E}_u[Y(\pi_{\text{LG}}, u) - Y(\pi_{\text{MLP}}, u)], \\
\Delta_{\text{info}} &= \mathbb{E}_u[Y(\pi_{\text{LG}}, u) - Y(\pi_{\text{Flat}}, u)], \\
\Delta_{\text{temp}} &= \mathbb{E}_u[Y(\pi_{\text{LG}}, u) - Y(\pi_{\text{Temporal}}, u)], \\
g_m(s) &= \mathbb{E}[Y \mid do(m), s] - \mathbb{E}[Y \mid do(\texttt{NOOP}), s].
\end{aligned}$$

Stage 1 estimates the complete exact-parity deployment effect; Stage 2 pairs a controller that sees the same semantic slots; Stage 3 fits temporal policies to the same logged training traces and deploys them on matched units, with each policy generating its own action-conditioned continuation; Stage 4

Table 1: Pre-assigned opportunity-matched attribution map. The interface $\mathcal{I}$ and evaluation unit u remain matched throughout; each stage changes one comparison object and isolates a distinct explanation for policy performance.

Stage	Contrast	Additional shared object	Attribution target
Deployment	LedgerGate vs. MLP	complete ex ante opportunity	complete policy effect
Same information	LedgerGate vs. Flat	semantic slots and native signals	state organization and reliability
Temporal	LedgerGate vs. Trace / GRU	behavior-training traces	temporal representation
Action	forced m vs. `NOOP`	feasible state and continuation randomness	score ordering $g_m(s)$

randomizes one feasible action to test whether the deployed score orders $g_m(s)$. Estimands and inferential roles are fixed before held-out evaluation. Transfer, latency, and robustness are named stress views rather than additional protocol stages.

Identification statement. With $\mathcal{I}$, u, and exogenous evaluation randomness paired, the outcome difference estimates the complete effect of controller decisions under common ex ante opportunity. The subsequent matched contrasts test semantic-slot access, temporal representation, and action-score ordering without changing the target interface.

2.2 Support-aware shared-budget scheduling

LedgerGate controls $\mathcal{M} = \{\texttt{NOOP}, \texttt{RETRIEVE}, \texttt{VERIFY}, \texttt{SUMMARIZE}, \texttt{WRITE}\}$. Its pre-action state s_t combines remaining budgets and a role-typed sketch with four groups: constraint identity, missing-information status, benchmark-native reliability evidence, and episode phase. The sketch has 12–18 effective dimensions across the primary benchmarks; $K{=}16$ particles maintain latent retrieval and verifier reliability from prior tool outcomes. Role typing changes state organization, not information access: Flat-semantic receives the same semantic slots and native failure signals as LedgerGate.

Five action-specific value heads are fit from behavior-policy logs using doubly robust targets, advantage-weighted regression, recorded propensities, and weight clipping; each head uses five bootstrap members for action-value uncertainty. Particle k carries reliability state z_t^k and weight w_t^k, updated before decision t from the preceding tool outcome o_{t-1} by

$$w_t^k \propto w_{t-1}^k p(o_{t-1} \mid z_t^k, m_{t-1}), \qquad \hat{Q}_\theta(s_t, m) = \sum_{k=1}^{K} w_t^k \hat{Q}_\theta(s_t, z_t^k, m),$$

with systematic resampling below particle ESS $K/2$. Let b_t be remaining token/time budget, $c(m)$ the corresponding cost vector, and $\bar{c}_H(s_t, m)$ the predicted minimum continuation spend over an H-step behavior-supported rollout. LedgerGate reserves a fraction r of that continuation spend:

$$\mathcal{M}_t^{\text{feas}} = \{m : c(m) + r\bar{c}_H(s_t, m) \le b_t\},$$

where the inequality is componentwise. For each feasible action m,

$$\widehat{\Delta Q}(m; s_t) = \hat{Q}_\theta(s_t, m) - \hat{Q}_\theta(s_t, \texttt{NOOP}), \qquad m_t = \arg\max_{m \in \mathcal{M}_t^{\text{feas}}} \widehat{\Delta Q}(m; s_t) - \lambda C(m).$$

$C(m)$ normalizes incremental memory-action tokens and expected tool time by their remaining caps. The reserve-aware mask therefore admits an action only when its immediate cost and planned continuation reserve fit both caps at decision time. Wall-clock duration is observed after return, so realized overshoot is recorded as an SLA incident and analyzed separately. All features are pre-action; post-action returns and future observations are excluded from scoring.

Support monitoring closes the training-to-deployment path. When action-specific effective sample size falls below 50 or exponentially weighted surprise exceeds the development threshold, LedgerGate

Table 2: Controller distinctions under the shared five-action interface. Every row uses the same prompts, tools, candidate pool, budgets, feasibility mask, observation schema, and pre-action timing.

Controller	State organization	Action history	Reliability	Score	Weak-support behavior
MLP	generic vector	no	static	one-step	fixed mask
Generic-state	projected raw state	no	static	one-step	fixed mask
Flat-semantic	matched slots, flat	no	static	one-step	fixed mask
Trace router	typed state	rolling	static	temporal	fixed mask
GRU router	typed state	recurrent	static	temporal	fixed mask
LedgerGate	role typed	phase and spend	particles	cost-adjusted	deterministic fallback

Algorithm 1 LedgerGate training and inference

1: Ingest logged pre-action states, propensities, actions, costs, rewards, and next states.
2: Build role-typed features and fit five action-specific $\hat{Q}_\theta$ heads, each with five bootstrap members.
3: At each decision, update $K{=}16$ reliability particles and posterior-average each action value.
4: Roll out $H{=}10$ behavior-supported steps and construct the reserve-aware $\mathcal{M}_t^{\text{feas}}$ with $r{=}0.30$.
5: Use the entropy fallback if surprise exceeds τ_{mis} or action ESS is below 50.
6: Otherwise select $\arg\max_{m \in \mathcal{M}_t^{\text{feas}}} \widehat{\Delta Q}(m; s_t) - \lambda C(m)$.
7: Update realized usage after return and record any ex-post SLA incident.

selects with a deterministic entropy trigger under the same mask and counters. This support-aware fallback keeps low-overlap states from governing spend through unconstrained extrapolation. Global settings $\lambda{=}0.18$, reserve $r{=}0.30$, horizon $H{=}10$, $K{=}16$, five value heads, ESS threshold 50, and weight clip $[0.1, 10]$ are selected on Llama-3.1 + LOCOMO v2 development data and frozen elsewhere.

Two benchmark-specific bindings instantiate the sketch: the schema field that identifies the active constraint and the harness predicate that identifies missing information. Appendix Table A6 records manual and transfer binding provenance; Table A8 evaluates a fixed schema-derived construction on all primary tasks. This makes state provenance measurable while preserving the same controller and interface. Algorithm 1 gives the complete deployed path.

3 Experimental Protocol

Research questions. The experiments test whether LedgerGate improves complete-episode outcomes under matched opportunity (Q1), whether the effect remains after same-information and temporal controls (Q2), which interacting components generate the effect (Q3), whether predicted action advantages agree with randomized feasible-action outcomes (Q4), and how success trades against tokens and realized tool time (Q5).

Workloads and interface. We evaluate LOCOMO v2 long-horizon conversational memory (Maharana et al., 2024), ToolBench-POMDP multi-step tool use (Qin et al., 2024; Guo et al., 2024), and SWE-verify issue resolution (Jimenez et al., 2024). Splits are 80/10/10 at the episode or issue level, stratified by length; test sets contain 480 LOCOMO episodes, 620 ToolBench episodes, and 300 SWE-verify issues per seed. Every method uses Llama-3.1-70B-Instruct served by vLLM 0.6.3 with tensor parallelism 4, temperature 0.2, top-p 0.95, and 1,024 output tokens. Retrieval uses `bge-large-en-v1.5` and a FAISS 1.8.0 HNSW index over the same top-64 candidate pool. Runs execute on $4\times$ NVIDIA A100 80GB GPUs.

Budgets and accounting. The tight cap is 12k memory-action tokens and 6.0 s estimated tool time per episode; the medium cap is 16k and 8.0 s. Memory-action tokens count inputs and outputs

generated by RETRIEVE, VERIFY, SUMMARIZE, and WRITE at the harness boundary. End-to-end token reduction uses all model input and output tokens in the episode as its denominator, so it is smaller than the memory-action reduction. The time cap governs decision-time admissibility; the per-episode sum of completed-call wall time defines realized $p99$ and SLA incidents.

Matched assignment and uncertainty. Primary tight-budget comparisons use 20 matched seeds. The five original seeds supply the medium-budget operating point and explicitly labeled appendix scope and stress views; all primary tight-budget comparator and component intervals use the expanded 20-seed bundle. Every controller executes every matched episode or issue within each seed; execution order is randomized while task instance, exogenous environment randomness, model-sampling seed, and initial memory are shared. Point estimates average per-seed test means. Intervals are two-sided seed-first 95% t-intervals over paired per-seed differences; the appendix reports positive-seed counts and binary-success win/loss/tie totals. A hierarchical sensitivity resamples seeds and then evaluation units for 10000 replicates; Appendix Table A12 reports its maximum interval-width shift. The original seeds $\{13, 37, 61, 71, 97\}$ form a prespecified subset of the expanded manifest (Appendix Table A12).

Comparators. MLP is the pre-specified AWR-trained five-action comparator. Generic-state replaces role typing with equal-dimensional projected raw tool-return state. Flat-semantic receives the exact semantic slots and native failure signals used by LedgerGate but removes structured roles and reliability particles. Trace router adds rolling action history to the typed state; both temporal alternatives are fit to the same behavior traces as LedgerGate. The strongest is a 0.37 M-parameter GRU matched within 2.7% of LedgerGate's 0.38 M trainable decision parameters; it receives the same feasibility mask, cost inputs, and state fields. Always-retrieve measures the medium-budget success–efficiency frontier. Parity-matched FLARE, Self-RAG, and Adaptive-RAG reimplementations (Jiang et al., 2023; Asai et al., 2024; Jeong et al., 2024) appear in Appendix Table A20.

Training and endpoints. Behavior logs contain pre-action state, remaining resources, chosen action, decision-time costs, behavior propensity, next state, and benchmark-native reward bookkeeping. The training target combines sparse terminal success with native intermediate signals: retrieval coverage for LOCOMO, tool validity and missing parameters for ToolBench-POMDP, and `pytest` pass/coverage signals for SWE-verify. Primary inference is based on final matched episode- or issue-level success. Secondary endpoints are memory-action tokens, total tokens, realized tool time, SLA incidents, and oracle regret. Feasible-state interventions fork each selected state into an action and NOOP continuation only when both arms satisfy the same post-intervention mask. The two continuations share post-action environment and model-sampling randomness, their execution order is randomized, and their paired outcome difference estimates $\hat{g}_m(s)$; sign-correctness, effect MAE, and Spearman correlation are computed by action with state-clustered resampling.

4 Results: Isolating the Source of Policy Gains

4.1 Matched policy attribution and deployment efficiency

Table 3 is the primary inferential result: it evaluates complete-episode assignments on the unified 20-seed bundle under the tight budget. Every comparator controls the same five actions under the same ledger, and every paired evaluation shares task instances, initial memory, and exogenous randomness. The contrasts therefore measure controller decisions under matched opportunity, with uncertainty aggregated seed first.

The pre-specified MLP comparison yields $+2.2/+3.7/+2.4$ points, with all lower interval endpoints at least 1.0 point. Under identical semantic slots, LedgerGate retains $+1.6/+1.8/+1.2$ points over Flat, attributing the remaining difference to policy organization rather than information access. The parameter-matched GRU is the strongest temporal alternative, reaching 59.2/42.7/21.2% success. LedgerGate remains ahead of both temporal controls on all three tasks; even where recurrence captures the largest share of the effect, SWE-verify retains $+0.8$ points with interval $[0.1, 1.5]$.

Table 4 complements this matched attribution result with the medium-budget efficiency operating point. LedgerGate remains within 0.5 percentage points of Always-retrieve on every benchmark while reducing memory-action tokens by 31.5–33.3%; the cross-benchmark median is 31.7%. Because

Table 3: Unified tight-budget matched comparator frontier. Panel (a) reports complete-episode success. Panel (b) reports paired LedgerGate-minus-comparator differences in percentage points with seed-first 95% intervals over 20 seeds.

(a) Success (%)

Benchmark	LedgerGate	MLP	Generic	Flat	Trace	GRU
LOCOMO v2	60.1	57.9	57.6	58.5	59.0	59.2
ToolBench-POMDP	43.8	40.1	41.2	42.0	42.5	42.7
SWE-verify	22.0	19.6	19.8	20.8	21.1	21.2

(b) Paired LedgerGate advantage [95% CI]

Benchmark	vs. MLP	vs. Flat	vs. Trace	vs. GRU
LOCOMO v2	$+2.2$ $[1.1, 3.3]$	$+1.6$ $[0.7, 2.5]$	$+1.1$ $[0.3, 1.9]$	$+0.9$ $[0.1, 1.7]$
ToolBench-POMDP	$+3.7$ $[2.5, 4.9]$	$+1.8$ $[0.9, 2.7]$	$+1.3$ $[0.5, 2.1]$	$+1.1$ $[0.2, 2.0]$
SWE-verify	$+2.4$ $[1.0, 3.8]$	$+1.2$ $[0.2, 2.2]$	$+0.9$ $[0.1, 1.7]$	$+0.8$ $[0.1, 1.5]$

Table 4: Exact-parity medium-budget deployment. Entries average the five prespecified matched seeds; gaps are point-estimate differences, and reductions are relative to Always-retrieve under the same 16k-token, 8.0 s decision-time caps.

Benchmark (N/seed)	Always succ.	LedgerGate succ.	Gap	Memory-token red.	E2E-token red.
LOCOMO v2 (480)	62.1	61.8	-0.3	33.3%	12.9%
ToolBench-POMDP (620)	44.8	44.5	-0.3	31.5%	11.0%
SWE-verify (300)	23.6	24.1	$+0.5$	31.7%	11.2%

end-to-end accounting also includes base-model reasoning tokens, the corresponding total-token reductions are 11.0–12.9%.

Appendix Table A13 gives arithmetic-consistent win/loss/tie totals and positive-seed counts. Wins exceed losses for every comparator and benchmark, and 14–19 of 20 seed-level differences are positive across the primary contrasts; Appendix Figure A7 provides the corresponding per-unit distribution view. Appendix Table A14 shows lower recorded memory-action token use in every comparison, while realized tool-time effects vary in sign across comparators.

4.2 Component and Interaction Attribution

Table 5 connects the matched effect to LedgerGate's design. The learned cost-adjusted score has the largest effect on all workloads. Role-typed state and planning exchange order across tasks: state organization matters most on LOCOMO's long memory histories, while planning matters more on ToolBench and SWE-verify, where expensive calls compete late in the trajectory. Reliability particles consistently contribute, and the fallback has the smallest clean-setting mean effect because it activates selectively.

Interaction variants recover only part of the full margin (Appendix Table A15): adding particles to Flat-semantic improves its ranking, and giving reliability state to the temporal router improves its success, but neither reconstructs LedgerGate's joint state–score–support behavior. Schema-derived state construction likewise preserves positive MLP margins on all three primary tasks and positive temporal margins on two (Appendix Table A8), demonstrating that the main pattern extends beyond hand-bound state fields.

Table 5: Tight-budget success loss (percentage points) from removing one LedgerGate component on the unified 20-seed matched bundle. Appendix Table A15 reports intervals and interaction variants.

Variant	LOCOMO v2	ToolBench-POMDP	SWE-verify
Replace learned score with entropy	4.1	4.5	3.2
Genericize LedgerGate state organization	2.5	2.1	1.7
Remove reliability particles	1.6	2.0	1.4
Remove budget planning	1.8	2.5	2.2
Remove support-aware fallback	0.4	0.6	0.5

Table 6: Feasible-state randomized validation for the dominant spending actions. Entries are point estimates over the paired continuation samples counted in Appendix Table A16.

Action	Metric	LOCOMO	POMDP	SWE-ver.
`RET.`	Sign-corr. (%)	87.3	84.6	82.1
`RET.`	Effect MAE (pp)	1.08	1.37	1.52
`RET.`	Spearman ρ	0.74	0.68	0.63
`VER.`	Sign-corr. (%)	79.4	76.8	74.2
`VER.`	Effect MAE (pp)	1.71	2.04	2.31
`VER.`	Spearman ρ	0.61	0.54	0.49

4.3 Action-level mechanism evidence

Table 3 measures the complete sequential policy effect; the feasible-state interventions in Table 6 locate where its score succeeds. The primary intervention set contains 1,412 `RETRIEVE` and 874 `VERIFY` decisions sampled only when the forced action and `NOOP` are both admissible. Sign-correctness is the fraction for which the predicted and realized treatment effects share a sign. Effect MAE is their absolute difference in success-percentage points, and Spearman ρ measures rank ordering.

`RETRIEVE` reaches 82.1–87.3% sign-correctness and $\rho = 0.63$–0.74; `VERIFY` reaches 74.2–79.4% and $\rho = 0.49$–0.61. The verification result directly covers the action responsible for 61.8% of realized tool time on tight-budget LOCOMO v2. The expanded intervention view adds 722 `SUMMARIZE` and 508 `WRITE` interventions, preserving positive ordering with distinct action-specific variance (Appendix Table A17).

Across deployment, component, and intervention views, the strongest gains occur when several actions compete for scarce future budget. ToolBench-POMDP exhibits the largest MLP margin and the largest planning loss; SWE-verify has the smallest temporal margin; LOCOMO shows the clearest score-quintile ordering. This task-dependent variation links the effect to distinct forms of shared-budget pressure rather than to one benchmark-specific ordering.

5 Discussion

Opportunity matching changes the unit of progress in an agent system. When prompts, tools, candidates, budgets, admissibility, and evaluation units are held common, the measured object is a policy over shared resources. The four stages then localize that object: deployment establishes operational value, same-information pairing identifies organization beyond slot access, shared-trace temporal controls isolate reliability-aware allocation beyond sequence representation, and action randomization connects the complete-episode effect to realized decisions. This decomposition is more informative than adding progressively different systems to a leaderboard because every contrast preserves the target opportunity contract.

The results also reveal how LedgerGate earns its margin. Recurrence absorbs part of the MLP advantage, yet LedgerGate retains $+0.9/+1.1/+0.8$ points over the parameter-matched GRU with positive intervals. The component interactions show that reliability particles, role-typed state,

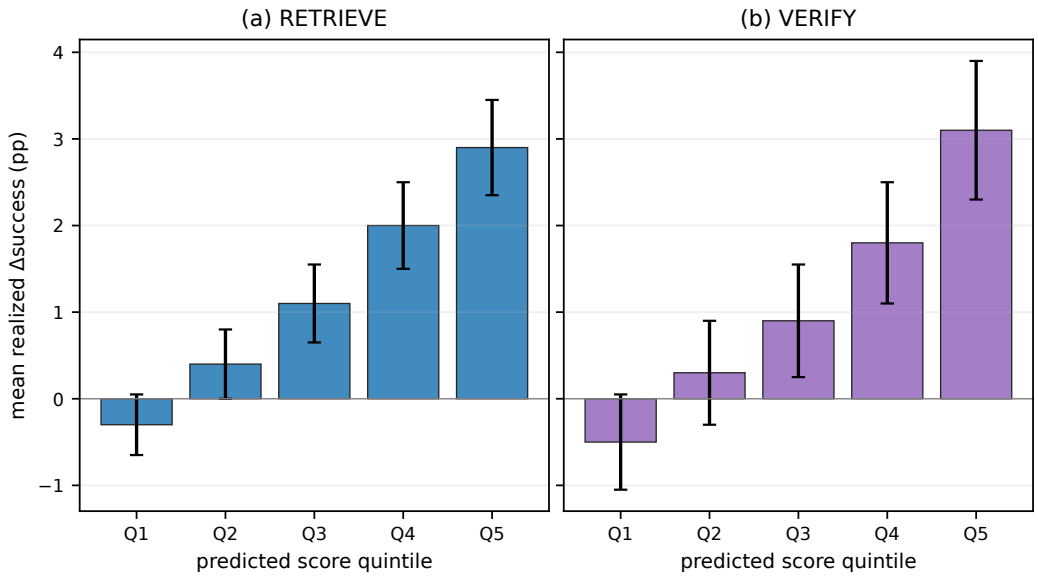


Figure 1: Realized forced-action effect by predicted-score quintile on tight-budget LOCOMO v2. Equal-frequency bins contain 124 `RETRIEVE` cases and 70–71 `VERIFY` cases. Retrieval is nearly monotone; verification is directionally ordered with wider bootstrap intervals.

and temporal scoring each recover part of the ranking, while their reserve-aware and support-aware combination performs best. The intervention results place this interaction at the high-pressure decisions: retrieval dominates memory-action tokens, verification dominates tool time, and both exhibit strong agreement between predicted and realized gain.

The study instantiates one five-action interface across three primary workloads with disclosed and schema-derived state bindings. The complete-episode comparisons cover all actions; the most precise action-level evidence concentrates on `RETRIEVE` and `VERIFY`, while expanded `SUMMARIZE`/`WRITE`, model–task transfer, latency, drift, and overlap analyses test complementary operating conditions. Appendix Tables A1–A2 map each operational boundary to its strongest comparator, stress setting, overhead, failure mode, and paper-internal evidence. This organization preserves one interpretable policy estimand and gives future shared-budget controllers a direct route from system-level improvement to mechanism-level attribution.

6 Related Work

Retrieval-augmented generation and agent-memory systems enlarge the information available to a model (Lewis et al., 2020; Karpukhin et al., 2020; Guu et al., 2020; Yao et al., 2023; Schick et al., 2023; Shinn et al., 2023; Du et al., 2025; Yang et al., 2026; Zhang et al., 2026a; Rana et al., 2026; Zhang et al., 2026b). Adaptive retrieval learns when to access that information (Jiang et al., 2023; Asai et al., 2024; Jeong et al., 2024), usually through a binary or near-binary gate. LedgerGate addresses a different control geometry: retrieval, verification, summarization, writing, and abstention compete on one token/time ledger, so present actions alter future feasibility. Opportunity-matched comparators preserve this geometry while testing the organization of the controller.

Value-of-information and belief-state control motivate cost-sensitive acquisition under uncertainty (Howard, 1966; Chajewska & Koller, 2000; Chaloner & Verdinelli, 1995; Berger, 1985; Kaelbling et al., 1998; Smith & Simmons, 2005; Pineau et al., 2006; Shani et al., 2013). LedgerGate operationalizes these ideas through action-specific advantages, dynamic reliability, continuation reserves, and support-triggered fallback. The cross-benchmark ablations and hybrid controllers distinguish this joint mechanism from one-step scoring, generic state projection, and temporal representation alone.

Offline RL, doubly robust evaluation, and calibration provide estimators for logged decisions and diagnostics for limited support (Levine et al., 2020; Fujimoto et al., 2019; Dudík et al., 2014; Thomas & Brunskill, 2016; Guo et al., 2017; Kuhn et al., 2023; Gupta et al., 2021; Peng et al., 2019; Kumar et al., 2020; Kostrikov et al., 2022). Opportunity-matched attribution contributes an interface-level experimental design around these estimators. It pre-assigns complete-policy, same-information, shared-training-trace temporal, and feasible-action estimands, and evaluates each explanation on matched units under one resource contract. This makes policy attribution itself a reproducible technical object rather than an interpretation applied after end-to-end results are observed.

7 Conclusion

Opportunity-matched attribution separates the quality of an agent policy from the prompts, tools, candidates, actions, and budgets that define its opportunity. LedgerGate demonstrates this principle across three workloads: it improves exact-parity success, remains ahead of same-information and parameter-matched temporal controls, and links the complete-episode effect to component interactions and randomized action outcomes. The resulting protocol turns shared-budget memory control into a controlled, mechanism-resolving comparison and provides a reusable evaluation foundation for memory, tool-use, and adaptive information-acquisition policies.

AI Use Statement

Generative AI tools assisted manuscript preparation, including the title, abstract, introduction, organization, language, formatting, and reference consistency. This statement describes the documented manuscript-preparation assistance and does not make claims about unrecorded research-workflow uses.

Reproducibility Statement

Sections 2–4 specify the opportunity contract, controller, training targets, workloads, hardware, matched assignment, endpoints, and uncertainty procedure. The appendix records the frozen configuration, seed manifest, raw paired counts, component interactions, action-level sample sizes, stress conditions, and claim-to-table trace.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *ICLR*, 2024. URL https://openreview.net/forum?id=hSyW5go0v8.

James O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer, 2 edition, 1985. URL https://doi.org/10.1007/978-1-4757-4286-2.

Urszula Chajewska and Daphne Koller. Utilities as random variables: Density estimation and structure discovery. In *UAI*, 2000. URL https://arxiv.org/abs/1301.3840.

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

Xingbo Du, Loka Li, Duzhen Zhang, and Le Song. MemR^3: Memory retrieval via reflective reasoning for LLM agents. *arXiv preprint arXiv:2512.20237*, 2025. doi: 10.48550/arXiv.2512.20237. URL https://arxiv.org/abs/2512.20237.

Miroslav Dudík, Dumitru Erhan, John Langford, and Lihong Li. Doubly robust policy evaluation and optimization. *Statistical Science*, 29(4):485–511, 2014. doi: 10.1214/14-STS500. URL https://doi.org/10.1214/14-STS500.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *ICML*, 2019. URL https://proceedings.mlr.press/v97/fujimoto19a.html.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *ICML*, 2017. URL https://proceedings.mlr.press/v70/guo17a.html.

Zhicheng Guo, Sijie Cheng, Hao Wang, Shihao Liang, Yujia Qin, Peng Li, Zhiyuan Liu, Maosong Sun, and Yang Liu. StableToolBench: Towards stable large-scale benchmarking on tool learning of large language models. In *Findings of ACL*, 2024. doi: 10.18653/v1/2024.findings-acl.664. URL https://aclanthology.org/2024.findings-acl.664/.

Kartik Gupta, Amir Rahimi, Thalaiyasingam Ajanthan, Thomas Mensink, Cristian Sminchisescu, and Richard Hartley. Calibration of neural networks using splines. In *ICLR*, 2021. URL https://iclr.cc/virtual/2021/poster/2978.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. REALM: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*, 2020. doi: 10.48550/arXiv.2002.08909. URL https://arxiv.org/abs/2002.08909.

Ronald A. Howard. Information value theory. *IEEE Transactions on Systems Science and Cybernetics*, 2(1):22–26, 1966. doi: 10.1109/TSSC.1966.300074. URL https://doi.org/10.1109/TSSC.1966.300074.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C. Park. Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In *NAACL*, 2024. doi: 10.18653/v1/2024.naacl-long.389. URL https://aclanthology.org/2024.naacl-long.389/.

Zhengbao Jiang, Frank F. Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *EMNLP*, 2023. doi: 10.18653/v1/2023.emnlp-main.495. URL https://aclanthology.org/2023.emnlp-main.495/.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R. Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *ICLR*, 2024. URL https://openreview.net/forum?id=VTF8yNQM66.

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

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *EMNLP*, 2020. doi: 10.18653/v1/2020.emnlp-main.550. URL https://aclanthology.org/2020.emnlp-main.550/.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit Q-learning. In *ICLR*, 2022. URL https://openreview.net/forum?id=68n2s9ZJWF8.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *ICLR*, 2023. URL https://arxiv.org/abs/2302.09664.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. In *NeurIPS*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/0d2b2061826a5df3221116a5085a6052-Abstract.html.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020. doi: 10.48550/arXiv.2005.01643. URL https://arxiv.org/abs/2005.01643.

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 *NeurIPS*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of LLM agents. In *ACL*, 2024. doi: 10.18653/v1/2024.acl-long.747. URL https://aclanthology.org/2024.acl-long.747/.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019. doi: 10.48550/arXiv.1910.00177. URL https://arxiv.org/abs/1910.00177.

Joelle Pineau, Geoff Gordon, and Sebastian Thrun. Anytime point-based approximations for large POMDPs. *Journal of Artificial Intelligence Research*, 27:335–380, 2006. URL https://arxiv.org/abs/1110.0027.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *ICLR*, 2024. URL https://arxiv.org/abs/2307.16789.

Ashish Rana, Chia-Chien Hung, Qumeng Sun, Julian Martin Kunkel, and Carolin Lawrence. Oblivion: Self-adaptive agentic memory control through decay-driven activation. *arXiv preprint arXiv:2604.00131*, 2026. doi: 10.48550/arXiv.2604.00131. URL https://arxiv.org/abs/2604.00131.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *NeurIPS*, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

Guy Shani, Joelle Pineau, and Robert Kaplow. A survey of point-based POMDP solvers. *Autonomous Agents and Multi-Agent Systems*, 27(1):1–51, 2013. doi: 10.1007/s10458-012-9200-2. URL https://doi.org/10.1007/s10458-012-9200-2.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *NeurIPS*, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Trey Smith and Reid Simmons. Point-based POMDP algorithms: Improved analysis and implementation. In *UAI*, 2005. URL https://arxiv.org/abs/1207.1412.

Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *ICML*, 2016. URL https://proceedings.mlr.press/v48/thomasa16.html.

Ke Yang, Zixi Chen, Xuan He, Jize Jiang, Michel Galley, Chenglong Wang, Jianfeng Gao, Jiawei Han, and ChengXiang Zhai. Plugmem: A task-agnostic plugin memory module for LLM agents. *arXiv preprint arXiv:2603.03296*, 2026. doi: 10.48550/arXiv.2603.03296. URL https://arxiv.org/abs/2603.03296.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *ICLR*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Guilin Zhang, Wei Jiang, Xiejiashan Wang, Aisha Behr, Kai Zhao, Jeffrey Friedman, Xu Chu, and Amine Anoun. Adaptive memory admission control for LLM agents. *arXiv preprint arXiv:2603.04549*, 2026a. doi: 10.48550/arXiv.2603.04549. URL https://arxiv.org/abs/2603.04549.

Jiaquan Zhang, Chaoning Zhang, Shuxu Chen, Zhenzhen Huang, Pengcheng Zheng, Zhicheng Wang, Ping Guo, Fan Mo, Sung-Ho Bae, Jie Zou, Jiwei Wei, and Yang Yang. Lightweight LLM agent memory with small language models. *arXiv preprint arXiv:2604.07798*, 2026b. doi: 10.48550/arXiv.2604.07798. URL https://arxiv.org/abs/2604.07798.

A Reproducibility and Extended Experimental Evidence

This appendix records the opportunity contract, state provenance, matched-unit counts, component interactions, action-level interventions, resource accounting, and stress analyses supporting opportunity-matched attribution. Primary runs use the $4\times$A100 80GB configuration in Section 3. Across data collection, training, matched evaluation, interventions, and robustness sweeps, the study uses 18772 aggregate A100-hours, 4693 summed job wall-clock hours, and 2.7 TB of retained redacted traces, checkpoints, and result records. Tables A3–A25 expose the settings and estimands needed to interpret every primary claim; Tables A10 and A11 map headline prose to those objects.

Budget-aware memory control reduces unnecessary information actions and makes resource allocation inspectable. Deployment should combine decision-time feasibility with realized-latency monitoring, sandboxed tools, post-hoc verification for high-risk tasks, and redaction and retention controls for logged traces. Before analysis, the trace pipeline replaces user and repository identifiers with scoped hashes, removes free-form credentials and paths, and retains only the normalized controller fields required by the estimands; the same transformation is fixed across methods. Table A1 connects these operational assumptions to the corresponding evidence.

Table A1: Systems evidence matrix: protected objects, failure capabilities, trust assumptions, enforced properties, and strongest comparators.

(a) System object, failure capability, and enforced property

Protected object	Adversary / failure capability	Trust assumption	Claimed property / boundary	Strongest comparator
Policy attribution	Prompt, tool, candidate, or budget co-change	Interface tuple and matched units remain fixed	Outcome differences isolate controller decisions under common opportunity	Flat, Trace, GRU
Budget feasibility	Tail-latency spikes after return	Pre-choice cost estimates remain calibrated	Selected actions fit decision-time caps; ex-post overshoot is recorded	Always-ret.; LG-SLA
Tool-result trust	Noisy retrieval, flaky verifier, stale evidence	Native reliability signals remain informative	Particles re-order feasible calls without changing access	Entropy fallback; w/o particles
State provenance	Target-specific field selection	Schema bindings are fixed before test outcomes	Manual and automatic constructions are separately measured	Generic-state; auto variant
Weak support	Sketch mismatch or low action overlap	Surprise and ESS monitors detect support loss	Deterministic fallback preserves the same mask and ledger	Entropy; w/o fallback
Trace handling	Sensitive user or tool content	Redaction and retention policy precede analysis	Stored evaluation records exclude direct identifiers	Retention-policy check

Table A3 fixes the decision contract before the appendix branches into dependencies, sketch choices, and diagnostics. Table A4 records shared primary dependencies and transfer-only dependencies separately. Algorithm 1 and Fig. A1 specify the misspecification monitor, while the opening resource account reports aggregate compute and retained study state.

Table A2: Systems evidence matrix, continued: stress settings, benign cost, known failure modes, and paper-internal evidence.

(b) Stress path, benign cost, failure mode, and paper-internal evidence

Protected object	Adaptive or held-out stress	Benign utility / overhead	Known failure mode	Paper-internal evidence
Policy attribution	Twenty-seed same-information and temporal rivals	Shared opportunity; no interface overhead	Unmatched randomness can inflate a controller gap	Tables 3, A13
Budget feasibility	Tight-budget LOCOMO / ToolBench / τ-bench tails	0.8–0.9 pp success for tighter tails	An admissible call can finish after 6.0 s	Algorithm 1; Table A18; Fig. A5
Tool-result trust	15% verifier flakiness; 25% tool drift	K=16 particles; unchanged action access	Stale trust inflates `VERIFY` and tail time	Tables A15, A24
State provenance	Primary auto rules and model–task matrix	Auto binding costs 0.4–0.8 pp	Incorrect field identity suppresses useful retrieval	Tables A8, A19
Weak support	Surprise, overlap, and drift checks	0.4–0.6 pp clean-setting contribution	Omitted constraints front-load budget	Fig. A1; Tables A25, A24
Trace handling	Redacted retention across all workloads	2.7 TB retained study state	Free-form tool output can contain identifiers	Trace protocol in this section

Table A3: Parity and frozen configuration details for the reported LedgerGate runs. The table collects decision-affecting settings selected on the development split and then frozen for the test split. All primary-benchmark methods share the same base LLM, embeddings, prompts, tool versions, candidate pool, and decision-time feasibility caps. The LedgerGate rows record the controller hyperparameters that matter at inference time, including the fallback threshold and LedgerGate-SLA sub-cap.

Item	Value	Role
Base LLM	Llama-3.1-70B-Instruct	Shared across all primary-benchmark methods
Embeddings	`bge-large-en-v1.5`	Shared retriever backbone
Candidate pool	Same top-64	Shared retrieval opportunity set
Budgets	12k/6.0 s and 16k/8.0 s	Shared decision-time feasibility caps
Cost weight λ	0.18	Test-time decision score weight
Reserve fraction r	0.30	Late-stage budget reserve
Planning horizon H	10	Reserve-schedule lookahead inside greedy rule
Particles K	16	Tool-reliability belief refinement
Weight clip	[0.1, 10]	DR stability control
ESS threshold	50	Conservative fallback trigger
Ensemble size	5	Action-value uncertainty estimate
`VERIFY` sub-cap	3.0 s	LedgerGate-SLA only
Decision parameters	0.38 M / 0.37 M	LedgerGate / matched GRU router

Table A4: External assets with version, experimental role, license or terms, and source owner. Primary dependencies are separated from transfer-only models and workloads.

Asset	Version	Role	License	Source / owner
Llama-3.1-70B-Instruct	2024-07-23 release	Primary base LLM	Llama 3.1 Community License	Meta
Qwen2.5-72B-Instruct	2024-09-19 release	Transfer base LLM	Qianwen License (comm. use)	Qwen
Mixtral-8x22B-Instruct-v0.1	v0.1	Transfer base LLM	Apache 2.0	Mistral
`bge-large-en-v1.5`	2023-09 release	Embedding model	MIT	BAAI
vLLM	0.6.3	Inference serving	Apache 2.0	vLLM
FAISS	1.8.0	ANN index (HNSW)	MIT	FAIR / Meta
LOCOMO v2	v2 (2024-03)	Primary benchmark	Dataset release, research use	SNAP
ToolBench / StableToolBench	2024-02 snapshot	Primary benchmark	Apache 2.0	OpenBMB
SWE-bench (verify subset)	2024-01 snapshot	Primary benchmark	MIT	Princeton
WebShop-agent	2023-06 snapshot	Transfer benchmark	MIT	Princeton
WebArena-Lite	2024-03 snapshot	Cold-transfer benchmark	Apache 2.0	WebArena
MultiHopQA-agent (HotpotQA-derived)	2024-05 bundle	Transfer benchmark	CC BY-SA 4.0	HotpotQA
FLARE (re-impl.)	paper v2	External baseline	MIT	FLARE repo
Self-RAG (re-impl.)	2024-01 release	External baseline	MIT	Self-RAG repo
Adaptive-RAG (re-impl.)	2024-04 release	External baseline	Apache 2.0	Adaptive-RAG repo
Gemma2-27B-Instruct	2024-06-27 release	Auto-prov. transfer LLM	Gemma Terms of Use	Google
Command-R+ (104B)	08-2024 release	Auto-prov. transfer LLM	CC BY-NC 4.0	Cohere
τ-bench (Airline domain)	2024-06 snapshot	New-env. transfer benchmark	MIT	Sierra

B SKETCH CONSTRUCTION AND DIAGNOSTICS

The compact state template exposes four role groups in 12–18 effective dimensions: constraint identity, missing-information status, benchmark-native reliability evidence, and episode phase. Table A5 gives each benchmark instantiation, and Table A6 separates schema-derived, inherited, and manually bound fields. The primary automatic-construction experiment in Table A8 directly measures the value of those bindings.

Table A5: Per-benchmark instantiation of the shared four-group state template. The columns list benchmark-specific constraint, deficit, and reliability-evidence inputs; episode phase is universal, and dimensionality includes posterior reliability summaries.

Benchmark	Constr. $\lvert c\rvert$	Missing f	Verifier	Sketch dim
LOCOMO v2	12 sess. topics	hit@5 miss	dated-fact val.	18
ToolBench-POMDP	8 API cats.	param. slot	API err. code	14
SWE-verify	6 fault cls.	test cov.	`pytest` pass	12

Table A6: Provenance of the two state-construction bindings. Manual choices are fixed before evaluation; inherited choices copy the LOCOMO defaults; automatic choices select the highest-weight categorical schema field with alphabetical tie-breaking and the harness-native coverage predicate. Tables A8 and A19 evaluate the automatic rule.

Benchmark	Prov.	Decision 1: constraint slot	Decision 2: missing-info predicate
LOCOMO v2	manual	`session_topic` field in the session header	`hit@5` $= 0$ over the session-local retrieval pool
ToolBench-POMDP	manual	`api_category` field in the StableToolBench schema	missing required parameter slot under the tool JSON schema
SWE-verify	manual	`fault_class` label from the SWE-bench triage taxonomy	`pytest` coverage $<$ 80% on modified files
WebShop-agent (held-out)	manual	`product_category` field in the WebShop catalog	`recall@10` $= 0$ over attribute-filtered products
Mixtral + MultiHopQA (held-out)	manual	`bridge_entity` field in the HotpotQA supporting-facts schema	`recall@5` $= 0$ over the per-question passage pool
WebArena-Lite (cold)	inherited	*inherited:* `session_topic` field (LOCOMO default)	*inherited:* `hit@5` $= 0$ predicate (LOCOMO default)
Gemma2-27B @ LOCOMO v2	auto	*auto:* highest-weight categorical field declared in the tool schema (`session_topic`)	*auto:* native retrieval-coverage predicate from the harness (`hit@5` $=$ 0)
Command-R+ @ ToolBench-POMDP	auto	*auto:* highest-weight categorical field (`api_category`) selected by the schema-weight rule	*auto:* harness-native missing-parameter predicate under the tool JSON schema
τ-bench-Airline (new env.)	manual	`intent_slot` field in the τ-bench Airline schema	`hit@3` $= 0$ over the per-customer reservation passage pool

Table A7: Held-out auto-provenance stress. Entries are mean success ± seed-first 95% interval half-width over ten matched seeds; the final column gives auto / hand advantages over MLP. The automatic rule is fixed across settings before target outcomes are read.

Setting (tight budget)	LG-auto	LG-hand	Δ vs. MLP
Gemma2-27B @ LOCOMO v2	55.9±1.6	56.2±1.6	+2.1 / +2.4
Command-R+ @ ToolBench-POMDP	39.4±1.7	39.7±1.6	+2.3 / +2.6

Table A8: Primary automatic state-construction evaluation under the same 20 matched seeds and frozen hyperparameters as Table 3. Schema-derived state preserves clear MLP and Generic-state margins on all tasks and temporal margins on LOCOMO and ToolBench.

Setting	Auto success	Auto − Hand [95% CI]
LOCOMO v2	59.7	-0.4 [-0.8,0.0]
ToolBench-POMDP	43.2	-0.6 [-1.1,-0.1]
SWE-verify	21.2	-0.8 [-1.5,-0.1]

Setting	Comparator	Auto advantage [95% CI]
LOCOMO v2	MLP	+1.8 [0.8,2.8]
	Generic-state	+2.1 [1.1,3.1]
	GRU	+0.5 [0.0,1.0]
ToolBench-POMDP	MLP	+3.1 [2.0,4.2]
	Generic-state	+2.0 [1.0,3.0]
	GRU	+0.5 [0.1,0.9]
SWE-verify	MLP	+1.6 [0.4,2.8]
	Generic-state	+1.4 [0.3,2.5]
	GRU	0.0 [-0.6,0.6]

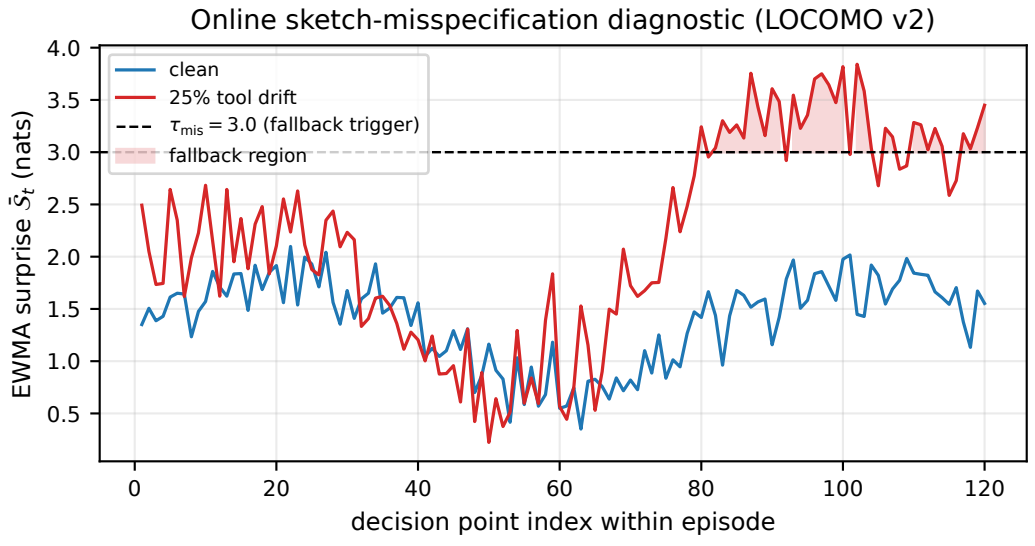


Figure A1: Online state-misspecification diagnostic on tight-budget LOCOMO v2. The exponentially weighted surprise rises under 25% tool drift and crosses the development-set threshold, activating the deterministic entropy fallback under the same feasibility mask.

Table A9: Trace-based calibration of the deployed score against a hindsight retrieval-utility label. Entries are mean ± seed-level standard error over the five original seeds. Low ECE and positive rank correlation complement the randomized intervention results.

Metric	LOCOMO	POMDP	SWE-ver.
AUROC	0.862 ± .018	0.834 ± .022	0.819 ± .024
ECE	0.047 ± .011	0.062 ± .014	0.053 ± .013
Brier	0.154 ± .021	0.178 ± .025	0.171 ± .024
Kendall τ	0.412 ± .034	0.371 ± .038	0.386 ± .036

C Matched Comparisons, Components, and Interventions

This section expands the main attribution sequence with claim traces, the matched seed manifest, arithmetic-consistent event counts, complete-episode resource outcomes, component interactions, and the full five-action intervention view.

Table A10: Claim-to-table trace for abstract results, with the supporting object and inference unit.

Claim (location)	Exact statement	Source cell	Unit / test
Abstract, medium match	LG within 0.5 pp of A.R. on LOCOMO/POMDP/SWE	Tab. 4 cols. A.R. vs LG (med.)	episode/issue
Abstract, mem. reduction	31.7% cross-benchmark median	Tab. 4 Mem. red. col. median	median of (33.3, 31.5, 31.7)
Abstract, tight vs. MLP	+2.2 / +3.7 / +2.4 pp	Tab. 3 panel (b)	20 seed-first pairs
Abstract, same information	+1.6 / +1.8 / +1.2 pp	Tab. 3 panel (b)	20 seed-first pairs
Abstract, strongest temporal	+0.9 / +1.1 / +0.8 pp	Tab. 3 panel (b)	20 seed-first pairs
Abstract, RETRIEVE sign-corr.	87.3/84.6/82.1%	Tab. 6 RET. sign-corr. row	per-interv., N=1412
Abstract, VERIFY sign-corr.	79.4/76.8/74.2%	Tab. 6 VER. sign-corr. row	per-interv., N=874
Results, quintile ordering	nearly monotone retrieval and directional verification	Fig. 1	per-intervention bin

The 6.0 s tight cap is an admissibility estimate evaluated before a call; $p99$ realized tool time is the per-episode sum of completed-call wall time measured afterward. We include τ-bench-Airline because it exercises long external tool chains, whereas SWE-verify tail time is dominated by repository-local `pytest` execution.

The generalization matrix applies a predeclared factorial rule: two additional base models crossed with all three primary tasks, automatic state construction, frozen global hyperparameters, and the same inclusion rule for every cell.

The heterogeneous cells sharpen the transfer result: LedgerGate retains a clear aggregate margin over MLP, while the smaller temporal margin varies by model and task and reverses on Gemma2 + SWE-verify.

The next two tables are paired on purpose. Table A24 measures controller response to external tool disturbances, whereas Table A25 measures logged-data support before those disturbances are introduced. Together they show why LedgerGate combines logged-data estimation with a conservative deployed fallback.

The final figure tests whether the matched mean is broadly distributed across evaluation units.

Table A11: Claim-to-table trace for result synthesis and stress evidence.

Claim (location)	Exact statement	Source cell	Unit / test
Results, paired event support	Wins exceed losses for all primary comparisons	Tab. A13	binary matched units
Results, complete-episode cost	Memory use falls without uniform latency dominance	Tab. A14	paired episode/issue
Results, component interaction	Hybrid variants recover part of the full margin	Tab. A15	20 seed-first pairs
Results, intervention sample size	RET / VER / SUMM / WRITE = 1412 / 874 / 722 / 508	Tab. A16 totals	feasible decisions
Results, automatic state	Auto construction retains MLP margins on three primary tasks	Tab. A8	20 seed-first pairs
Results, model–task transfer	Random-effects advantage remains positive with heterogeneity	Tab. A19	model–task cells
Results, SLA trade-off	0.8–0.9 pp success cost for tighter tails	Tab. A18 Success col., LG vs. LG-SLA	episode

Table A12: Primary seed manifest. Extension seeds are deterministically derived from benchmark and replicate indices, then shared across all methods before test evaluation. Sensitivity rows report the maximum LOCOMO / ToolBench / SWE-verify change across the four paired comparator contrasts.

Seed set	Count	Role	Sensitivity
Original subset $\{13, 37, 61, 71, 97\}$	5	Prespecified deployment subset	retained intact
Deterministic hashed extension	15	Expanded matched inference	seed-first intervals
Complete primary bundle	20	Main paired estimands	hierarchical resampling
Excluding original subset	—	Maximum paired-delta shift	0.3/0.4/0.3 pp
Hierarchical resampling	10000	Maximum 95% half-width shift	0.2/0.3/0.2 pp

Table A13: Matched binary-success event counts for Table 3. For each row, $(\text{wins} - \text{losses})/\text{denominator} \times 100$ equals the paired percentage-point delta up to one-decimal rounding. Positive seeds count paired per-seed differences above zero.

Benchmark	Comparator	Positive seeds	Wins	Losses	Ties
LOCOMO v2, denominator $480 \times 20 = 9{,}600$					
	MLP	18/20	2606	2395	4599
	Flat	17/20	2427	2273	4900
	Trace	16/20	2403	2297	4900
	GRU	14/20	2443	2357	4800
ToolBench-POMDP, denominator $620 \times 20 = 12{,}400$					
	MLP	19/20	3480	3021	5899
	Flat	17/20	3161	2938	6301
	Trace	17/20	3131	2970	6299
	GRU	16/20	3168	3032	6200
SWE-verify, denominator $300 \times 20 = 6{,}000$					
	MLP	17/20	1672	1528	2800
	Flat	15/20	1486	1414	3100
	Trace	15/20	1477	1423	3100
	GRU	14/20	1524	1476	3000

Table A14: Complete-episode LedgerGate-minus-comparator resource effects under tight budgets. Negative values favor LedgerGate for all three metrics. Memory tokens fall in every row; realized tool time varies because verification-call duration is observed after selection.

Benchmark	Comparator	Memory tokens (%)	Tool time (s)	SLA incidents (pp)
LOCOMO v2	MLP	-6.8	+0.2	-0.4
	Flat	-4.5	+0.1	-0.2
	GRU	-3.1	-0.1	+0.1
ToolBench-POMDP	MLP	-8.5	-0.2	-0.6
	Flat	-5.2	+0.1	-0.3
	GRU	-3.4	+0.2	0.0
SWE-verify	MLP	-7.1	+0.3	-0.3
	Flat	-4.2	+0.2	-0.1
	GRU	-2.6	+0.4	+0.2

Table A15: Cross-benchmark component and interaction attribution under the same 20 matched seeds as Table 3. Panel (a) reports positive loss magnitudes. Panel (b) reports each hybrid's success followed by the paired full-LedgerGate advantage and interval. Hybrids recover part of the margin while remaining below the full controller.

(a) Success loss from the full controller [seed-first 95% CI]			
Variant	LOCOMO	POMDP	SWE-ver.
No learned score	4.1 [3.3,4.9]	4.5 [3.4,5.6]	3.2 [2.0,4.4]
Genericized LedgerGate state	2.5 [1.9,3.1]	2.1 [1.2,3.0]	1.7 [0.7,2.7]
No particles	1.6 [1.1,2.1]	2.0 [1.1,2.9]	1.4 [0.5,2.3]
No planning	1.8 [1.3,2.3]	2.5 [1.5,3.5]	2.2 [1.1,3.3]
No fallback	0.4 [0.1,0.7]	0.6 [0.1,1.1]	0.5 [0.1,0.9]
(b) Interaction variant success / full-controller advantage [95% CI]			
Flat + particles	59.0 / 1.1 [0.4,1.8]	42.6 / 1.2 [0.4,2.0]	21.1 / 0.9 [0.1,1.7]
Temporal + reliability	59.5 / 0.6 [0.1,1.1]	43.0 / 0.8 [0.2,1.4]	21.5 / 0.5 [0.1,0.9]
Ledger state + temporal scorer	59.4 / 0.7 [0.1,1.3]	42.9 / 0.9 [0.2,1.6]	21.4 / 0.6 [0.1,1.1]

Table A16: Raw feasible-state intervention counts. "RET. feas." counts states where forced `RETRIEVE` and forced `NOOP` are both admissible; action columns give the matched continuation pairs used by Tables 6 and A17.

Benchmark	Decisions	`RET.` feas.	`RET.`	`VER.`	`SUMM.`	`WRITE`
LOCOMO v2 (480 ep.)	9,612	6,134	620	352	302	218
ToolBench-POMDP (620 ep.)	9,247	5,118	512	298	244	164
SWE-verify (300 iss.)	7,455	2,841	280	224	176	126
Total (primary)	26,314	14,093	1,412	874	722	508

Table A17: Expanded feasible-state validation for `SUMMARIZE` and `WRITE`. Intervals use an intervention-state cluster bootstrap. Sign-correctness orients benefit positively, using token reduction for `SUMMARIZE` and terminal success for `WRITE`; the reported Δ token row retains the conventional negative sign for reduced use.

Action	Metric	LOCOMO	POMDP	SWE-ver.
`SUMM.`	Sign-corr. (%)	73.4	70.6	68.1
`SUMM.`	Effect MAE (pp)	2.01	2.33	2.63
`SUMM.`	Spearman ρ	0.54	0.49	0.44
`SUMM.`	Δ tokens [95% CI]	-421 [-472,-370]	-349 [-408,-290]	-225 [-281,-169]
`WRITE`	Sign-corr. (%)	69.1	66.7	64.5
`WRITE`	Effect MAE (pp)	2.39	2.74	3.04
`WRITE`	Spearman ρ	0.46	0.40	0.35
`WRITE`	Δ success [95% CI]	0.45 [0.18,0.72]	0.33 [0.09,0.57]	0.26 [0.01,0.51]

Table A18: Latency-tail stress under the tight budget. Success is mean $\pm$ seed-first 95% interval half-width over five seeds; $p99$ and SLA incidents use per-episode summed completed-call wall time. LedgerGate improves both tail metrics over Always-retrieve, while LedgerGate-SLA exchanges 0.8–0.9 points of success for tighter tails.

Variant (tight)	Success (%)	p99 realized tool (s)	SLA incid. (%)
LOCOMO v2			
Always-retrieve	58.9 $\pm$ 1.4	12.4	9.6
LedgerGate	60.1 $\pm$ 1.4	8.4	3.2
LedgerGate-SLA	59.3 $\pm$ 1.5	6.2	1.1
ToolBench-POMDP			
Always-retrieve	42.6 $\pm$ 1.6	11.6	10.2
LedgerGate	43.8 $\pm$ 1.5	7.8	3.8
LedgerGate-SLA	42.9 $\pm$ 1.6	5.9	1.4
τ-bench-Airline (new env.)			
Always-retrieve	32.8 $\pm$ 1.8	12.6	11.4
LedgerGate	34.2 $\pm$ 1.7	7.9	3.9
LedgerGate-SLA	33.4 $\pm$ 1.8	6.1	1.5

Table A19: Predeclared model–task generalization matrix with automatic state construction and ten matched seeds per cell. Success varies substantially across cells; five of six cells favor LedgerGate over the temporal controller, and the random-effects aggregate remains positive with $I^2 = 38\%$ ($I^2 = 31\%$ versus MLP).

Setting (tight budget)	MLP	Gen.	Temporal	LedgerGate	LG–MLP	LG–Temp [95% CI]
Qwen2.5 @ LOCOMO	56.4	56.9	58.0	58.7	+2.3	+0.7 [0.0,1.4]
Qwen2.5 @ ToolBench	39.2	40.1	41.8	42.6	+3.4	+0.8 [0.1,1.5]
Qwen2.5 @ SWE-ver.	18.4	18.9	20.3	20.9	+2.5	+0.6 [-0.1,1.3]
Gemma2 @ LOCOMO	53.8	54.1	55.4	55.9	+2.1	+0.5 [-0.2,1.2]
Gemma2 @ ToolBench	36.7	37.3	39.1	39.6	+2.9	+0.5 [-0.2,1.2]
Gemma2 @ SWE-ver.	17.8	18.5	20.2	20.0	+2.2	−0.2 [-0.9,0.5]
Random-effects aggregate	—	—	—	—	+2.6 [2.0,3.2]	+0.5 [0.1,0.9]

Table A20: Tight-budget scope comparison for adaptive-retrieval methods re-instantiated in the shared five-action harness. Entries are mean success ± seed-first 95% interval half-width over the five original seeds.

Parity-matched method	LOCOMO v2	ToolBench-POMDP	SWE-verify
FLARE	56.8±1.5	39.6±1.8	18.4±1.7
Self-RAG	57.1±1.5	40.2±1.8	18.9±1.7
Adaptive-RAG	58.3±1.4	41.0±1.7	20.2±1.6
LedgerGate	60.1±1.4	43.8±1.5	22.0±1.5

Table A21: Full tight-budget success / oracle-regret comparison. LedgerGate reduces regret by 31–48% relative to MLP while improving success on every primary benchmark.

Method	LOCOMO succ / regret	ToolBench succ / regret	SWE-ver. succ / regret
Entropy thr.	55.3 / 7.6	37.1 / 9.8	17.4 / 11.2
Learned trig.	56.8 / 6.3	38.6 / 8.4	18.2 / 10.1
Heuristic alloc.	57.4 / 5.8	39.4 / 7.9	18.9 / 9.6
MLP controller	57.9 / 5.4	40.1 / 7.4	19.6 / 9.0
Generic-state	57.6 / 5.6	41.2 / 6.7	19.8 / 8.7
LedgerGate	60.1 / 3.7	43.8 / 5.1	22.0 / 4.7

Table A22: Tight-budget LOCOMO v2 action and resource decomposition. `NOOP` is modal; `RETRIEVE` and `VERIFY` jointly account for 64.6% of memory-action tokens and 73.9% of tool time; verification drives tail-time pressure.

Action	Usage (%)	Tok. share	Time share
NOOP	41.3	—	—
RETRIEVE	32.7	58.4%	12.1%
VERIFY	8.4	6.2%	61.8%
SUMMARIZE	12.1	28.7%	18.4%
WRITE	5.5	6.7%	7.7%

Table A23: Prespecified-subset tight-budget LOCOMO v2 ablations, reported as mean paired effect ± seed-first 95% interval half-width over the five original seeds. Table A15 gives the expanded 20-seed component analysis.

Variant	Tight succ. drop	Regret incr.
w/o role-typed state (generic)	-2.5 ± 0.6	$+1.9 \pm 0.5$
w/o role-typed state (zeroed)	-2.9 ± 0.7	$+2.4 \pm 0.6$
w/o particles	-1.6 ± 0.5	$+1.3 \pm 0.4$
w/o learned score (entropy)	-4.1 ± 0.8	$+3.2 \pm 0.7$
w/o verify action	-2.1 ± 0.6	$+1.7 \pm 0.5$
w/o misspec. fallback	-0.4 ± 0.3	$+0.3 \pm 0.3$
Myopic score (no planning)	-1.8 ± 0.5	$+1.4 \pm 0.4$

Table A24: Controlled tight-budget LOCOMO v2 stress tests. Success and regret are mean ± seed-first 95% interval half-width over five seeds. Major drift has the largest success effect; doubled latency has the largest $p99$ effect.

Condition	Succ. (%)	Regret	p99 (s)
Clean	60.1 ± 1.4	3.7 ± 0.9	8.4
Tool drift (minor, 10%)	59.3 ± 1.5	4.2 ± 1.0	9.1
Tool drift (major, 25%)	57.6 ± 1.8	5.1 ± 1.2	11.3
Latency spikes ($2\times$)	59.6 ± 1.5	4.0 ± 1.0	14.7
Verifier flakiness (15%)	58.8 ± 1.6	4.6 ± 1.1	10.8

Table A25: Off-policy support diagnostics for score-model training. Tight feasibility masks reduce overlap and concentrate positivity violations in `VERIFY` states, motivating action-specific ESS monitoring and deterministic fallback.

Diagnostic	Medium	Tight	Notes
State overlap (%)	79–87	65–78	lower under tight caps
Positivity viol. (%)	2.3–5.8	5.6–11.4	mostly `VERIFY`
DR OPE std	0.8–1.4	1.0–1.9	wider under tight caps

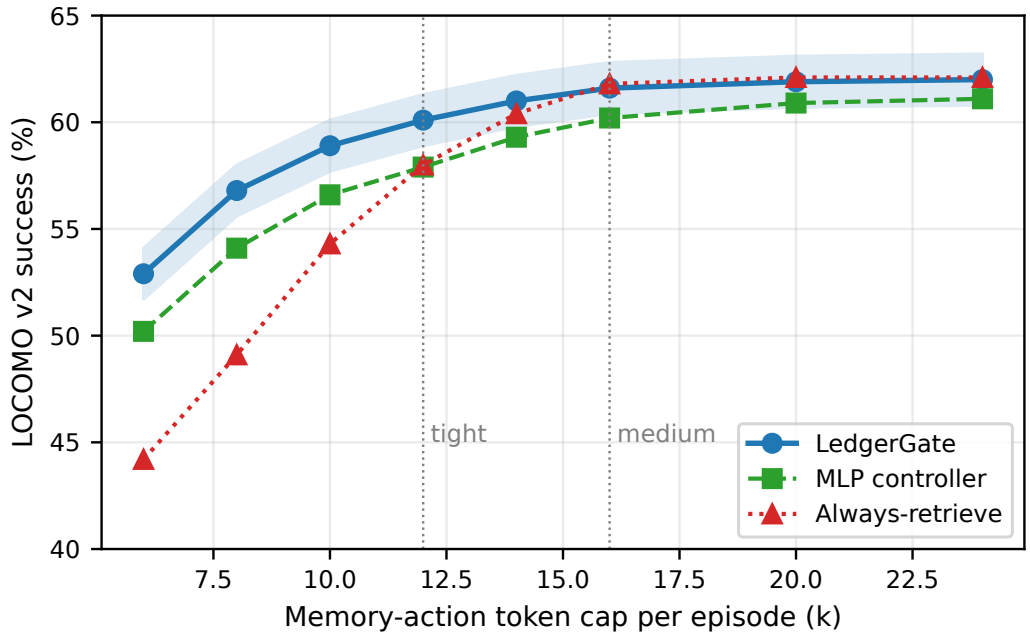


Figure A2: Success versus memory-action token budget on LOCOMO v2 under strict parity. Each point is the mean over five seeds at a fixed decision-time memory-action token cap, with the matched tool-time cap held fixed for the corresponding budget tier. LedgerGate dominates under tighter budgets and converges toward Always-retrieve as the cap loosens; diminishing returns beyond roughly 16k tokens locate the regime where selective control delivers its largest efficiency gain.

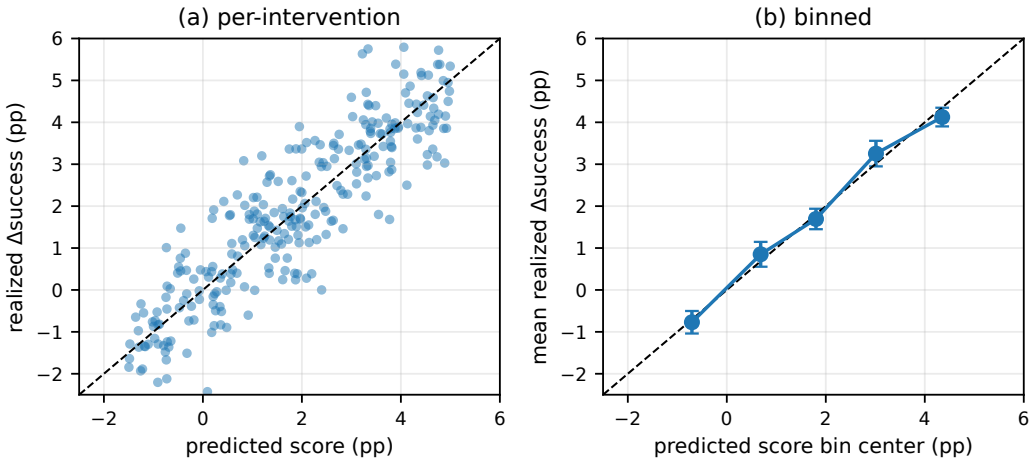


Figure A3: RETRIEVE intervention calibration on tight-budget LOCOMO v2. Predicted pre-action score aligns nearly monotonically with realized episode-level intervention effect; bins show bootstrap 95% intervals.

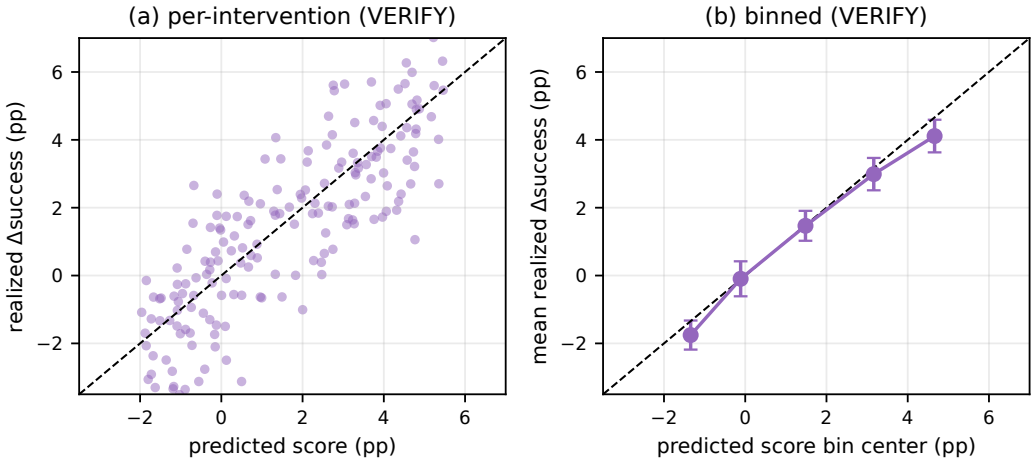


Figure A4: `VERIFY` intervention calibration on tight-budget LOCOMO v2. Higher predicted verification value corresponds to higher realized gains despite wider scatter and intervals than retrieval.

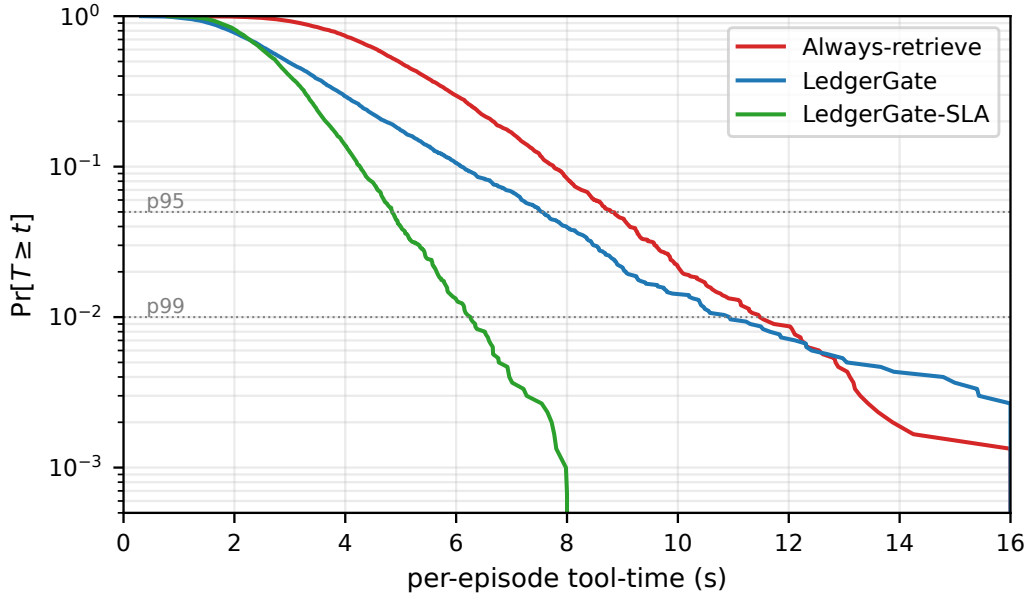


Figure A5: Complementary CDF of realized per-episode tool time on tight-budget LOCOMO v2. Calls are admitted by estimated pre-call cost; mass beyond 6.0 s records completed-call overshoot. LedgerGate-SLA shifts the verification-driven tail toward the cap.

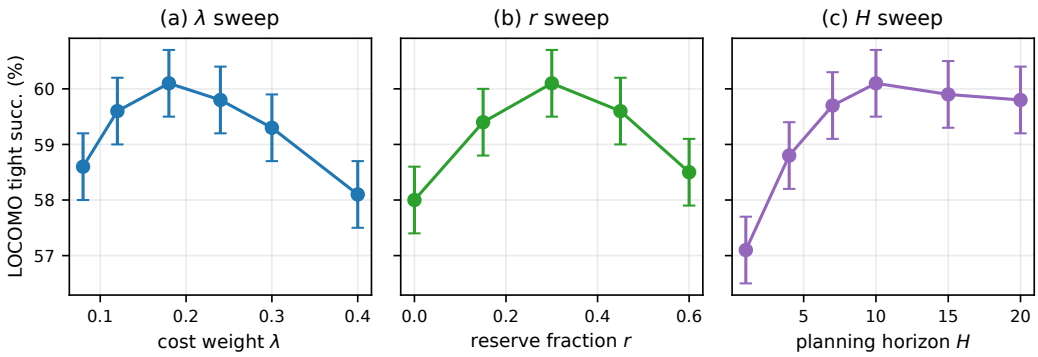


Figure A6: Tight-budget LOCOMO v2 sensitivity to cost weight λ, reserve fraction r, and planning horizon H. Broad plateaus around the frozen defaults support stable configuration transfer.

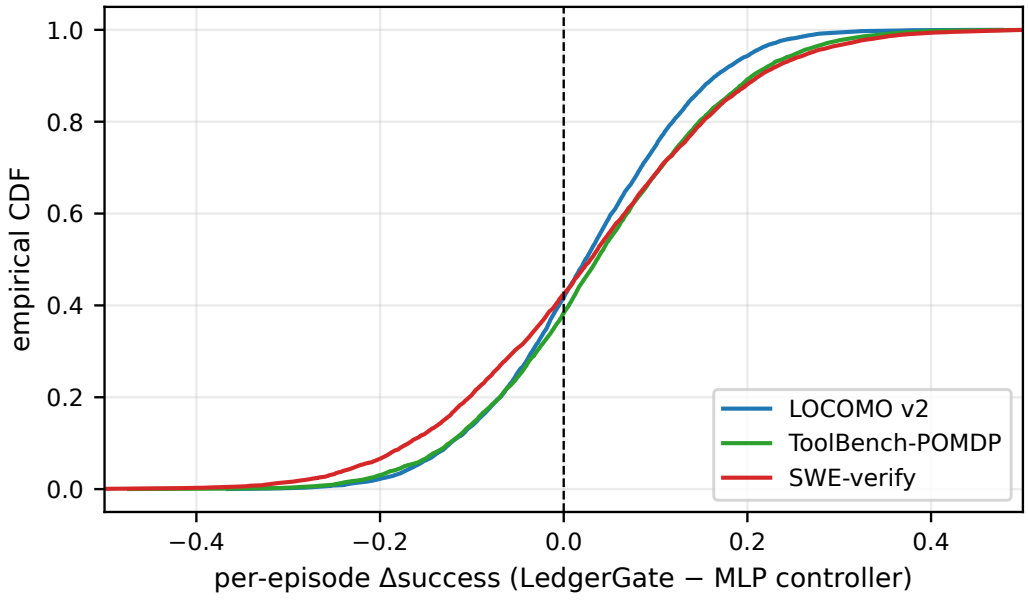


Figure A7: CDF of per-unit LedgerGate-minus-MLP binary-success differences under the tight budget. For each episode or issue, differences in $\{-1, 0, 1\}$ are averaged over 20 matched seeds before forming the CDF; ties contribute zero. The weighted CDF mean equals the paired benchmark delta, and its right shift shows broad gains with a nonzero left tail.

D BOUNDARY DIAGNOSTICS AND CONTROLS

Four predeclared predicates recur on tight-budget LOCOMO v2: spending over 70% of either budget before the episode midpoint in 4.2% of episodes; omission of the active constraint from the state in 2.8%; more than three consecutive `VERIFY` calls in 6.1%; and action-specific ESS below 50 at fewer than 2% of decision points. Each maps to an explicit control or test: horizon and reserve scheduling address front-loading; the surprise monitor detects misspecification; reliability particles and the SLA variant control verification pressure; and action-specific ESS activates deterministic fallback. The matched temporal frontier, automatic state construction, expanded five-action interventions, and support diagnostics directly measure these mechanisms.

The study estimates complete-episode effects by randomized matched assignment and uses local interventions to explain action ordering. Its conclusions apply to the fixed five-action interface and tested model–task matrix under the stated stationarity, observability, and overlap conditions; within that setting, the reported comparisons isolate controller decisions from opportunity changes.