THE CONTEXT GATHERING DECISION PROCESS: A POMDP FRAMEWORK FOR AGENTIC SEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

A coding assistant asked about an unfamiliar repository cannot fit it into its prompt, so it searches: it issues a query, reads the result, and issues another. Harnesses run this phase, *context gathering*, by appending each passage to the prompt. One LLM call then chooses the next query, remembers what earlier rounds found, and decides when to stop. None of the three can be changed separately. We formalize the phase as a Context Gathering Decision Process, a finite-horizon POMDP over a fixed corpus, which separates two decisions the fused call makes together. The first is whether the agent's notes are rewritten into a bounded form after each observation, or accumulate. The second is who rewrites them: the policy, prompted to summarize itself, or the orchestrator, the code that runs between LLM calls. We call the latter an *Orchestrator-Managed State* (OMS). Rewriting raises accuracy over four published harnesses and three question-answering domains. Ownership does not change accuracy. It changes whether the agent declines a question its corpus cannot answer, where correct abstention rises 15.9 points under OMS. A stopping rule reading the state in code ends stagnant searches early, with no LLM call. On two frontier policies, rewriting still repairs a deleted history but no longer beats an intact one. Accuracy and abstention therefore come from different parts of a harness, a reason to expose the update and the stopping rule as orchestrator interfaces. Pieces of an agent loop usually written as prompt engineering can be separated, measured and replaced without retraining.

1 INTRODUCTION

A coding assistant tracing a bug can retrieve the same file on three successive rounds, then answer without checking the caller it meant to look at next. Nothing in its prompt records that the file has already been read. The repository is far larger than the assistant's reliable working context (Yang et al., 2024), so the assistant is in the phase we call *context gathering*: online search over a source the agent's actions do not change within an episode. Our claims cover that setting and no other.

Longer context windows do not remove the difficulty (Li et al., 2024; Wu et al., 2024). Long-context models remain position sensitive (Liu et al., 2024). A *harness* has two halves: the prompt the policy LLM sees, and the *orchestrator*, the code that runs between LLM calls and decides what the next prompt contains. Both are part of the same system, and every harness has both. In most harnesses the orchestrator does nothing but append the raw observation, so the division is easy to miss. A single LLM call then chooses the next query, remembers what earlier rounds found, and decides when to answer, with the growing prompt as the only record of any of it. Our own baseline trajectories show three patterns at scale: the agent repeats queries it has already issued, drifts away from sub-questions it raised and never resolved, and stops after one or two rounds (Appendix I).

We formalize this as a Context Gathering Decision Process (CGDP), a finite-horizon POMDP (Kaelbling et al., 1998) whose latent state is a corpus the agent's actions do not change, whose actions are costly queries and whose reward arrives only at the final answer. It gives us a vocabulary for the agent loop. Its Bellman recursion has exactly four operations (choose a query, receive text, revise what carries forward, and decide whether to answer), which we name ACT, OBSERVE, UPDATE and STOP and treat as separate interfaces. Harnesses usually implement OBSERVE with a tool and fuse the other three inside one LLM call, which we argue against.

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.

UPDATE is the operation all three fused decisions share. The next query and the next stopping decision can read only what it writes. An exact agent would carry the Bayesian posterior b_t over what the corpus contains, but that posterior is intractable, and every implementation keeps a bounded state z_t instead. Following a k-hop chain among N candidates needs $\Omega(k \log N)$ bits of persistent state, however that state is worded (Observation 1). A bounded state can also *alias*: if two histories that call for different next queries leave the same z_t, a policy reading only z_t must answer both the same way, and loses at least half the margin between them (Observation 2).

We build two lightweight implementations on the decomposition. The **Orchestrator-Managed State** (OMS, §5.1) is a bounded natural-language state that a separate extraction call rewrites every round. The **Stagnation Gate** (§5.2) is a programmatic STOP monitor that spends no LLM call of its own and fires once queries repeat, observations stop being novel, and the state has almost stopped changing.

Contributions.

1. **Formulation.** We define the CGDP as a static-source, costly-query POMDP specialization, and we keep its exact posterior b_t distinct from the bounded state z_t an implementation carries (§3).
2. **Architecture.** We expose ACT, OBSERVE, UPDATE and STOP as separate harness interfaces and instantiate UPDATE with a bounded orchestrator-managed state (§5).
3. **Experimental evidence.** Enforcing the bounded update raises accuracy. It recovers all nine cells in which we delete the harness's history, and it beats each harness's own memory significantly in six of twelve while losing significantly in none. Choosing the owner moves abstention instead. Who writes the state leaves pooled accuracy unchanged, while under an orchestrator-owned state the agent abstains correctly 15.9 points more often on questions the corpus cannot answer (§6.1, §6.2). A Stagnation Gate with thresholds frozen from earlier runs then cuts redundant search under a prespecified non-inferiority margin (§6.3).

2 RELATED WORK

Most systems in this area change exactly one layer of the agent loop, and we organize prior work that way (Table 41, Appendix K). The policy layer improves *which* query is issued, by structuring the trajectory (Trivedi et al., 2023; Yao et al., 2023), conditioning the next query on the previous generation (Shao et al., 2023), or by training the model to interleave search with reasoning (Jin et al., 2025; Li et al., 2025b), and the corpus layer reorganizes the documents offline into structures such as graphs and summary hierarchies (Gutiérrez et al., 2024; Edge et al., 2025). Both compose with ours, since we leave the corpus alone and organize online what the agent learns.

Online state and trajectory management. This is the layer we modify. StateAct (Rozanov & Rei, 2025) prompts the policy to re-emit goals, FAIR-RAG (Aghajani Asl et al., 2025) maintains an evidence checklist, MemGPT (Packer et al., 2023) exposes memory as tools the policy calls, A-MEM (Xu et al., 2025) grows a linked memory network, and SLIM (Yen et al., 2026) summarizes periodically. In each of them the policy owns the update. Ours is a mandatory orchestrator step that runs after *every* observation, stays inside a size bound, and replaces the history. §6.2 varies the owner directly and runs the StateAct, FAIR-RAG and SLIM rules in our UPDATE step at matched budget.

Stopping. FLARE (Jiang et al., 2023) and DRAGIN (Su et al., 2024) decide *whether to retrieve* from token confidence or attention. Our gate addresses the opposite end of the round, namely whether repeated retrieval has stopped producing new evidence, and it reads the state itself, since the policy's own verbalized confidence is poorly calibrated (Chen et al., 2023).

3 THE CONTEXT GATHERING DECISION PROCESS

3.1 DEFINITION AND OPTIMAL VALUE

The tuple below names the components of a gathering episode, its per-round cost and its horizon.

Definition 1 (CGDP)**.** A Context Gathering Decision Process is a tuple $\big(\mathcal{Q}, \mu, \mathcal{C}, \mathcal{A}_Q, \mathcal{O}, P, \mathcal{Y}, R, \kappa, H, \lambda\big)$ in which

- $q \in \mathcal{Q}$ is a task, drawn with a latent corpus $c \in \mathcal{C}$ from $(q, c) \sim \mathcal{D}$, where $\mu(\cdot \mid q)$ denotes the conditional law of c;
- $\mathcal{A}_Q$ is a finite set of *query actions* and $\mathcal{O}$ a set of observations, related by an observation kernel $P(o \mid a, c)$;
- $\mathcal{Y}$ is the terminal answer space, including abstention when the task permits it, and $R(q, c, y) \in [0, 1]$ is a bounded terminal utility;
- $\kappa(b, a) \geq 0$ is the cost of action a in tokens, summed across policy, orchestrator and tool, where b is the posterior of Eq. 2, and $\lambda > 0$ converts cost into utility units;
- H is the horizon: an episode ends at an answer or after H queries.

Query actions reveal the corpus without changing it, and at round t the agent knows only the history $h_t = (q, a_0, o_0, \ldots, a_{t-1}, o_{t-1})$.

A policy maps a history to a query action or an answer, and the objective is

$$\arg\max_{\pi} \; \mathbb{E}_{(q,c)\sim\mathcal{D}}\Big[\, R\big(q, c, y^{\pi}\big) \;-\; \lambda \textstyle\sum_{t=0}^{T^{\pi}-1} \kappa(b_t, a_t)\Big], \tag{1}$$

where $T^{\pi} \leq H$ counts the queries issued and y^{π} is the answer. Because the corpus is static, the only state variable is the exact posterior

$$b_t(c) \;=\; \mathbb{P}\big(c \mid h_t\big) \;\propto\; \mu(c \mid q) \prod_{s<t} P(o_s \mid a_s, c), \tag{2}$$

a sufficient statistic of h_t. Write $\tau(b, a, o)$ for the Bayes update of b under (a, o) and $P(o \mid b, a) = \mathbb{E}_{c\sim b}[P(o \mid a, c)]$.

With r queries left, the optimal value is the better of answering now and issuing one more query.

Proposition 1 (Optimal value)**.** *Let $V_r(b)$ be the optimal expected value with r query actions remaining. Then $V_0(b) = \max_{y\in\mathcal{Y}} \mathbb{E}_{c\sim b}[R(q, c, y)]$ and, for $r \geq 1$,*

$$V_r(b) \;=\; \max\Big\{ \underbrace{\max_{y\in\mathcal{Y}} \mathbb{E}_{c\sim b}\big[R(q, c, y)\big]}_{\textit{stop and answer}},\; \underbrace{\max_{a\in\mathcal{A}_Q} \Big[\, -\lambda\kappa(b, a) + \textstyle\sum_o P(o \mid b, a)\, V_{r-1}\big(\tau(b, a, o)\big)\Big]}_{\textit{gather more evidence}} \Big\}. \tag{3}$$

Stopping is optimal exactly when no query returns more than its cost. Note that a rule reading neither quantity can stop while a cheap informative query remains, or continue once every query is redundant.

3.2 Exact posterior versus bounded textual state

Eq. 2 is not implementable: $\mathcal{C}$ is a corpus space, and an LLM harness carries text and no distribution over it. Reserving b_t for the exact posterior, we write

$$z_t \;=\; \phi(h_t) \in \mathcal{Z}, \qquad z_{t+1} \;=\; U(z_t, q, a_t, o_t), \tag{4}$$

for the *bounded textual information state* an implementation maintains, with ϕ an encoder, U an update rule, and $\mathcal{Z}$ a bounded set of texts. A standard harness is the special case

$$z_{t+1} \;=\; \mathrm{Truncate}\big(z_t \oplus \{a_t, o_t\}\big), \tag{5}$$

i.e. append the round verbatim and drop the oldest tokens once the window fills, with $a_{t+1} = \mathrm{LLM}(z_{t+1})$. We want z_t to be *decision-sufficient*: two histories with $\phi(h) = \phi(h')$ should admit a common optimal continuation. Eq. 5 does not guarantee this, and nor does any bounded summarizer. Suppose two histories have resolved the same sub-questions, and one of them has already searched the release notes and come back empty. A state that records only what was found cannot separate them, and yet the first agent should look somewhere else while the second should look there.

A collision is costly. We show in Appendix A that if two histories reaching the same z_t call for different unique optimal actions, each beating its alternatives by a margin of at least Δ, then every policy reading z_t incurs expected regret at least $\Delta/2$ on the pair (Observation 2). Once a lossy update merges two histories that require different continuations, no downstream policy, prompt or test-time compute recovers the difference. So we need an update that preserves what the continuation depends on, for example the unresolved sub-questions, the queries already tried and provenance (Appendix I).

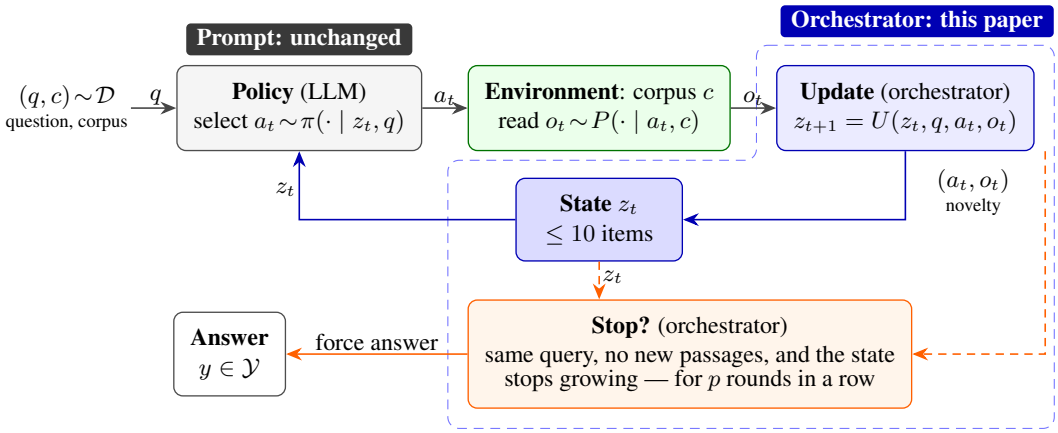


Figure 1: The CGDP agent loop as four separable interfaces. An episode draws a question and a corpus, $(q, c) \sim \mathcal{D}$, and the policy reads the corpus only through its own queries. A harness has two halves: the prompt the policy LLM sees, and the orchestrator code that runs between calls. Harnesses that append raw observations fuse **Act**, **Update** and **Stop** into the call and leave the orchestrator to concatenate. We move **Update** and **Stop** out into the orchestrator (dashed), where a bounded textual state z_t (§5.1) replaces the raw trajectory and a programmatic monitor (§5.2) forces an answer once the search stops making progress; the three conditions in its box are the three terms of Eq. 7. The monitor spends no LLM call of its own and saves 16.7% of the tokens (§6.3). We leave every harness's prompt unchanged and retrain nothing.

Longer context windows do not substitute for the state. Aliasing is a property of the encoder ϕ, and enlarging the window changes which histories collide without removing them, since a text of fixed size takes fewer values than there are histories.

Observation 1 (Compositional memory lower bound)**.** *Consider identifying a chain $X = (X_1, \ldots, X_k)$ drawn uniformly from $[N]^k$, where the evidence for hop i is observable only in round i, and the agent's only carrier of information across rounds is a persistent state of m bits, with the terminal answer a function of the final state alone. Then any agent satisfies $\mathbb{P}(\hat{X} = X) \leq 2^m / N^k$, so $\mathbb{P}(\hat{X} = X) \geq 1 - \delta$ requires*

$$m \;\geq\; k \log_2 N + \log_2(1 - \delta). \tag{6}$$

The corpus size $|\mathcal{C}|$ does not appear. The requirement scales with the unresolved hops, not with how much text the corpus holds. A fixed-size state should therefore degrade as the hop count rises, and we test that in §6.3 on synthetic chains that vary depth alone.

4 The CGDP Agent Loop

The recursion of Eq. 3 performs four operations in temporal order: choose a query, receive text, revise the state, and decide whether to answer. We expose them as four *interfaces*, Act, Observe, Update and Stop. Algorithm 1 states them as the four marked steps of one round, and Figure 1 organizes them. In a standard harness the orchestrator implements Observe with a retrieval tool and appends what it returns to the prompt. One LLM call then fuses the other three, and none of them can be inspected, bounded or replaced on its own. The Stop interface covers three conditions that are easily conflated: a budget stop, an evidence-sufficiency stop when the answer branch of Eq. 3 dominates, and a stagnation intervention that fires when continuing is predicted to add nothing. Only the third is ours (§5.2); the first two are in every harness we study.

We map four published harnesses onto these interfaces in Table 1. None of the four owns Update outside the policy, and only ReAct exposes Stop as an action the orchestrator can observe.

5 Two Orchestrator-Owned Interventions

5.1 Orchestrator-Managed State (OMS)

OMS implements Eq. 4's update $z_{t+1} = U(z_t, q, a_t, o_t)$ as a separate LLM call after every observation, which we call the *extractor* and cap at $<$500 output tokens. The orchestrator injects z_{t+1} into the policy prompt and *discards* o_t. The state is a list of short natural-language *items*, one per fact the extractor keeps, and round t reaches $t + 1$ only through it (Figure 2).

Algorithm 1 The CGDP agent loop; the marked steps are the four interfaces of §4. Each round consults STOP before acting, on the state the previous UPDATE produced. UPDATE is bounded and discards the raw observation, so the policy at round $t+1$ conditions on z_{t+1} alone.

Input: task q; horizon H; token budget B; state updater U; stopping monitor Stop; policy π
Output: terminal answer $y \in \mathcal{Y}$

$z_0 \leftarrow U_{\text{init}}(q)$ {sub-questions implied by q; no evidence yet}
for $t = 0$ **to** $H - 1$ **do**
  Stop? $(\text{halt}, \text{reason}) \leftarrow \mathrm{Stop}(z_t,\ t,\ \text{budget spent})$
  if halt **then**
    return $y \leftarrow \pi_{\text{answer}}(q, z_t)$ {terminal answer from the state alone}
  end if
  Act $a_t \sim \pi(\cdot \mid q, z_t)$
  Observe $o_t \sim P(\cdot \mid a_t, c)$
  Update $z_{t+1} \leftarrow U(z_t, q, a_t, o_t)$ {bounded; raw o_t is discarded}
end for
return $y \leftarrow \pi_{\text{answer}}(q, z_H)$

Table 1: Four harnesses mapped onto the interfaces of Algorithm 1. *Fused* = the operation occurs inside the policy's own LLM call on the shared prompt, and is not separately inspectable, bounded, or replaceable. It says nothing about quality: Iter-RetGen keeps no separate state, carrying information forward inside its previous generation, and it is the strongest native baseline on all three domains. §6 tests which of these fusions actually costs accuracy.

	IRCoT	**ReAct**	**Iter-RetGen**	**MemGPT**
State z_t	CoT + passages, context-bounded	trajectory, context-bounded	previous generation only	LLM-managed memory block
STOP	fused (generation ends)	explicit `Finish` action	fixed round count	fused (LLM tool call)
ACT	fused in next CoT sentence	explicit `Search` action	conditioned on previous generation	explicit tool call
UPDATE	fused, append-only	fused, append-only	none across rounds	fused (LLM memory edit)

Three invariants separate this from summarize-the-context prompting. **(i) Boundedness.** A reorganizer call fires whenever the state exceeds $K_{\text{trigger}} = 10$ items and curates it back to $K = 6$, and the footprint does not grow with H. It triggers in 20.8% of 16,280 episodes (Appendix H.4). **(ii) Orchestrator-side composition.** The extractor returns only what is new in o_t, the orchestrator merges it in and drops any item that duplicates or prefixes one the state already holds, and the policy never writes to z_t. That separates OMS from Eq. 5 and from the model-managed control of §6.2, in which the policy replaces the state in full every round. **(iii) Unresolved gaps.** The state should keep what is unknown as well as what is known. This invariant holds less often than we intended, and the model-managed control does better. In episodes with an unproductive round, an explicit negative finding appears in 4.1% of OMS episodes against 7.7% of model-managed ones (Appendix B.5).

Representation. We write the state as *freeform* natural language. We tried a *structured* JSON state with an `open_predicates` field for unresolved gaps, and it does worse: it forces the extractor to fill fields it cannot support, and it writes `no evidence` into them (§6.1, Appendix F.5).

5.2 THE STAGNATION GATE

Our gate implements STOP and targets one failure mode, *redundant continuation*: the policy keeps issuing near-duplicate queries that return already-seen text. When it fires it says only that the trajectory has stopped producing novelty, and nothing about whether the answer is in c.

We track three signals over a trailing window of two rounds. **Action similarity** Jac_t is the token-level Jaccard overlap between the query string of a_t and those of the window's actions. **Novelty** UPR_t is the fraction of passages returned at round t whose chunk ids are new. **Growth** Δz_t is the net change in the number of state items, and a round that only curates counts as no progress. The orchestrator

Q. *What is the capital of the county that shares a border with the county where the band Tenement was formed?* **Gold:** Green Bay.

z_5, in full — what the policy reads at the last round

```
- Tenement is a three piece rock band from Appleton,
Wisconsin formed in 2006, which is located in
Outagamie County, Wisconsin.
- Brown County shares a border with Outagamie County,
Wisconsin.
- Green Bay is the county seat and capital of Brown
County, Wisconsin.
```

Answer: **Green Bay**. Native IRCoT, reading its full trajectory, returns *"Green Bay (Brown County), Oshkosh (Winnebago County), or Chilton (Calumet County)"*.

t	$\lvert z_t \rvert$	Δz_t	what the round settles
1	6	$+6$	the band's city
2	9	$+3$	the city's county
3	9	0	nothing; query repeats
4	6	-3	the bordering county
5	3	-3	that county's seat

Items never reach the $K_{\text{trigger}} = 10$ cap. The reorganizer fires at rounds 4 and 5, and the raw passages are discarded every round.

Figure 2: **The bounded state, rendered from one logged episode.** Four hops separate this question from its answer: the band's city, that city's county, a county bordering it, and that county's seat. The box is the whole of z_5, the three items the policy conditions on at the last round, with the four hops fused into them. The state grows while hops are open and contracts once they resolve, so its size tracks the unresolved chain. Round 3 adds nothing, and that $\Delta z_t = 0$ is the term the STOP monitor of §5.2 reads, which a harness that appends observations to a prompt cannot compute.

evaluates

$$\text{Stagnated}_t \;:=\; \big(\text{Jac}_t > \tau_J\big) \wedge \big(\text{UPR}_t < \tau_U\big) \wedge \big(\Delta z_t < \tau_D\big). \tag{7}$$

When all three hold for p consecutive rounds, the orchestrator interrupts the loop and requires the terminal answer, using the same prompt as any other stop. We isolate *when* an episode stops and hold *how* fixed. We use $\tau_J = 0.6$, $\tau_U = 0.3$, $\tau_D = 1.0$ and $p = 2$, chosen on a `gpt-4o-mini` sweep of 16 configurations, then *frozen* and applied unchanged to Llama-3.3-70B (§6.3).

The third conjunct couples STOP to UPDATE. The orchestrator can compute Δz_t only when some module owns UPDATE. Without OMS the conjunct is vacuous, and it holds in 100.0% of $3{,}014$ history-removed rounds against 57.7% of $2{,}848$ OMS ones. A gate without that term only catches exact repetition, firing on 13.1 pp more episodes (§6.3).

6 EXPERIMENTS

Every harness we study makes two decisions, and it makes both inside a single LLM call. The first is whether a bounded update is enforced after every observation, i.e. whether round $t + 1$ reads a rewritten state or an appended transcript. The second is who performs that rewrite, an orchestrator module or the policy itself. The two are independent, and §6.3 reports where that independence breaks down.

Setup. We cross four harnesses (IRCoT (Trivedi et al., 2023), ReAct (Yao et al., 2023), Iter-RetGen (Shao et al., 2023) and MemGPT (Packer et al., 2023)) with three read-only domains, as the CGDP requires, giving 12 cells. LoCoMo (Maharana et al., 2024) asks about long conversations, MuSiQue (Trivedi et al., 2022) asks multi-hop Wikipedia questions, and SWE-QA-Pro (Cai et al., 2026) asks 260 questions over 26 repository checkouts. We run policies, extractor and orchestrator on `gpt-4o-mini` at temperature 0, a `gpt-4o` rubric judge scores answers 0–100 with Token-F1, EM, ROUGE and SBERT agreeing on the ordering (Appendix F.2), and cost is total API tokens including orchestrator calls. We give retrieval, prompts, seeds, counts and ablations in Appendices D–E and H.

Conditions. We run three arms in every comparison below. *Native* is the harness as published. *History-removed* deletes the one channel through which that harness carries information across rounds — its trajectory, its scratchpad or its memory block — and puts nothing in its place. Anything the agent learned in round t is gone by round $t + 1$. This is what our interventions must recover from. Iter-RetGen has no such carrier, so deletion applies in nine of twelve cells. *+OMS* adds the orchestrator-owned state, making it the *only* cross-round channel. For example, ReAct's native prompt carries its whole thought–action–observation trajectory forward, history removal leaves round $t + 1$ with the question and the tool alone, and +OMS gives it the question and at most ten state items.

Analysis. Our unit of analysis is the question. §6.1 uses the full question pool. We draw the ownership and gate families from a fixed 300-question subsample, and the abstention tests from

Table 2: **Enforcing the update improves accuracy.** Rubric accuracy (0–100) by harness and domain. Deleting a harness's native cross-round carrier (*history-removed*) costs accuracy in 8 of the 9 cells that have one, and putting the orchestrator-owned state there instead (**+OMS**) beats *native* in 10 of 12 cells as prose (*free*) and 6 of 12 as a fixed predicate schema (*struct*). Iter-RetGen keeps no carrier separate from its own generation, and its row has no history-removed cell. Bold is best in the row; * marks an OMS cell that beats *native* at $p < .05$, question-clustered bootstrap with Holm–Bonferroni over the 12 cells (Appendix F).

Domain	Harness	Native	History-rem.	+OMS (struct)	+OMS (free)
LoCoMo	IRCoT	63.0	57.2	63.6	**65.4***
	ReAct	65.1	59.4	65.7	**66.6**
	Iter-RetGen	**67.3**	–	65.0	65.9
	MemGPT	57.2	56.5	**65.4***	62.5*
MuSiQue	IRCoT	14.4	9.0	21.1	**25.9***
	ReAct	29.6	12.2	30.1	**36.8***
	Iter-RetGen	37.1	–	33.8	**39.6**
	MemGPT	31.4	30.6	29.6	**34.4**
SWE-QA-Pro	IRCoT	27.3	31.7	33.3	**34.6***
	ReAct	39.8	33.7	37.3	**43.0***
	Iter-RetGen	50.9	–	48.4	**51.4**
	MemGPT	**48.9**	43.6	42.3	47.7

	Accuracy	Correct abstention	
OMS − control	three domains	LoCoMo	MuSiQue
Model-managed state	$+0.9$	$\mathbf{+15.9}$	$\mathbf{+23.4}$
the policy owns UPDATE	$[-1.4, +3.2]$	$[+9.5, +22.3]$	$[+16.4, +30.2]$
Generic summarization	$+1.0$	-0.9	$+2.0$
the orchestrator owns it, different prompt	$[-0.7, +2.7]$	$[-5.9, +3.9]$	$[-2.3, +6.4]$

Table 3: **Who owns the update leaves accuracy alone and changes abstention.** Each cell is OMS minus a control in percentage points, above its 95% question-clustered bootstrap interval. We pool accuracy over 12 harness-by-domain cells and 300 questions; the abstention columns are LoCoMo's and MuSiQue's unanswerable strata, 440 episodes each, where a refusal is the right answer. Moving UPDATE inside the policy costs 16 to 23 pp of those refusals: on LoCoMo it refuses correctly on 49.1% against OMS's 65.0%. Rewording the orchestrator's extractor leaves both unchanged.

unanswerable strata of 440 questions. We report paired differences with 95% CIs from a cluster bootstrap over questions (Efron & Tibshirani, 1993), controlling the family-wise error rate with Holm–Bonferroni (Holm, 1979) in three prespecified families: the 12 cells, the ownership controls and the gate contrasts. For the gate we test *non-inferiority* against a prespecified margin of $\varepsilon = 2$ points, and do not test for improvement. An identical configuration re-run months later moved accuracy by 8.6 pp, so we re-ran every headline contrast in one window (Appendices B.1 and F.4).

6.1 ENFORCING THE UPDATE IMPROVES ACCURACY

Table 2 reports all 12 cells for the three arms of §6, and we read three results off it.

(1) Six cells gain significantly against the native harness, and none loses significantly. Freeform scores 54.8 against 51.8 on 2,035 questions, a gain of 3.0 pp (95% CI $[+2.2, +3.8]$, $p < 10^{-13}$). It improves 10 of 12 cells, and the two declines are at most 1.4 pp. The `gpt-4o-mini` alias moved between the native runs and ours, and against one snapshot OMS gains 4.4 pp ($[+2.5, +6.3]$) on 13.4% fewer tokens, so we treat the table as conservative (Appendices B.3 and B.10).

(2) All nine recoveries from history removal are significant. The recovered arm scores 54.8 against 48.2, a gain of 6.6 pp ($[+5.8, +7.5]$), and across the nine cells from $+2.9$ to $+24.6$ pp. When we re-run inside one window, deletion costs 6.3 pp and the state returns 8.4 (Appendix B.9). The exception is SWE-QA-Pro with IRCoT, where deletion *helps* by 4.4 pp. The arms share policy, retriever and budget, and differ only in whether the orchestrator supplies a carrier (Appendix B.2).

(3) Freeform beats the predicate schema. The fixed schema of §5.1 is significantly worse in 7 of 12 cells and better in 1. Its `open_predicates` field emits `no evidence` in 37.7–93.7% of episodes, against 0% for freeform. The schema requires a value in every round.

6.2 OWNERSHIP OF THE UPDATE CHANGES ABSTENTION

§6.1 shows that enforcing the update helps, but not whether the help comes from the orchestrator owning UPDATE or from the prompt merely containing a summary. We separate them with two controls, both in the freeform representation. Under *model-managed* state the policy gets the same state and an update call every round, and that call tells it the state is its own to curate. Under *generic summarization* we swap the OMS extractor for "summarize what you have learned so far".

Pooled accuracy is the same under either owner. In $1{,}188$ episodes over 300 questions the orchestrator-managed state scores 54.3 against 53.4, and we cannot separate the arms ($+0.9$ pp, 95% CI $[-1.4, +3.2]$, $p = .491$), while the policy-owned arm is 19.0% cheaper. Generic summarization agrees with that, 54.2 against 53.2 over $1{,}189$ episodes ($+1.0$ pp, CI $[-0.7, +2.7]$, Appendix B.4).

The state text is less faithful when the policy owns it. We scored every state against the passages its own episode retrieved. Notes the agent marks as verbatim quotations are absent from the corpus 34.1% of the time under model-managed state, against 8.9% under OMS and 5.5% under generic summaries, and the policy-written arm is worst in all 12 cells. The rate is already 38.6% in round one. Repeated overwriting does not accumulate it (Appendix B.5).

Ownership matters on unanswerable questions. We prespecified that retaining unresolved gaps (§5.1) should matter most when the correct answer is *"the corpus does not say"*, and Table 3 tests that prediction. The two strata are the datasets' own. LoCoMo supplies its category-5 adversarial questions, which ask about something the two speakers never discussed, and MuSiQue supplies multi-hop items whose corpus is missing the paragraph for one hop. An episode abstains correctly when its terminal answer is the literal token UNANSWERABLE. A hedged answer is scored at half credit, and any answer that supplies a specific person, date or file counts as the confabulation we report below. Moving UPDATE inside the policy costs 15.9 pp of correct abstention on LoCoMo and 23.4 pp on MuSiQue, significantly in all eight cells (SWE-QA-Pro has no unanswerable stratum), and we replicate the split on 75 fresh LoCoMo questions (Appendix B.6). Under model-managed state, the agent confabulates instead, on 41.6% of LoCoMo's unanswerable questions and 61.6% of MuSiQue's, against 25.5% and 37.0% for OMS. Under OMS the agent also refuses when it should not, on 19.0% of answerable questions against 13.5% under model-managed state, and the abstention gain is about three times that cost (Appendix B.4).

The effect holds when we control for the write mechanics. The two regimes differ on three things: the orchestrator merges new material into the state where the policy overwrites it in full, it reorganizes over capacity where the policy arm never does, and the two prompts ask for different things. We crossed the first two in a 2×2 on the same stratum, and neither mechanical factor favors OMS. The reorganizer has no effect (-0.2 pp, $[-2.8, +2.5]$, though it rarely fires under overwriting), and merging is worse than overwriting (-12.7 pp, $[-17.5, -8.1]$), which runs against OMS. One cell of the 2×2 matches OMS on both mechanics and differs from it only in keeping the model-managed instruction, and OMS still beats that cell by 32.5 pp ($[+25.9, +39.1]$, Appendix B.6). The wording does not explain it either, since generic summarization asks for something else and abstains at OMS's rate. In both of the arms that abstain well, the instruction the update follows was written somewhere other than the policy's own call. We never ran the reverse condition, an OMS-style extractor prompt placed inside that call, so we do not claim that our particular instruction is what matters. Our claim is the narrower one about where an instruction can come from: a harness can hand the update a specification the policy did not author only if UPDATE lives outside that call.

OMS matches three published update rules. We ran StateAct's goal/subgoal/state triple, FAIR-RAG's evidence checklist and SLIM's periodic compression in our UPDATE step, at matched model, format, capacity and update rate. In 590 paired questions OMS matches StateAct ($+0.8$ pp, CI $[-1.8, +3.5]$) and SLIM ($+1.7$ pp, $[-0.7, +4.1]$), and beats FAIR-RAG by 8.8 pp ($[+5.9, +11.7]$). The two nulls matter more: if published rules achieve comparable accuracy in the same position, the effects in §6.1 are not an artifact of our prompt (Appendix B.12).

6.3 ROBUSTNESS AND SCOPE

Other policies. On Llama-3.3-70B-Instruct the recovery transfers but shrinks. OMS improves 7 of 9 history-removed cells at a pooled paired gain of 5.8 pp, against 10.6 on a matched `gpt-4o-mini` re-run. Against *native* Llama harnesses it makes no difference (47.9 and 48.1), improving six of

twelve cells. Deletion costs the frontier policies more on SWE-QA-Pro and about the same on MuSiQue: for `gpt-5.6-terra`, 25.5 and 22.0 pp; for `gemini-2.5-pro`, 21.6 and 15.4; for `gpt-4o-mini`, 6.1 and 17.4. A bounded state returns the loss in all but one case, but gains over an *unmodified* frontier harness are policy-specific and we do not claim them (Appendix B.15).

The stopping rule. With OMS in place the frozen gate is non-inferior ($+0.3$ pp, worst case -1.1, inside the prespecified 2-point margin) and saves 16.7% of the tokens while interrupting 30% of episodes (Table 32). The same thresholds lose non-inferiority and save less when no module owns UPDATE, where Eq. 7's third conjunct is vacuous and the gate fires on 13.1 pp more episodes (CI $[+8.2, +18.1]$, $p < 10^{-6}$). OMS saves 2.3–30.7k tokens per question on MemGPT and costs more where the native loop is short. In both cases it caps the prompt length (Figure 3).

Chain depth. None of the three domains varies the quantity Observation 1 is about. Thus we built its chain task as a synthetic domain at $k = 2, \ldots, 6$, where each round exposes the evidence for one hop and the answer needs all k. Removing the history costs 18.3 pp at $k = 2$ and the whole task from $k = 3$ on. OMS returns 13.3 pp at $k = 2$, 8.3 at $k = 3$ and almost nothing beyond. The bound does not predict that collapse. The budget is far above the $k \log_2 N$ bits the chain needs, and the extractor stops writing the next hop into the state (Appendix B.13).

7 DISCUSSION

Findings. The paper's contribution is to take UPDATE out of the policy's LLM call and treat it as a module with two independent choices: how often it runs, and who writes the specification it follows. Enforcing the update every round recovers what deleting a harness's history costs. Writing it to a specification the policy does not author shifts abstention without moving pooled accuracy. We moved one UPDATE and one STOP module, unmodified, into four harnesses, three model families, and three published update rules. Had the effect appeared only inside a harness we wrote ourselves, we could not tell our code from the phenomenon we claim to have found. We scope the abstention claim to questions the corpus cannot answer, since abstention is the only place we tested who owns the update.

Checking a harness's design. The CGDP specializes a known model class, and Proposition 1 is not novel; we use both to check what a harness must represent, not as a new result. An omission then shows up as a specific defect in the design rather than just worse performance, one that prompt engineering alone cannot fix. Tomlinson et al. (2026) make the same argument, more precisely, for pointer chasing, which is our k-hop task in other notation. A reader might assume OMS only maintains memory across rounds and the gate only detects repeated queries, and for two of the gate's three conditions that assumption is correct. The third condition is not: it checks whether the state itself, Δz_t, stopped changing at round t, and nothing can compute that unless a module other than the policy's free-form prose owns UPDATE. A harness that only appends observations to the prompt has no such quantity to check (§6.3). ACT, OBSERVE, UPDATE and STOP describe what every harness already does; separating UPDATE into its own module is what lets us measure the one thing a harness without that separation cannot compute.

The design principle. We own the *operation* and leave the *representation* to the model. Dictating the schema the update fills loses in 7 of 12 cells (Appendix J), which supports the second half. The first half, owning the operation, does not hold everywhere: in one arm we replace the single search action with real repository tools, and there the policy's own state beats ours by 6.7 pp ($[+4.3, +9.2]$), because our extractor carries unresolved gaps into a domain in which nothing is unanswerable.

Limitations. Our domains are read-only. Definition 1's static-corpus assumption holds by construction, and we claim nothing about agents that write. Accuracy is a rubric judge's score, validated against gold answers and a judge from another family, though neither covers SWE-QA-Pro (Appendix F.3), and we reimplement published state rules instead of running them. We address the frontier arm's $n = 100$ cells, the frozen threshold, the lexical detectors and the extractor's losses in Appendix L. Two stay here. In the repository-tool arm above, deleting the history costs 17.7 pp, against 6.1 pp through search alone. And 55.9% of the policy's actions there repeat an earlier call, which no bounded state we tried repairs (Appendix B.14).

Reproducibility Statement

All harnesses, interventions, prompts and thresholds are specified in §5 and Appendices D–E; per-arm commands, dates and sample sizes are in Appendix B. These results use hosted model aliases whose behavior drifted by $+8.6$ pp on an identical configuration (Appendix B.3). Each reported contrast comes from arms run in the same window, and we recommend reproducing contrasts rather than absolute levels. Datasets are public, and we clone the 26 SWE-QA-Pro repositories at recorded commits. We will release code and run manifests with the camera-ready version.

Ethics Statement

This work evaluates read-only question answering over public corpora and introduces no new data collection involving human subjects. We designed the interventions to reduce unsupported assertions. On the unanswerable stratum, orchestrator-managed state lowered unsupported answers and raised abstention (§6.2), which we regard as the safety-relevant outcome. Conversely, a stopping monitor that interrupts search can suppress answers that further search would have found, and we report fire rates alongside accuracy for that reason. Our experiments consumed hosted-API inference, and we report the total token counts in §6.3.

The Use of Large Language Models

Large language models are the object of study in this paper: all agents, extractors, orchestrator gates and the rubric judge are LLMs, as specified in §6. Beyond that experimental role, the authors used an LLM-based coding assistant for routine software engineering on the experimental harness (refactoring, plotting and test scaffolding) and for copy-editing of prose, in both cases under author review. The research questions, the formulation in §3, the formal statements and proofs, the experimental design, the statistical plan, and all claims and their scoping are the authors' own. Every number reported in this paper was produced by the logged experimental pipeline, never by an AI assistant.

References

Mohammad Aghajani Asl, Majid Asgari-Bidhendi, and Behrooz Minaei-Bidgoli. FAIR-RAG: Faithful adaptive iterative refinement for retrieval-augmented generation. *arXiv preprint arXiv:2510.22344*, 2025.

Karl Johan Åström. Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10(1):174–205, 1965.

Songcheng Cai, Zhiheng Lyu, Yuansheng Ni, Xiangchao Chen, Baichuan Zhou, Shenzhe Zhu, Yi Lu, Haozhe Wang, Chi Ruan, Benjamin Schneider, Weixu Zhang, Xiang Li, Andy Zheng, Yuyu Zhang, Ping Nie, and Wenhu Chen. SWE-QA-Pro: A representative benchmark and scalable training recipe for repository-level code understanding. *arXiv preprint arXiv:2603.16124*, 2026.

Yangyi Chen, Lifan Yuan, Ganqu Cui, Zhiyuan Liu, and Heng Ji. A close look into the calibration of pre-trained language models. In *ACL*, pp. 1343–1367, 2023.

Herman Chernoff. Sequential design of experiments. *The Annals of Mathematical Statistics*, 30(3): 755–770, 1959.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. From local to global: A graph RAG approach to query-focused summarization. *arXiv preprint arXiv:2404.16130v2*, 2025. URL `https://arxiv.org/abs/2404.16130v2`.

Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*, volume 57 of *Monographs on Statistics and Applied Probability*. Chapman & Hall, 1993.

Shangbin Feng, Weijia Shi, Yike Wang, Wenxuan Ding, Vidhisha Balachandran, and Yulia Tsvetkov. Don't hallucinate, abstain: Identifying LLM knowledge gaps via multi-LLM collaboration. In *ACL*, pp. 14664–14690, 2024.

Aurélien Garivier and Emilie Kaufmann. Optimal best arm identification with fixed confidence. In *COLT*, pp. 998–1027, 2016.

Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. HippoRAG: Neurobiologically inspired long-term memory for large language models. In *NeurIPS*, 2024.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979.

Zhengbao Jiang, Frank Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *EMNLP*, pp. 7969–7992, 2023.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Ö. Arık, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. In *COLM*, 2025.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998.

Yoonsang Lee, Howard Yen, Xi Ye, and Danqi Chen. Agentic aggregation for parallel scaling of long-horizon agentic tasks. In *COLM*, 2026.

Jiaqi Li, Mengmeng Wang, Zilong Zheng, and Muhan Zhang. LooGLE: Can long-context language models understand long contexts? In *ACL*, pp. 16304–16333, 2024.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. In *EMNLP*, pp. 5420–5438, 2025a.

Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yongkang Wu, Ji-Rong Wen, Yutao Zhu, and Zhicheng Dou. WebThinker: Empowering large reasoning models with deep research capability. In *NeurIPS*, 2025b.

Hao Liu, Zhengren Wang, Xi Chen, Zhiyu Li, Feiyu Xiong, Qinhan Yu, and Wentao Zhang. HopRAG: Multi-hop reasoning for logic-aware retrieval-augmented generation. In *ACL Findings*, pp. 1897–1913, 2025.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

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*, pp. 13851–13870, 2024.

Elliot Meyerson, Giuseppe Paolo, Roberto Dailey, Hormoz Shahrzad, Olivier Francon, Conor F. Hayes, Xin Qiu, Babak Hodjat, and Risto Miikkulainen. Solving a million-step LLM task with zero errors. *arXiv preprint arXiv:2511.09030*, 2025.

Niels Mündler, Jingxuan He, Slobodan Jenko, and Martin Vechev. Self-contradictory hallucinations of large language models: Evaluation, detection and mitigation. In *ICLR*, 2024.

Mohammad Naghshvar and Tara Javidi. Active sequential hypothesis testing. *The Annals of Statistics*, 41(6):2703–2738, 2013.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023. URL https://arxiv.org/abs/2310.08560.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *UIST*, 2023.

Nikolai Rozanov and Marek Rei. StateAct: Enhancing LLM base agents via self-prompting and state-tracking. In *REALM*, pp. 367–385, 2025.

Daniel Russo. Simple Bayesian algorithms for best arm identification. In *COLT*, pp. 1417–1418, 2016.

Tobias Schnabel, Kiran Tomlinson, Adith Swaminathan, and Jennifer Neville. Lost in transmission: When and why LLMs fail to reason globally. In *NeurIPS*, 2025.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In *EMNLP Findings*, pp. 9248–9274, 2023.

Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. DRAGIN: Dynamic retrieval augmented generation based on the real-time information needs of large language models. In *ACL*, pp. 12991–13013, 2024.

Jayakumar Subramanian, Amit Sinha, Raihan Seraj, and Aditya Mahajan. Approximate information state for approximate planning and reinforcement learning in partially observed systems. *Journal of Machine Learning Research*, 23(12):1–83, 2022.

Kiran Tomlinson, Tobias Schnabel, Adith Swaminathan, and Jennifer Neville. Reasoning about reasoning: BAPO bounds on chain-of-thought token complexity in LLMs. In *ICML*, 2026.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *ACL*, pp. 10014–10037, 2023.

Yingsheng Wu, Yuxuan Gu, Xiaocheng Feng, Weihong Zhong, Dongliang Xu, Qing Yang, Hongtao Liu, and Bing Qin. Extending context window of large language models from a distributional perspective. In *EMNLP*, pp. 7288–7301, 2024.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-Mem: Agentic memory for LLM agents. In *NeurIPS*, 2025.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *NeurIPS*, pp. 50528–50652, 2024.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *ICLR*, 2023.

Howard Yen, Yoonsang Lee, Ashwin Paranjape, Mengzhou Xia, Thejas Venkatesh, Jack Hessel, Danqi Chen, and Yuhao Zhang. Lost in the maze: Overcoming context limitations in long-horizon agentic search. In *COLM*, 2026.

A PROOFS

Throughout, $\big(\mathcal{Q}, \mu, \mathcal{C}, \mathcal{A}_Q, \mathcal{O}, P, \mathcal{Y}, R, \kappa, H, \lambda\big)$ is a CGDP as in Definition 1, with $\mathcal{A}_Q$, $\mathcal{O}$ and $\mathcal{Y}$ finite, $R(q, c, y) \in [0, 1]$ and $\kappa(b, a) \geq 0$. Throughout, the task q is fixed and suppressed from the notation where it plays no role.

The CGDP is an instance of sequential identification with costs (Chernoff, 1959; Naghshvar & Javidi, 2013; Russo, 2016; Garivier & Kaufmann, 2016), in which queries are paid for and the latent object is a corpus rather than a hypothesis index, and z_t is an *approximate information state* in the sense of Subramanian et al. (2022). In that language, Observation 2 bounds an approximation that is not decision-sufficient.

A.1 Proposition 1: the optimal value recursion

We restate Proposition 1 of §3.1 and prove it.

The statement is the finite-horizon POMDP recursion (Åström, 1965; Kaelbling et al., 1998), specialized to our restrictions. We give the derivation for completeness, and to make explicit which restriction is used where.

Proof. Step 1: the posterior is a sufficient statistic. By Definition 1 the latent corpus does not depend on t, so the joint law of (c, h_t) factors as $\mu(c \mid q)\prod_{s<t} P(o_s \mid a_s, c)$ times the (policy-determined) probabilities of the actions, which do not depend on c given the history. Hence for any history h_t,

$$b_t(c) \;=\; \mathbb{P}(c \mid h_t) \;=\; \frac{\mu(c \mid q)\prod_{s<t} P(o_s \mid a_s, c)}{\sum_{c' \in \mathcal{C}} \mu(c' \mid q)\prod_{s<t} P(o_s \mid a_s, c')},$$

which is Eq. 2 and is independent of the policy that generated the actions. The conditional law of the next observation is $P(o \mid b_t, a) = \mathbb{E}_{c\sim b_t}[P(o \mid a, c)]$, and Bayes' rule gives the deterministic update $b_{t+1} = \tau(b_t, a_t, o_t)$ with

$$\tau(b, a, o)(c) \;=\; \frac{P(o \mid a, c)\, b(c)}{\sum_{c'} P(o \mid a, c')\, b(c')} \qquad \text{whenever } P(o \mid b, a) > 0.$$

Therefore the process $(b_t)_t$ is a controlled Markov chain on the belief simplex whose transition law and rewards depend on h_t only through b_t. The terminal reward has conditional expectation $\mathbb{E}_{c\sim b_t}[R(q, c, y)]$, and the immediate cost $-\lambda\kappa(b_t, a)$ depends on h_t only through b_t. The original problem is thus an MDP on beliefs with a horizon of H query actions.

Step 2: backward induction. Let $V_r(b)$ denote the supremum over policies of the expected value in Eq. 1 from a state with belief b and r query actions remaining. With $r = 0$ no query is permitted, so the only admissible decision is a terminal answer, and $V_0(b) = \max_{y\in\mathcal{Y}} \mathbb{E}_{c\sim b}[R(q, c, y)]$. The maximum is attained because $\mathcal{Y}$ is finite. For $r \geq 1$ the admissible decisions partition into (i) answering now, with value $\max_y \mathbb{E}_{c\sim b}[R(q, c, y)]$, and (ii) issuing some $a \in \mathcal{A}_Q$, paying $\lambda\kappa(b, a)$, observing $o \sim P(\cdot \mid b, a)$ and continuing optimally from $\tau(b, a, o)$ with $r - 1$ queries remaining, whose value is $-\lambda\kappa(b, a) + \sum_o P(o \mid b, a)V_{r-1}(\tau(b, a, o))$ by the induction hypothesis and the tower property. Since rewards are bounded ($R \in [0, 1]$, $\kappa \geq 0$, $H < \infty$) all expectations are finite and the interchange of supremum and expectation is justified by the finiteness of $\mathcal{A}_Q$ and $\mathcal{O}$. Taking the maximum over the two groups of decisions gives Eq. 3. □

Two remarks that the main text uses. First, the optimal stopping rule is *exactly* the comparison inside Eq. 3: continuing is optimal iff some action's expected improvement in terminal value exceeds $\lambda\kappa(b, a)$. A stopping rule that conditions on neither cost nor expected novelty is therefore misspecified in both directions — it can stop while a cheap informative query remains and continue while every available query is redundant. Second, Step 1 is the only place we use the static-corpus restriction, and it is essential. With environment-modifying actions the factorization above fails, the posterior over a *time-varying* c_t needs a transition kernel $P(c' \mid c, a)$, and Eq. 2 is no longer valid.

A.2 Observation 2: the cost of state aliasing

We restate the setting precisely. An encoder is a map ϕ from histories to a set $\mathcal{Z}$. A *z-policy* is any measurable map $\nu : \mathcal{Z} \times \{0, \ldots, H\} \to \Delta(\mathcal{A}_Q \cup \mathcal{Y})$: its input is the encoded state and the remaining horizon, and it may randomize. For a history h with r query actions remaining, let $Q(h, a)$ be the value of taking decision a at h and continuing optimally, so that the regret of decision a at h is $\mathrm{reg}(h, a) = \max_{a'} Q(h, a') - Q(h, a) \geq 0$.

Observation 2 (State-aliasing lower bound)**.** *Fix a CGDP, an encoder ϕ, and two histories h, h' with the same remaining horizon and $\phi(h) = \phi(h')$. Suppose the optimal continuation is unique at each and they differ, and that at both the optimal action exceeds every other action's Q-value by at least $\Delta > 0$. Then every z-policy, randomized included, suffers expected regret at least $\Delta/2$ under the equal mixture of the two.*

Proof. Let h, h' have the same remaining horizon r and satisfy $\phi(h) = \phi(h') = z$. Let $a^\star$ and $a'^\star$ be their unique optimal decisions, with $a^\star \neq a'^\star$, and suppose the action gap is at least Δ at both, i.e. $\mathrm{reg}(h, a) \geq \Delta$ for every $a \neq a^\star$ and $\mathrm{reg}(h', a) \geq \Delta$ for every $a \neq a'^\star$.

A z-policy receives the same input at both histories, hence plays the same distribution $\nu := \nu(z, r) \in \Delta(\mathcal{A}_Q \cup \mathcal{Y})$ at both. Its expected regret at h is

$$\mathbb{E}_{a\sim\nu}\big[\mathrm{reg}(h, a)\big] \;\geq\; \Delta\big(1 - \nu(a^\star)\big),$$

because every decision other than $a^\star$ incurs at least Δ and $\mathrm{reg} \geq 0$. Symmetrically, the expected regret at h' is at least $\Delta(1 - \nu(a'^\star))$. Under the mixture ρ that places probability $1/2$ on each history, the expected regret is therefore at least

$$\tfrac{1}{2}\Delta\big(1 - \nu(a^\star)\big) + \tfrac{1}{2}\Delta\big(1 - \nu(a'^\star)\big) \;=\; \Delta\Big(1 - \tfrac{\nu(a^\star)+\nu(a'^\star)}{2}\Big) \;\geq\; \frac{\Delta}{2},$$

where the last inequality uses $\nu(a^\star) + \nu(a'^\star) \leq 1$, valid because $a^\star \neq a'^\star$ and ν is a probability distribution. Randomization does not help: the bound holds for every ν, and it is attained by $\nu(a^\star) = \nu(a'^\star) = 1/2$. □

The bound is tight in the stated setting, since the even split attains it. Its force is structural: the loss is bounded below by a quantity that no downstream component ever sees, because the policy, the prompt, and any amount of test-time computation act only on $\phi(h_t)$. In §5.1 we use this as follows. The update U must preserve the distinctions the continuation decision depends on: which sub-questions remain open, which queries were already issued, and where evidence came from. These are precisely the coordinates that separate h from h' in typical context-gathering instances. A raw-append state with truncation (Eq. 5) offers no such guarantee. Two histories that differ only in an evicted early round encode identically.

Approximate decision sufficiency. Since the exact condition in §3.2 is brittle, the usual relaxation applies here. Suppose ϕ is an ϵ-approximate information state in the sense of Subramanian et al. (2022), so that its induced reward and transition predictions are within ϵ in the appropriate metrics. Then the policy computed from ϕ is $O(H\epsilon)$-optimal. Observation 2 is the complementary statement: if ϕ collapses histories whose optimal continuations differ by Δ, then the two bounds together force $\epsilon = \Omega(\Delta/H)$, so a guarantee of that form cannot be tight for such a ϕ. We do not claim that our implemented update is an ϵ-approximate information state for any particular ϵ. We use the pair of statements to motivate which distinctions the update has to retain.

A.3 Observation 1: compositional memory capacity

We restate Observation 1 of §3.2 and prove it.

Proof. Let $X = (X_1, \ldots, X_k) \sim \mathrm{Unif}([N]^k)$, so X takes N^k values uniformly. By assumption the evidence identifying X_i is observable only in round i, the only quantity carried from round i to round $i + 1$ is a persistent state $s_i \in \mathcal{S}$ with $|\mathcal{S}| \leq 2^m$, and the answer is $\hat{X} = g(s_k)$ for a fixed map $g : \mathcal{S} \to [N]^k$. Then

$$\begin{aligned}\mathbb{P}(\hat{X} = X) \;&=\; \sum_{x\in[N]^k} \mathbb{P}(X = x)\,\mathbb{P}\big(g(s_k) = x \mid X = x\big) \\ &\leq\; \frac{1}{N^k}\,\big|\{x : \exists s \in \mathcal{S},\; g(s) = x\}\big| \;\leq\; \frac{2^m}{N^k},\end{aligned}$$

since $\hat{X}$ can only take values in the image of g, which has at most $|\mathcal{S}| \leq 2^m$ elements. (If the agent randomizes, condition on its random seed and average: the bound holds for every realization of g and $\mathcal{S}$, hence in expectation.) Requiring $\mathbb{P}(\hat{X} = X) \geq 1 - \delta$ gives $2^m \geq (1 - \delta)N^k$, i.e. $m \geq k \log_2 N + \log_2(1 - \delta)$, which is Eq. 6. □

Three clarifications, since the statement is easy to over-read. **(i)** It constrains *persistent* capacity, and the context window is a separate matter. An agent that can re-observe hop-i evidence in a later round falls outside the hypothesis, so we state it for evidence visible in one round only. Re-observation is exactly what a retrieval agent gets by issuing another query, and it is why bounded-state agents can still

solve long chains at the cost of extra rounds. **(ii)** It is information-theoretic and therefore indifferent to how the m bits are arranged — JSON, prose, or embeddings. That indifference limits the scope of the statement, and it is not a prediction our runs confirm. Dictating a predicate schema did change accuracy, for the worse in 7 of 12 cells (§6.1), while moving ownership of the update left pooled accuracy where it was (§6.2). Neither effect is visible to Eq. 6. **(iii)** The readout hypothesis does real work. We take $\hat{X} = g(s_k)$, which is what OMS does, since the orchestrator discards o_t once the extractor has run. An agent that answers from the final state *and* the last observation, $\hat{X} = g(s_k, o_k)$, need never carry X_k, and the same counting then gives only $m \geq (k-1)\log_2 N + \log_2(1-\delta)$. This is weaker by one hop and has the same form. **(iv)** It says nothing about achievability: $m = k \log_2 N$ bits suffice only if the update is lossless, and an LLM extractor is not. We read Eq. 6 as a scaling law. The state budget must grow at least linearly in the number of unresolved hops, so a fixed item budget K assumes that chains are short. In Appendix B.13 the capacity $\times$ chain-length arm tests that bet directly, and the failure arrives earlier than the bound requires. Recovery is gone by $k = 6$ at every $K \in \{3, 6, 12\}$, so on this task the binding constraint is the lossiness of U rather than the budget that Eq. 6 bounds.

The distance between the two is what Tomlinson et al. (2026) isolate for pointer chasing, which is the k-hop task in another notation. Pointer chasing is BAPO-hard, so a model whose attention is split cannot answer it in one pass (Schnabel et al., 2025). A chain of thought does solve it, in a number of tokens that grows with the length of the chain, however large the input is, and our m bits play the part of that trace. Eq. 6 says how wide the trace has to be, and it says nothing about whether an LLM extractor will fill it.

B Protocol, Statistical Plan, and Supporting Runs

B.1 Statistical plan

Unit of analysis. The unit is the *question*. Several harnesses produce multiple episodes per question (e.g. across conditions or seeds), so episode-level tests would treat correlated outcomes as independent. We aggregate to the question before testing, and report question-level n alongside every contrast. If we also report an episode-level test we say so. In every arm we ran the two agree in direction; the cross-model recovery, for instance, gives $p = .001$ at the episode level and $p = .0014$ at the question level.

Intervals. All confidence intervals are 95% percentile intervals from a cluster bootstrap that resamples questions with replacement ($10{,}000$ draws), keeping all episodes of a resampled question together. We do this to respect the pairing within a question, and the clustering that harness $\times$ condition replication induces.

Multiplicity. We fixed three families of tests before running the additional arms, and we correct within each family using Holm–Bonferroni. (F1) is the 12 harness $\times$ domain state cells. (F2) is the ownership controls, meaning the two matched controls and the prespecified unanswerable stratum. (F3) is the gate contrasts, meaning the two frozen-threshold arms and their interaction. We correct per-cell p-values reported inside a family. Descriptive per-harness breakdowns inside a single contrast (e.g. the four per-harness values on the unanswerable stratum) are uncorrected, and we label them descriptive.

Non-inferiority. For the gate we test non-inferiority instead of superiority, because the intervention's purpose is to save tokens. We prespecified $\varepsilon = 2$ accuracy points, one-sided at $\alpha = .05$, and declare non-inferiority when the lower end of the two-sided 95% interval for the accuracy change lies above $-\varepsilon$. The margin is a judgment call. However, 2 points is smaller than every significant state effect in Table 2, and smaller than the run-to-run drift documented below, so a gate that stayed within it could not erase a state gain.

Matched arms. Each contrast is computed within a single run window against a comparison arm executed in that window (see §B.3). We do not compare a new condition against a stored baseline from an earlier window.

B.2 THE FOUR COMPARATORS

The paper reports gains against four different comparators, and which one is in play changes what a gain means, so we collect them here. Row 2 is the cleanest contrast we have, since the two arms share policy, retriever and retrieval budget and differ only in whether the orchestrator supplies a carrier. Row 1 is the one a practitioner would weigh, because it asks whether the state is worth adding to a harness that already works. The commands in this appendix use the condition names the code carries: `baseline` is the native harness, `lobotomized` is the same harness with its cross-round carrier deleted, and a `_st_` suffix marks a run with OMS supplying the state.

Table 4: Comparison inventory. Every headline number in the paper, with the arm it is measured against. All rows are `gpt-4o-mini` unless the treatment column names another policy.

	Treatment arm	**Comparator arm**	**Where**
$+3.0$	History-removed harness + OMS	The same harness with its native memory intact	§6.1
$+6.6$	History-removed harness + OMS	The same harness with the history removed and nothing put back	§6.1
-6.3, $+8.4$, $+2.0$	Those same three arms, re-run together in one window	Each other	§B.9
$+4.4$	The freeform state, re-run against native in one window	The same harness with its native memory intact	§B.10
$+45.5$, $+44.2$, $+32.0$	The strongest same-window arm in each domain	A policy shown no corpus at all	§B.11
$+0.9$	OMS	Model-managed state, in the same step with the same instructions	§B.4
$+1.0$	OMS	Generic summarization in the same step	§B.4
$+10.8$	OMS	Model-managed state, on LoCoMo's category-5 questions only	§B.6
$+16.0$, $+0.7$	OMS, in a three-arm replication of that stratum	Model-managed state, and generic summarization	§B.6
$+0.8$, $+1.7$, $+8.8$	OMS	StateAct, SLIM and FAIR-RAG in the same step at matched budget	§B.12
-6.7	OMS	Model-managed state, on the read-only tool interface	§B.14
$+5.8$, -0.2	Llama-3.3-70B history-removed + OMS	History-removed Llama, and native Llama	§B.7
$+13.3$ at $k = 2$	OMS on the synthetic chain	The same chain with the history removed	§B.13
Loss returned	History-removed frontier policy + OMS	The same frontier policy with the history removed	§B.15
-1 to -5	Intact frontier harness + OMS	The same harness without it	§B.15
$+0.3$, $+1.6$	Frozen stagnation gate	A fixed-round budget, and the same gate without a state	§B.8
$+8.6$	One condition, unchanged	Itself, three months earlier	§B.3

The rows that set OMS against model-managed state, against generic summarization and against the three published update rules hold the update operation fixed and vary only who performs it. They therefore speak to ownership alone; they do not test whether a bounded state helps at all. The last row gives no treatment effect. It is one condition re-run against a moved alias, and it sets the scale against which every other row should be read.

B.3 ARM REGISTRY AND ENDPOINT DRIFT

Hosted model aliases are not fixed artifacts. When we re-ran one identical configuration in a later window, accuracy moved by $+8.6$ pp ($p = .024$, $n = 99$ questions) and the mean number of query rounds fell from 3.96 to 2.97. The harness, prompts, retriever, seeds and thresholds were the same, and only the date differed on the same `gpt-4o-mini` alias. The drift is larger than several effects we report. For that reason every table below names its comparison arm, and Table 5 lists the arms and their windows.

Table 5: Every run reported in this paper, with the window it was executed in. Every contrast in the main text is computed inside one row's window, against that row's comparison arm. n is questions; the Llama model is Llama-3.3-70B-Instruct, and e6-tools runs SWE-QA-Pro over the repository checkouts rather than the chunked corpus. An arm name in this table names a contrast. The underscored tags in the commands below, such as `t1_control` and `e1_endpoints`, name the stored runs, and a contrast is computed inside one of them, so a single tag can carry several arms.

Arm	Policy model	Conditions compared	Domains	n
t1-own	`gpt-4o-mini`	OMS vs. model-managed state	all three	1188
t1-gen	`gpt-4o-mini`	OMS vs. generic summarization	all three	1189
t3-unans	`gpt-4o-mini`	OMS vs. model-managed, unans. only	LoCoMo	110
t2-llama	Llama-3.3-70B	OMS vs. history-removed / native	all three	539
t2-ref	`gpt-4o-mini`	OMS vs. history-removed (matched)	all three	535
p2-gate	Llama-3.3-70B	gate on/off $\times$ state on/off	all three	1079
e1-endpoints	`gpt-4o-mini`	native / hist.-removed / +OMS, one window	all three	100
b2-native	`gpt-4o-mini`	native vs. +OMS (freeform), one window	all three	100
e4a-ksweep	`gpt-4o-mini`	state capacity $K \in \{3, 6, 12\}$, free-form	all three	100
e5-matched	`gpt-4o-mini`	StateAct / FAIR-RAG / SLIM in our update step	all three	100
e4b-khop	`gpt-4o-mini`	four states vs. history-removed, k=2 . . . 6	k-hop chains	300
e6-tools	`gpt-4o-mini`	six regimes, read-only tool interface	SWE-QA-Pro	260
t6-frontier	`gpt-5.6-terra` `gemini-2.5-pro`	six regimes	all three	100

B.4 Ownership controls

Both controls keep the state in the same place and vary only who writes it, so the comparison isolates ownership of UPDATE, since state-shaped text is in the prompt either way. In *model-managed* state we instruct the policy to maintain the same bounded state inside its own single call. In *generic summarization* the orchestrator still makes the extra call, but with a generic "summarize what you have learned so far" prompt in place of the OMS extractor prompt. The update is then not conditioned on the question, and it carries no requirement to retain unresolved gaps. Table 6 reports both controls.

Table 6: Ownership controls. Accuracy is the rubric score; Δ is OMS minus control, positive favoring orchestrator ownership, with the cluster-bootstrap interval over questions. False abstention is the share of *answerable* questions the arm declines, OMS against the control. Tokens are OMS relative to the control, mean total per episode including orchestrator calls. Neither accuracy contrast is conclusive at these sample sizes.

Control	n	OMS	Control	Δ (pp)	95% CI	False abst.	Tokens
Model-managed state	1188	54.3	53.4	+0.9 (p=.491)	$[-1.4, +3.2]$	19.0 vs. 13.5	+23.8%
Generic summarization	1189	54.2	53.2	+1.0 (p=.249)	$[-0.7, +2.7]$	19.0 vs. 18.8	−4.3%

Only two cells reach significance against model-managed state, and they point in opposite directions: LoCoMo/ReAct $+10.2$ pp ($p = .018$) and SWE-QA-Pro/IRCoT -3.4 pp ($p = .019$). Set against generic summarization, the only significant cell is SWE-QA-Pro/IRCoT at $+5.0$ pp ($p = .001$). Model-managed state is 19.0% cheaper, $10{,}158$ tokens against $12{,}533$ over the $1{,}185$ episodes all three arms answered. It is not cheaper because it skips a call. Every arm runs an update call in every round, and the reorganizer that only OMS runs is 558 tokens, under a quarter of the $2{,}375$-token gap. The remainder is in the policy's own turns ($1{,}117$ tokens) and in the update call itself (700), and these runs do not tell us why either is shorter. Generic summarization costs about the same as OMS (4.4% more), which is the point of the control. We do not claim a pooled accuracy advantage for orchestrator ownership. Ownership instead gives us the invariants of §5.1 by construction, since the size bound and the update rule do not depend on the policy cooperating. In our runs we see this in how well calibrated the answers are, and in the fidelity of the state text (§B.5), but not in pooled accuracy.

B.5 State fidelity under the three ownership regimes

Table 6 shows that ownership does not move pooled accuracy, and §B.4 argues that what it changes instead is whether the invariants hold. Since the invariants are properties of the state text, we score that text directly and offline, on the runs already stored under the `t1_control` tag: all 36 cells,

3,535 episodes, four harnesses × three domains × three ownership regimes. This involves no new rollouts and no judge calls. As ground truth we use the retriever cache the episodes actually drew from, never a re-download that might have drifted.

We measure three quantities, all deterministic and requiring no model calls. *Quotation fidelity* uses the fact that when a note is marked as copied out of a passage and is at least 25 characters long, we can check directly whether the corpus contains that literal string. If it does not, the agent has fabricated an observation, and the state then carries it into every later round as though it were evidence. *Paraphrase grounding* is the fraction of unmarked notes whose content words are recoverable at recall ≥ 0.6 from some passage the episode retrieved. Since a paraphrase legitimately compresses its source, we use this measure only to compare regimes against each other, never as an absolute level. It is low on SWE-QA-Pro because notes there describe code without restating prose. *Negative-evidence retention* is the fraction of episodes with at least one unproductive round – a round that surfaced nothing the agent had not already seen – in which the state explicitly records that nothing was found. In Table 31 this is the behavior the predicate schema fails at.

Table 7: State fidelity by ownership regime, pooled over the four harnesses. *Quotations* counts marked verbatim notes and *unverified* the fraction of them absent from the corpus. *Grounded* is the fraction of paraphrased notes recoverable from a retrieved passage at recall ≥ 0.6. *Neg. ret.* is negative-evidence retention among episodes with an unproductive round.

Domain	Regime	Episodes	Quotations	Unverified	Grounded	Neg. ret.
LoCoMo	OMS (append-only)	391	805	4.1%	55.8%	1.4%
	Generic summary	400	934	2.9%	35.7%	1.1%
	Model-managed	400	271	11.4%	44.0%	10.0%
MuSiQue	OMS (append-only)	377	442	13.3%	42.6%	1.8%
	Generic summary	400	465	10.1%	35.3%	6.1%
	Model-managed	400	181	60.2%	40.2%	15.1%
SWE-QA-Pro	OMS (append-only)	367	634	12.0%	9.6%	9.5%
	Generic summary	400	383	6.3%	6.5%	7.8%
	Model-managed	400	150	43.3%	6.0%	4.6%
Pooled	OMS (append-only)	1,135	1,881	8.9%	33.4%	4.1%
	Generic summary	1,200	1,782	5.5%	24.9%	5.0%
	Model-managed	1,200	602	34.1%	28.1%	7.7%

This quotation result holds in every cell we measured. Model-managed state has the highest unverified-quotation rate in all 12 (domain, harness) cells, pooling to 34.1% against 8.9% for OMS and 5.5% for generic summarization. It also emits roughly a third as many verbatim quotations in the first place (602 against 1,881 and 1,782). These two both follow from allowing the state to be replaced. When the policy rewrites its whole state each round it quotes the corpus less often, and about a third of what it does quote cannot be found there. An append-only state cannot overwrite what it has already committed, so a quotation entered in round 2 is still the same string in round 6. Observation 2 bounds what a lossy update costs in the aggregate, and fabricated quotations are one way to see it: a state that can be rewritten drifts toward a description that no longer distinguishes the histories that produced it. The two orchestrator-owned regimes differ from each other by 3.4 pp on this measure, and from the model-managed one by an order of magnitude. This is what we mean in §B.4 when we say that ownership makes the invariants enforceable.

However, two of our measures do not support this reading, and we report them here. Paraphrase grounding favors OMS when pooled (33.4% against 24.9% and 28.1%) but not within every domain, and since the measure is a lexical proxy we cannot read a fabrication rate off it. Negative-evidence retention runs the other way: model-managed state records absences somewhat more often (7.7% against 4.1% and 5.0%). Any policy free to write down what it finds salient would do that, and we do not claim the measure favors orchestrator ownership. We rest the argument on the quotation rate, the only one of the three with an unambiguous ground truth.

B.6 The unanswerable stratum

We prespecified this subgroup on the reasoning in §5.1: retaining unresolved gaps should matter most when the correct answer is that the corpus does not contain one. LoCoMo labels such questions, so

we evaluate them separately. Here we compare against model-managed state, because we want to know who should own UPDATE when both arms already carry a bounded state. To guard against a subgroup found by inspection, we ran the 35 labeled questions that fell inside the t1 window first, and then drew an independent set of 75 from the questions that draw had not touched. Table 8 reports both draws and the 110-question union.

Table 8: LoCoMo unanswerable questions, OMS against model-managed state. Both arms run in the same window on the same questions, so this contrast is between ownership regimes, with no native harness in it. The initial and subsequent draws are independent question sets; the pooled row is their union. Behavioral margins are computed on the pooled set.

Draw	n	**OMS**	**Model-mgd.**	**Δ (pp)**	**95% CI**
Initial	35	73.9	62.1	+11.8 (p=.056)	[−0.3, +23.9]
Subsequent (out of sample)	75	69.5	59.2	+10.3 (p=.001)	[+4.3, +16.4]
Pooled	110	70.9	60.1	+10.8 (p<.001)	[+5.3, +16.3]

Per-harness values on the out-of-sample draw (descriptive, uncorrected): IRCoT $+13.3$ ($p = .005$), MemGPT $+12.7$ ($p = .012$), ReAct $+10.7$ ($p = .059$), Iter-RetGen $+4.7$ ($p = .403$). As the mechanism predicts, the behavioral margins move: abstention rises from 55.0% to 66.3%, and answers that assert an unsupported fact fall from 36.7% to 27.3%. Iter-RetGen gives the smallest effect and is the one harness with no cross-round carrier to replace. The mechanism fits, although it is a single cell.

A replication with the generic control in it. In both draws above OMS runs against model-managed state, which leaves open whether the margin comes from who owns the update or from the extractor prompt we happened to write. To separate the two we re-ran the whole stratum in a fresh window with generic summarization added as a third arm, on the same 110 questions and all four harnesses, and the two contrasts come apart. In this draw OMS gains 16.0 pp of stratum score on model-managed state ($[+9.9, +22.0]$, $p < .001$, 440 episodes over 110 questions), which is the $+15.9$ pp of correct abstention ($[+9.5, +22.3]$) that §6.2 reports. That is larger than the 10.8 pp of Table 8, and it is significant in every one of the four cells. For generic summarization the gain is 0.7 pp ($[-3.4, +4.9]$, $p = .747$), and the four cells run from -7.3 to $+11.8$ pp. The abstention profile tracks the accuracy: correct abstention is 65.0% for OMS and 65.9% for generic summarization, while model-managed state reaches 49.1% and confabulates on 41.6% of these questions against roughly 26% for either orchestrator-owned arm. The two arms that gain are written by the orchestrator, and the one that loses is written by the policy. This result should be held next to two others. In §B.11 a policy given no corpus at all scores 94.3 on these same questions, since abstaining is the correct answer to them. The false-abstention column of §B.4 then gives what the caution costs on questions that do have an answer.

Deconfounding the ownership effect. The comparison above still confounds three things. The orchestrator merges each extraction into the state and deduplicates it, while the model-managed arm receives the whole state and returns a full rewrite. The orchestrator reorganizes the state once it exceeds capacity, and the rewrite arm skips that step because a rewrite is already a reorganization. The two prompts also ask for different work: OMS asks for the facts a passage states, and the managed prompt tells the model that it owns the memory and should keep what it finds useful. The OMS-against-summary null already argues against the third factor on its own, so we crossed the first two in a 2×2 and held the ownership wording fixed in all four cells. The merge cells use the same append-and-deduplicate path as OMS and differ from it only in the prompt. We ran all four cells plus the OMS reference on the same 110 questions and the same four harnesses, on `gpt-4o-mini` through the model gateway, for 2,200 episodes at no API cost. Table 9 gives the operating point.

M1 is the arm the main comparison uses, and it reproduces: OMS beats it by 19.5 pp of correct abstention ($[+13.4, +25.9]$, $p < .001$, question-clustered bootstrap, design effect 1.29), against the 16.0 pp of the previous draw and the 49.1% that arm reached there. Neither crossed factor accounts for it. The reorganizer main effect is -0.2 pp ($[-2.8, +2.5]$, $p = .870$), and it is null within each level of the write factor. That null is weaker than it looks on the rewrite side, since the reorganizer only fires once the state exceeds K_{trigger} and a rewriting arm rarely gets there: over the 330 episodes per cell it fires 48 times for OMS, 26 for M4, and once for M2. The write main effect runs the

Table 9: Mechanism 2×2 on the LoCoMo unanswerable stratum, 440 episodes per cell over 110 questions and four harnesses. The ownership wording is identical in M1–M4. Correct abstention is a literal UNANSWERABLE; a hedge scores 0.5.

cell	write	reorganizer	prompt	abstain	hedged	confabulated	rubric
OMS	merge	on	extract	66.8	7.5	25.7	70.6
M1	rewrite	off	own it	47.3	11.4	41.4	53.0
M2	rewrite	on	own it	46.8	8.6	44.5	50.9
M3	merge	off	own it	34.3	8.2	57.5	38.4
M4	merge	on	own it	34.3	8.0	57.7	38.3

wrong way, since merging costs 12.7 pp against rewriting ($[-17.5, -8.1]$, $p < .001$) and does so at both levels of the reorganizer. M4 was built as a positive control, because it merges and reorganizes exactly as OMS does and only the prompt differs, so we expected it to be level. It is 32.5 pp below OMS ($[+25.9, +39.1]$, $p < .001$), the largest contrast in the arm, and the rubric score gives the same picture (+32.3 pp). The remaining difference is the content of the update instruction. The arms that abstain well ask the extractor to record what the passages state, and the four that abstain badly ask the model to decide for itself what to keep, which is the same direction as the fidelity gap of §B.5.

Two caveats. OMS against M4 is not a clean prompt-framing manipulation, because the OMS extractor also reasons in a scratchpad, is told not to repeat what the state already holds, and has an explicit "Nothing relevant" escape that the managed prompts lack. Any of those could carry part of the 32.5 pp. Separately, merging without an ownership prompt is what OMS already does, so the negative write effect says that a merge path works badly when the model is also told to curate, and it says nothing against merging as such.

Reproduction:

```
bash run_t3_unans.sh
bash judge.sh --tag t3_unans --tasks locomo --workers 12
python t3_analysis.py
python abstention_profile.py t3_unans

bash b1_unans_summary.sh
bash judge.sh --tag b1_unans --tasks locomo --workers 12
python t1_analysis.py --tag b1_unans
python abstention_profile.py b1_unans

bash p1_mech.sh
python judge_tag.py --tag p1_mech --tasks locomo --workers 16
python p1_mech_analysis.py
```

The last of those regenerates the abstention operating point quoted above, reading the stored rollouts and their judge files without any further API calls.

The calibration effect on a second dataset. Everything above rests on one stratum, so we ran the same three arms on MuSiQue's unanswerable items. These are multi-hop questions whose corpus is missing the paragraph for at least one hop, and each question sees only its own 20 paragraphs. The pooled corpus will not do here. It is assembled from every item's paragraphs, so a paragraph withheld from an unanswerable item comes back through its answerable twin, and an arm that abstains is then being rewarded for ignoring evidence it can see. Per-question retrievers are the standard setting for this dataset. We judged with an unanswerable rubric written for the multi-hop case and scored on the same 0–2 scale as the LoCoMo one, so the two numbers read alike. The arm is 1,320 episodes on `gpt-4o-mini` through the model gateway, at no API cost.

The pattern repeats. OMS beats the model-managed state by 23.4 pp of correct abstention ($[+16.4, +30.2]$, $p < .001$, question-clustered bootstrap, design effect 1.42), the generic summary beats it by 21.4 pp ($[+14.8, +27.7]$), and the two orchestrator-owned arms score the same (+2.0 pp, $[-2.3, +6.4]$, $p = .356$). The rubric score gives the same three numbers to within half a point. Every harness moves on its own: IRCoT +18.2, ReAct +32.7, Iter-RetGen +17.3 and MemGPT +25.5, all at $p \leq .001$. The margins are wider than on LoCoMo, where the same contrast was 16.0

Table 10: Calibration on MuSiQue's unanswerable stratum, 440 episodes per arm over 110 questions and four harnesses. Columns as in Table 9.

arm	n	abstain	hedged	confabulated	rubric
OMS freeform	440	58.4	4.5	37.0	60.7
generic summary	440	56.4	4.3	39.3	58.6
model-managed	440	35.0	3.4	61.6	36.9

pp, and the difference is in the confabulation column. The model-managed arm answers confidently on 61.6% of questions its corpus cannot support, against 37.0% for OMS.

Reproduction:

```
bash p2_musique_unans.sh
python judge_tag.py --tag p2_mq_unans --tasks musique_unans \
    --rubric-task musique --workers 20
python p2_mq_analysis.py
```

B.7 Cross-model transfer

We repeated the nine-cell history-removed design with Llama-3.3-70B-Instruct as policy, extractor and orchestrator, and re-ran a matched `gpt-4o-mini` on the same questions in the same window as the reference. Table 11 gives the three contrasts.

Table 11: Cross-model comparison. Recovery is OMS minus history-removed over the nine cells with a removable carrier; the native comparison uses all 12 cells.

Model	**Contrast**	**Cells**	n	**OMS**	**Ref.**	**Δ (pp)**
Llama-3.3-70B	vs. history-removed	7/9 up	539	44.9	39.0	+5.8 (p=.001)
`gpt-4o-mini`	vs. history-removed	8/9 up	535	55.6	45.0	+10.6
Llama-3.3-70B	vs. native	6/12 up	539	47.9	48.1	−0.2 (p=.874)

On the larger model the recovery effect transfers in direction and shrinks by 4.8 points, while the advantage over *native* Llama harnesses is indistinguishable from zero. These two fit the framework's prediction that the value of an explicit bounded state depends on how much implicit state handling the policy already does. Neither establishes what a current frontier model would do, and §B.15 runs that arm.

B.8 Gate × state

We carried the frozen thresholds ($\tau_J = 0.6$, $\tau_U = 0.3$, $\tau_D = 1.0$, $p = 2$, mode `full`) over unchanged from the `gpt-4o-mini` sweep in §G, and applied them to Llama-3.3-70B-Instruct in a 2×2 design over gate (on/off) and UPDATE owner (OMS / none). Table 32 is the main-text summary, and Table 12 collects the three estimators of the interaction.

Table 12: Three estimators of the gate × state interaction on accuracy, all in percentage points. DiD is a question-level difference-in-differences over the questions present in all four arms.

Estimator	n	**Interaction**	**95% CI**
Pooled arm difference (+0.3 vs. −1.3)	1079	+1.7	—
Question-level DiD	180	+1.7	[−0.7, +4.1] (p=.164)
DiD, gate-active questions only	106	+2.9	—

The behavioral difference is unambiguous. With no UPDATE owner in the loop, the same thresholds fire on 13.1 pp more episodes (CI $[+8.2, +18.1]$, $p = 4.6 \times 10^{-7}$). Per domain the differences are +25.0 pp on LoCoMo ($p = 3.3 \times 10^{-8}$), +12.2 pp on MuSiQue ($p = .01$) and +2.2 pp on SWE-QA-Pro ($p = .58$). For the episodes where the two configurations disagree about firing, a McNemar test gives 131 vs. 60, $\chi^2 = 25.7$, $p = 4.1 \times 10^{-7}$.

Mechanism. Note that the third conjunct of Eq. 7 asks whether the state changed. With OMS it holds in 57.7% of $2{,}848$ rounds. If no module owns UPDATE there is no state to change, so the conjunct holds in 100.0% of $3{,}014$ rounds and the gate degenerates to a lexical loop detector. In an earlier sweep we concluded that the state-difference signal added little (§G). However, that sweep varied thresholds *within* stateful runs, where the conjunct fails in 42.3% of rounds and is only occasionally the binding one, and it never compared against a configuration with no UPDATE owner at all. Both results stand. The state term is a weak *threshold* knob and a decisive *structural* one.

Reproduction:

```
python gate_posthoc.py --tag t2_llama33 --conditions lobotomized
python gate_interaction.py
python gate_did.py
```

B.9 SAME-WINDOW ENDPOINTS

We ran the ownership controls of §6.2 together, but the endpoints they are read against (untouched native, history-removed, and the structured state) came from the April full-scale run, which does not reproduce under the current alias (Appendix B.3: $+8.6$ pp on an identical configuration). For this arm we put the three endpoints in the same window as everything else: 4 harnesses $\times$ 3 domains at $n = 100$, seed 42, `gpt-4o-mini` on the internal gateway. Iter-RetGen is memoryless and has no history to remove, so it runs `baseline` and `baseline_st` only, and we exclude it from the recovery pool.

Table 13: The three endpoints re-run in one window, paired within question. Δ_{rec} is +OMS (struct) minus history-removed, Δ_{nat} is history-removed minus native. The Iter-RetGen rows have no history-removed arm; their Δ column is `baseline_st` minus `baseline`.

Domain	Harness	n	Native	Hist.-rem.	+OMS (s)	Δ_{rec}	Δ_{nat}
LoCoMo	IRCoT	100	74.2	71.8	78.0	$+6.2$	-2.5
	ReAct	100	73.5	68.2	77.0	$+8.8$	-5.2
	Iter-RetGen	100	81.5	—	76.5	-5.0	—
	MemGPT	100	58.2	54.8	74.2	$+19.5$	-3.5
MuSiQue	IRCoT	100	26.5	18.0	34.2	$+16.2$	-8.5
	ReAct	100	41.0	17.8	38.5	$+20.8$	-23.2
	Iter-RetGen	100	42.5	—	38.8	-3.8	—
	MemGPT	100	51.0	41.5	42.5	$+1.0$	-9.5
SWE-QA-Pro	IRCoT	100	26.2	31.0	32.7	$+1.7$	$+4.8$
	ReAct	100	38.8	33.0	38.2	$+5.2$	-5.8
	Iter-RetGen	100	50.0	—	46.9	-3.1	—
	MemGPT	100	47.2	43.5	39.3	-4.2	-3.6
pooled, 9 cells			recovery $+8.4$ pp $[+5.9, +10.8]$, $p < .001$, $n = 900$				
			history removal -6.3 pp $[-8.6, -4.1]$, $p < .001$, $n = 900$				
			vs. native $+2.0$ pp $[-0.4, +4.4]$, $p = .096$, $n = 900$, -30.9% tok				

Result. The two contrasts the paper rests on both hold under the alias move. The recovery is in fact larger here ($+8.4$ pp) than in Table 2 ($+6.6$ pp for the free-form state), and on the structured state, which is the weaker of the two. In absolute terms the scores moved a long way: MuSiQue/IRCoT native is 26.5 here against 14.4 in the April run. This is why we report contrasts, and why we run every comparison inside one window. In two cells the effect goes the other way, and both are on SWE-QA-Pro, where the history-removed harness is not clearly worse than native to begin with ($+4.8$ pp for IRCoT). Iter-RetGen loses 3–5 pp from the structured state in all three domains. This is the schema artifact of §5.1. There is no history available to replace here, so a history effect cannot explain it.

Reproduction:

```
bash rebuttal/e1_endpoints.sh
bash rebuttal/judge.sh --tag e1_endpoints
EMPTY_AS_ZERO=1 python rebuttal/contrast.py --tag e1_endpoints \
```

```
  --methods ircot,react,memgpt \
  --ref lobotomized --arms lobotomized_st,baseline
```

B.10 THE FREEFORM STATE AGAINST NATIVE IN ONE WINDOW

§B.9 puts three endpoints in a single window, but its +OMS column is the structured state, so the headline contrast of §6.1, the freeform state against an intact native harness, still rested on the April full-scale run that does not reproduce (Appendix B.3). We run that contrast here and nothing else: `baseline` against `{base}_st_freeform`, 4 harnesses × 3 domains, $n = 100$ per cell at seed 42. Both §B.9 and §B.11 use the same draw, so all three arms pair question by question.

Table 14: The freeform state against an intact native harness, one window, paired within question. Δ is +OMS minus native, and the token column is the change in mean tokens per episode. * marks the cells that survive Holm–Bonferroni over the twelve-cell family at .05.

Domain	Harness	n	Native	+OMS (f)	Δ	Tokens
LoCoMo	IRCoT	100	73.2	80.8	+7.5	+86.8%
	ReAct	100	68.0	68.2	+0.2	+87.4%
	Iter-RetGen	100	81.5	75.8	−5.8	+114.3%
	MemGPT	100	61.5	71.2	+9.8	−49.8%
MuSiQue	IRCoT	100	24.8	37.0	+12.2*	+105.7%
	ReAct	100	33.8	49.8	+16.0*	+3.6%
	Iter-RetGen	100	44.8	47.8	+3.0	+127.6%
	MemGPT	100	42.5	45.2	+2.8	−66.8%
SWE-QA-Pro	IRCoT	100	24.6	34.6	+10.0*	−6.3%
	ReAct	100	43.2	40.4	−2.8	+11.4%
	Iter-RetGen	100	50.8	51.5	+0.7	+77.3%
	MemGPT	100	47.6	46.5	−1.1	−75.1%
by domain			LoCoMo +2.9 [−0.6, +6.4], MuSiQue +8.5 [+4.4, +12.6] SWE-QA-Pro +1.7 [+0.1, +3.3]			
pooled, 12 cells			+4.4 pp [+2.5, +6.3], $p < .001$, $n = 1{,}200$, −13.4% tok			

Result. The contrast replicates, and it comes out somewhat larger than the cross-window figure, +4.4 pp against +3.0 pp, while spending 13.4% fewer tokens. Holm over the twelve cells leaves three gains standing and no losses. As a cell-level result it is thinner than what Table 2 reports, although the sample here is roughly a seventh the size. The three domains separate cleanly. MuSiQue gains 8.5 pp, SWE-QA-Pro 1.7 pp, and LoCoMo 2.9 pp, which is not significant.

We read LoCoMo's null as the interesting one. Its native harnesses score 71.1 against 36.4 on MuSiQue, so a conversation that already fits in the window leaves an extractor little to recover. The two LoCoMo cells that move in opposite directions are the ones this predicts. MemGPT gains 9.8 pp and has the weakest native carrier of the four at 61.5, while Iter-RetGen loses 5.8 pp and has no history to remove, so the state is pure overhead there.

We did not expect the token column. Pooled it favors the state, but the cell-level spread runs from −75.1% to +127.6%, and the sign tracks whether the native harness accumulates raw text. MemGPT carries its own paged history and pays 52,644 tokens on SWE-QA-Pro against the state's 13,105, and it is cheaper with the state in all three domains. Iter-RetGen carries nothing between rounds, so it pays only for what the state adds, and it is dearer in all three. Such a summary is worth what the text it replaces costs, and that varies by harness.

Reproduction:

```
bash rebuttal/b2_native_same_window.sh
bash rebuttal/judge.sh --tag b2_native
EMPTY_AS_ZERO=1 python rebuttal/b2_analysis.py --tag b2_native
```

B.11 A CLOSED-BOOK CONTROL

Deleting a harness's native history only tells us something if the questions need the corpus in the first place, since anything the policy can answer from parametric memory is unaffected by any amount of history removal. SWE-QA-Pro arrives with that screen already applied by its authors, who drop questions solvable without repository interaction, and LoCoMo and MuSiQue do not, so we measured it on all three. The arm gives `gpt-4o-mini` the same seeded 100-question draw the same-window endpoints use, with no conversation, no passages and no repository, one call per question and no retrieval loop at all. We asked for a best guess and made the route to abstention a grudging one, because a control of this kind is only informative when the arm has been given every chance to succeed. The same `gpt-4o` rubric scores the answers.

Table 15: Closed-book control. Scores are the rubric score on the answerable questions of the seed-42 draw, so LoCoMo's adversarial stratum is left out; the last two columns are the weakest and the strongest arm of Table 13 on the identical questions. Abstention is the share of answers that were the literal token `UNANSWERABLE`.

Domain	n	Closed book	Abstained	Weakest arm	Strongest arm
LoCoMo	65	4.6	91%	65.0	80.8
MuSiQue	100	6.8	75%	17.8	51.0
SWE-QA-Pro	100	18.0	78%	26.2	50.0

Result. In LoCoMo and MuSiQue the policy recovers almost nothing without the corpus, scoring 4.6 and 6.8 where the weakest retrieval arm on the same questions reaches 65.0 and 17.8. Contamination therefore does not explain the history-removal contrast in either domain. SWE-QA-Pro is higher at 18.0. We read this as the generic part of a repository question — naming conventions, common library idioms, the shape an answer should take — rather than recall of the checkout, and it still falls below every arm in Table 13. Paired against the strongest arm in each domain, over the full 100-question draw rather than the answerable subset the table reports, the gaps are $+45.5$, $+44.2$ and $+32.0$ pp, all at $p < .001$. LoCoMo's is the smallest of the three relative to its table entry, since the adversarial stratum enters there and the closed-book arm does well on it by abstaining.

One figure here should temper how the unanswerable stratum gets read. In LoCoMo's category-5 questions the closed-book policy scores 94.3, since a policy with no corpus abstains and abstention is the correct answer there. This stratum rewards caution on its own. For that reason we report a false-abstention column beside every margin taken on it (§B.4), and we read the $+16.0$ pp of §B.6 as a statement about calibration that carries no claim about information gathering.

Reproduction:

```
bash b3_closed_book.sh
bash judge.sh --tag b3_closedbook \
     --tasks locomo musique swe_qa_pro
python b3_analysis.py --ref e1_endpoints
```

B.12 PUBLISHED UPDATE RULES AT MATCHED BUDGET

The reimplementations. We wrote StateAct's goal/subgoal/state triple (Rozanov & Rei, 2025), FAIR-RAG's evidence checklist (Aghajani Asl et al., 2025) and SLIM's periodic compression (Yen et al., 2026) as UPDATE prompts, each from the description in its own paper, and dropped them into the orchestrator's update step. Everything else is held: the same policy model, the same free-form state representation, the same K, the same render into the policy prompt, and one update call per round. SLIM pays one extra compression call every three rounds. That call is part of how SLIM works, so we charge its tokens to the round and they appear in the accounting; it is not an artifact of our port. We re-run the comparison arm `lobotomized_st_freeform` inside this tag instead of reading it from `t1_control`, so we run both sides of every contrast in one window under one model alias.

Scope of the ports. These are ports. Each paper's rule is embedded in a harness with its own retriever, action space and stopping rule, and running the rules under ours isolates the update rule at

the cost of discarding what the rest of each system contributes. Any loss here is evidence about the update rule, and it says nothing about the published system.

Table 16: Three published update rules against freeform OMS in the same step, paired within question. Positive Δ favors OMS.

Update rule	Domain / harness	n	rule	OMS	Δ (pp)	p
StateAct	LoCoMo / IRCoT	100	66.0	78.5	$+12.5$	$< .001$
	LoCoMo / ReAct	100	64.2	70.0	$+5.8$	.124
	MuSiQue / IRCoT	100	44.8	39.5	-5.2	.236
	MuSiQue / ReAct	100	49.2	46.8	-2.5	.466
	SWE-QA-Pro / IRCoT	100	39.5	34.0	-5.4	$< .001$
	SWE-QA-Pro / ReAct	90	44.0	43.9	-0.1	.956
	pooled	590	51.4	52.2	$+\mathbf{0.8}$ $[-1.8, +3.5]$	.528
FAIR-RAG	LoCoMo / IRCoT	100	68.5	78.5	$+10.0$	.007
	LoCoMo / ReAct	100	64.8	70.0	$+5.2$	.217
	MuSiQue / IRCoT	100	29.8	39.5	$+9.8$	.020
	MuSiQue / ReAct	100	32.5	46.8	$+14.2$	.003
	SWE-QA-Pro / IRCoT	100	32.1	34.0	$+2.0$	.232
	SWE-QA-Pro / ReAct	90	31.9	43.9	$+11.9$	$< .001$
	pooled	590	43.4	52.2	$+\mathbf{8.8}$ $[+5.9, +11.7]$	$< .001$
SLIM	LoCoMo / IRCoT	100	77.2	78.5	$+1.2$	.590
	LoCoMo / ReAct	100	76.0	70.0	-6.0	.080
	MuSiQue / IRCoT	100	27.3	39.5	$+12.2$	.004
	MuSiQue / ReAct	100	50.2	46.8	-3.5	.304
	SWE-QA-Pro / IRCoT	100	30.3	34.0	$+3.7$	.016
	SWE-QA-Pro / ReAct	87	41.9	44.5	$+2.6$	.197
	pooled	587	50.7	52.4	$+\mathbf{1.7}$ $[-0.7, +4.1]$	.173

Cost. Measured against OMS, StateAct spends 4.4% fewer tokens per episode, FAIR-RAG 3.8% more and SLIM 24.2% more. SLIM's overhead is its compression pass. In the SWE-QA-Pro/ReAct cell, where episodes are longest, it runs $31{,}249$ tokens against OMS's $22{,}207$ for no accuracy gain.

Result. Of the three rules, two are statistically indistinguishable from OMS and one is clearly worse. We read this as evidence that the extractor prompt is not carrying the paper's effects. If rules written by three other groups, installed in the same step under the same enforcement, mostly match it, then the wording of ours is not what the contrasts measure. FAIR-RAG's loss has a visible cause. Its checklist is rewritten whole each round, which is the replacement failure of §B.5 in a different guise, and its two largest losses are on MuSiQue and SWE-QA-Pro, where evidence has to accumulate across rounds.

Reproduction:

```
bash rebuttal/e5_matched.sh
bash rebuttal/judge.sh --tag e5_matched
python rebuttal/contrast.py --tag e5_matched \
  --methods ircot,react \
  --ref '{b}_st_freeform' \
  --arms '{b}_st_stateact,{b}_st_fairrag,{b}_st_slim'
```

B.13 SYNTHETIC k-HOP CHAINS

The task. We pool a corpus of records over invented entity names, so that no part of a chain can be answered from parametric knowledge. Each record names one successor and carries a short alphanumeric code, and at every hop a confusable distractor record shares the predecessor's surface form and carries a different code. Each question of depth k gives the chain's head and asks for the codes of the first k records along it, in order. The generator is `rebuttal/khop_task.py`. We use 60 questions at each of $k = 2, \ldots, 6$, and run all 300 in every cell, seed 42.

Origin of the codes. In a pilot that asked for the chain's terminal entity, the history-removed policy scored 1.000 at every depth. Following a chain is Markov: the record in hand names the next one, so a one-round window suffices, and the task measures nothing about carrying information across rounds. Asking for one fact per hop is what makes depth bite. It is also the form the counting argument behind Observation 1 is about.

Scoring. We report exact match on the whole code sequence, and per-position accuracy, which is the fraction of the k codes that are right. These are both string comparisons, so this arm consumes no judge calls and no external credit, and `gpt-4o-mini` runs on the internal gateway. Intervals are 10,000-sample bootstrap over questions within a depth, $n = 60$ per depth and per cell. Reproduction: `bash rebuttal/e4b_khop.sh` then `python rebuttal/score_khop.py -tag e4b_khop -reference lobotomized`.

Table 17: ReAct on synthetic k-hop chains. Exact match on the code sequence, with per-position accuracy in parentheses. History removal costs the whole task from $k = 3$ on. Every bounded state recovers part of it at small k and less of it as k grows, and quadrupling the item budget from $K = 3$ to $K = 12$ moves exact match by at most 5.0 pp at any depth. The last column is the ordinary-least-squares slope of the exact-match contrast against history-removed, in points per hop.

Condition	k=2	k=3	k=4	k=5	k=6	**slope**
Native	1.000 (1.00)	0.967 (.97)	0.983 (.98)	0.867 (.97)	1.000 (1.00)	+15.3
History-removed	0.817 (.91)	0.000 (.46)	0.000 (.35)	0.000 (.29)	0.000 (.22)	—
+OMS (K=6)	0.950 (.95)	0.083 (.67)	0.017 (.47)	0.050 (.38)	0.000 (.30)	−3.0
+generic summary	0.967 (.97)	0.083 (.64)	0.017 (.42)	0.000 (.32)	0.000 (.27)	−3.8
+model-managed	0.933 (.96)	0.450 (.79)	0.133 (.57)	0.017 (.47)	0.000 (.38)	−6.7
+OMS (K=3)	0.950 (.96)	0.133 (.63)	0.017 (.45)	0.017 (.40)	0.000 (.34)	−3.8
+OMS (K=12)	0.950 (.95)	0.133 (.68)	0.017 (.47)	0.000 (.36)	0.000 (.28)	−4.0
Native + OMS	1.000 (1.00)	0.950 (.97)	0.983 (.99)	0.733 (.92)	0.900 (.94)	+12.2

Capacity at depth. Re-running the free-form arm at $K = 3$ and $K = 12$ moves exact match by at most 5.0 pp at any depth, and moves the slope from -3.0 at $K = 6$ only to -3.8 and -4.0. Set against the $+15.3$ pp per hop that deleting the history costs, a factor of four in the item budget is close to irrelevant. Recovery is bounded by what the extractor writes into the state. Appendix H.2 reaches the same reading on natural corpora. This is also where §6.1's "representation does not matter" stops being benign, since a larger K does not fix a lossy U.

Which owner wins depends on the task. At $k = 3$, model-managed state recovers $+45.0$ pp ($[+31.7, +58.3]$) against $+8.3$ for append-only OMS, and it leads on per-position accuracy at every depth. Rewriting the whole state each round loses quotations on natural corpora (§6.2), but here it is the cheapest way to carry a short list of codes. We read that as a property of the task, and it does not correct RQ2. Whether the state has to accumulate evidence or only be maintained is open, and we have no evidence that either ordering is the general one.

IRCoT. Native IRCoT scores 0.000 exact at every depth (per-position .15, .06, .03, .02, .01 for $k = 2, \ldots, 6$). It issues a fixed number of retrieval rounds against the question text, and has no action vocabulary with which to name the record it just read, so it cannot walk a chain that is only discoverable one link at a time. OMS supplies exactly that. With the state carrying the last record forward as text the next retrieval query can be written against, exact match goes to 0.817, 0.717, 0.550, 0.233, 0.050 for $k = 2, \ldots, 6$, with every interval excluding zero through $k = 5$ at $n = 60$. This is the largest effect an orchestrator-owned state has anywhere in the paper, and the one case where it repairs a harness that never worked; elsewhere it recovers a deletion. The history-removed IRCoT cells recover only at $k = 2$ (0.633 free-form, 0.483 generic summary, 0.350 policy-rewritten), which is the shortest chain and the only depth at which two retrieval rounds can cover the whole answer.

B.14 A READ-ONLY TOOL INTERFACE

Every number elsewhere in the paper comes from a single `Search[query]` action over a corpus chunked in advance. An information state could then be compensating for a weak retrieval interface, with no missing history to carry. Since the theorems say nothing about how an observation arrives, only about what the policy carries between rounds, they predict that the ownership effects carry over to a richer interface. We test that here.

The arm. We keep the same 260 SWE-QA-Pro questions, the same ReAct policy and the same six conditions. In place of `Search` we give the policy `Glob`, `Grep`, `Outline` and `View` over the actual checkout (`rebuttal/tool_env.py`; nothing writes, spawns a shell, or resolves a path outside the repository root). The two interfaces are not matched on what is reachable — the Search arm sees AST chunks of `*.py`, the tool arm sees 26 whole repositories — so every comparison we draw here is between ownership regimes *within* this arm. Round cap 16, `gpt-4o-mini` on the internal gateway, judged by the same rubric as the Search arm. We leave `Search` wired so that a stray `Search[...]` does not lose an episode. We do not advertise it, and its share of actions is 0.000 in every cell. Reproduction: `bash rebuttal/e6_tools.sh`, then `judge.sh -tag e6_tools -tasks swe_tools -rubric-task swe_qa_pro`, then `rebuttal/score_tools.py -tag e6_tools`.

Table 18: The read-only tool interface, $n = 260$ questions per cell (253–260 paired). Accuracy and every descriptive column are over all 260; Δ and its interval are paired per question and drop the judge errors in each contrast, so the two columns do not sum exactly. Accuracy is the rubric judge. *Abstain* is the share of episodes answering `UNANSWERABLE`; *committed* is accuracy on the rest. *Visited* is the share of gold-cited files the episode opened, *cited* the share it named in its answer. *Redund.* is the share of actions repeating a (verb, argument) pair already issued.

Condition	Acc.	Δ vs hist.-rem.	Abstain	Comm.	Visited	Cited	Redund.	Rounds
Native	45.8	+17.7 [15.1, 20.2]	16.2%	51.1	.418	.202	.010	11.3
Native + OMS	45.6	+17.6 [15.3, 19.9]	12.3%	49.3	.389	.180	.006	9.6
History-removed	28.0	—	60.4%	43.0	.389	.021	.559	16.0
+OMS	34.1	+6.0 [3.8, 8.3]	40.8%	44.8	.393	.040	.529	15.7
+generic summary	38.3	+10.2 [7.8, 12.6]	26.5%	45.6	.399	.051	.537	15.7
+model-managed	40.8	+13.0 [10.7, 15.3]	19.2%	46.3	.387	.054	.577	15.9

The deletion effect is three times larger here. Removing the history costs 17.7 pp on the tool interface, against 6.1 pp for the same harness and questions on the Search interface (ReAct $\times$ SWE-QA-Pro, Table 2). A richer interface makes the carrier matter *more*, since a policy with four verbs over a whole repository issues more calls of more kinds and so has more to forget. Every bounded state recovers a significant part of the loss ($p < .001$ in all three cases).

The files are still opened, but no longer cited. Visited recall is flat across all six conditions (.387–.418), so every regime opens about the same share of the files the gold answer rests on. However, cited recall falls from .202 to .021, and the share of answers naming any file from 54.3% to 9.4%. The history-removed policy finds the evidence and then cannot carry it to the answer. This is what a lost information state looks like when the retrieval interface is not the bottleneck.

The ownership ordering reverses. Here the orchestrator-owned state is the *weakest* of the three. Model-managed beats OMS by +6.7 pp (CI $[+4.3, +9.2]$, $p < 10^{-6}$, $n = 258$), and generic summarization beats it by +4.4 pp (CI $[+1.9, +6.9]$, $p = .0008$, $n = 255$). The null results on the chunked corpora go the other way (Appendix B.4), and we report this as a loss. In the last two accuracy columns the mechanism is visible. The three states abstain at 40.8%, 26.5% and 19.2%, and conditional on committing to an answer they score 44.8, 45.6 and 46.3 — a 1.5 pp spread where the pooled spread is 6.7 pp. Almost the whole ordering is willingness to answer. Conditioning on committing is post-treatment and diagnostic only, but the direction is what OMS's requirement (iii) asks for. Any state that carries unresolved gaps forward makes the policy say so. Since SWE-QA-Pro contains no unanswerable questions, saying so is pure loss on this benchmark. In the unanswerable stratum this same property is an asset (Appendix B.6).

No bounded state repairs redundancy. Without the history, 55.9% of actions repeat a (verb, argument) pair the policy has already seen, and no bounded state moves that: .529, .537, .577 against .010 native. All four history-removed cells run to the round cap. Knowing a call is redundant requires holding the earlier call, and a state written to summarize *findings* does not retain the action trace. Thus there is a gap between what Observation 2 says a sufficient state must support and what our extractor writes. We think closing it is the clearest direction for the next version of U.

The one place OMS pays here. Added on top of the unmodified harness it is accuracy-neutral (-0.2 pp, CI $[-2.5, +2.2]$, $p = .891$, $n = 260$), while cutting 5.8% of tokens, 1.6 rounds and 1.8 actions per question, and raising cited precision from .690 to .756. We saw the same cost-ceiling result in §6.3, here on an interface where the raw observations are whole source files.

B.15 Two frontier policies

Every contrast in the body runs on `gpt-4o-mini` or `Llama-3.3-70B`, so a bounded state might be a crutch for a weak policy. If a stronger model could read its own raw trajectory well enough that owning the update makes no difference, then the finding is a transient property of small models. §3.1 predicts otherwise, since the carrier is what aliases the model that reads it.

The arm. We keep the same six conditions and the same ReAct policy, and swap the policy for a frontier model. We run `gpt-5.6-terra` on the internal gateway across all three domains, and `gemini-2.5-pro` through OpenRouter, which the gateway does not serve, on all three as well. We run $n = 100$ questions per cell, judged by the same `gpt-4o` rubric as everything else. Reproduction: `bash rebuttal/t6_frontier.sh`, then `judge.sh -tag t6_frontier -tasks locomo,musique,swe_qa_pro`. An earlier frontier pilot (`t5_frontier`) is on disk and we do *not* analyze it, because an action-parser bug ended roughly half of its episodes in `Finish[I don't know]`. Every cell reported here has zero parse failures. Some `gemini-2.5-pro` episodes returned an empty answer and the judge drops them, so the MuSiQue cells are $n = 99$ and the SWE-QA-Pro cells run from $n = 97$ up to the full $n = 100$.

Table 19: `gpt-5.6-terra`, ReAct, $n = 100$ per cell. Accuracy is the rubric judge. Δ columns are paired per question with a bootstrap interval; tokens are per question, relative to the row's own reference. Table 20 is the same design under `gemini-2.5-pro`.

Condition	Acc.	Δ vs hist.-rem.	Δ vs native	Tok.	Rounds
LoCoMo					
Native	63.7	-0.5 $[-11.2, +10.2]$	—	2748	3.7
Native + OMS	74.0	$+9.8$ $[+1.4, +18.1]$	$+10.2$ $[+2.2, +18.3]$	2252	2.8
History-removed	64.2	—	$+0.5$	2887	5.8
+OMS	74.2	$+10.0$ $[+1.6, +18.4]$	$+10.5$ $[+1.3, +19.7]$	2894	3.5
+generic summary	71.0	$+6.8$ $[-1.8, +15.3]$	$+7.2$ $[-0.7, +15.2]$	3451	3.8
+model-managed	76.0	$+11.8$ $[+3.0, +20.5]$	$+12.2$ $[+2.6, +21.9]$	2354	3.3
MuSiQue					
Native	61.0	$+22.0$ $[+12.0, +32.0]$	—	5112	3.8
Native + OMS	68.8	$+29.8$ $[+20.0, +39.5]$	$+7.8$ $[+0.5, +15.0]$	6033	3.4
History-removed	39.0	—	-22.0	4707	5.7
+OMS	68.5	$+29.5$ $[+19.9, +39.1]$	$+7.5$ $[+0.7, +14.3]$	7789	4.9
+generic summary	67.0	$+28.0$ $[+18.3, +37.7]$	$+6.0$ $[-2.1, +14.1]$	9506	5.0
+model-managed	64.8	$+25.8$ $[+17.1, +34.4]$	$+3.8$ $[-5.1, +12.6]$	6572	4.4
SWE-QA-Pro					
Native	63.6	$+25.5$ $[+20.2, +30.8]$	—	24070	5.2
Native + OMS	62.3	$+24.2$ $[+18.7, +29.7]$	-1.3 $[-4.4, +1.8]$	21324	4.6
History-removed	38.1	—	-25.5	14593	6.7
+OMS	60.4	$+22.2$ $[+17.1, +27.4]$	-3.3 $[-6.4, -0.1]$	24679	6.1
+generic summary	57.8	$+19.6$ $[+14.4, +24.9]$	-5.9 $[-8.7, -3.0]$	27063	6.3
+model-managed	64.6	$+26.5$ $[+21.5, +31.5]$	$+1.0$ $[-2.3, +4.3]$	21922	5.5

Deleting the history costs a frontier policy more. Changing the policy more than doubles native MuSiQue accuracy, 29.6 to 61.0 (ReAct, Table 2), so the arm has ample room to show a null. It does

Table 20: `gemini-2.5-pro`, ReAct, $n = 100$ per cell, less the episodes that returned an empty answer and that the judge drops ($n = 99$ on two MuSiQue cells, $n = 97$ to 100 on SWE-QA-Pro). Same design and same judge as Table 19.

Condition	Acc.	Δ vs hist.-rem.	Δ vs native	Tok.	Rounds
LoCoMo					
Native	65.8	−0.2 [−8.6, +8.1]	—	8298	4.7
Native + OMS	63.7	−2.2 [−9.4, +4.9]	−2.0 [−9.6, +5.6]	8112	3.5
History-removed	66.0	—	+0.2	4841	4.4
+OMS	65.2	−0.8 [−8.6, +7.1]	−0.5 [−8.6, +7.6]	10336	4.0
+generic summary	68.5	+2.5 [−6.0, +11.0]	+2.8 [−5.5, +11.0]	11064	4.4
+model-managed	56.5	−9.5 [−18.2, −0.8]	−9.2 [−17.9, −0.6]	7000	3.6
MuSiQue					
Native	46.7	+15.4 [+5.2, +25.6]	—	22207	6.0
Native + OMS	47.8	+16.8 [+7.1, +26.4]	+1.5 [−7.3, +10.3]	31780	5.8
History-removed	31.0	—	−15.4	18961	5.9
+OMS	49.2	+17.9 [+8.5, +27.3]	+3.6 [−4.9, +12.0]	24016	5.4
+generic summary	49.2	+18.2 [+8.7, +27.8]	+3.0 [−6.1, +12.1]	28076	5.6
+model-managed	49.0	+18.0 [+8.9, +27.1]	+2.0 [−6.6, +10.6]	17654	5.0
SWE-QA-Pro					
Native	65.3	+21.6 [+16.9, +26.3]	—	52300	6.9
Native + OMS	59.9	+16.0 [+11.1, +20.9]	−5.4 [−8.8, −2.1]	48059	7.0
History-removed	44.0	—	−21.6	24413	6.9
+OMS	55.0	+11.2 [+6.5, +15.8]	−10.1 [−13.5, −6.6]	36017	6.5
+generic summary	50.5	+6.4 [+1.4, +11.3]	−14.8 [−19.0, −10.6]	39458	6.7
+model-managed	57.9	+14.0 [+9.6, +18.4]	−7.4 [−10.4, −4.4]	28605	6.1

not. With the trajectory removed the cost is 22.0 pp [12.0, 32.0] on MuSiQue, against 17.4 pp for `gpt-4o-mini` under the same harness, and 25.5 pp [20.2, 30.8] on SWE-QA-Pro, against 6.1 pp. This is four times the deficit, and the largest single effect in the arm. `gemini-2.5-pro` agrees, losing 15.4 pp [5.2, 25.6] on MuSiQue ($p = .004$) from a native 46.7. Being a better reader of a raw trajectory does not make the trajectory a sufficient carrier. If the deleted information is a chain of entity bridges or a walk through a repository, the stronger policy has more of it to lose.

On LoCoMo the deletion is free. Removing the history changes nothing for this policy (−0.5 pp, CI [−11.2, +10.2], $p = .927$), where it costs `gpt-4o-mini` 5.7 pp. On native LoCoMo the frontier policy also gains almost nothing over `gpt-4o-mini` (63.7 against 65.1), so the errors left on this benchmark are not the ones a stronger reader fixes. The two policies close in about the same number of rounds (3.7 against 3.1), and a policy that retrieves the right passage in the first round or two never needs the trajectory it was denied. Thus the deletion effect the body reports replicates on MuSiQue and SWE-QA-Pro, at equal or larger magnitude, and disappears on LoCoMo. `gemini-2.5-pro` gives the same LoCoMo null (+0.2 pp, CI [−8.1, +8.6], $p = .953$), so the exception belongs to the benchmark rather than to any one policy.

Gains over an unmodified harness are policy-specific. Added on top of the native harness, OMS gains `gpt-5.6-terra` +10.2 pp [2.2, 18.3] on LoCoMo at 18% fewer tokens, and +7.8 pp [0.5, 15.0] on MuSiQue. It gains nothing on SWE-QA-Pro (−1.3 pp [−4.4, +1.8], $p = .406$) at 11% fewer tokens and 4.6 rounds against 5.2. For `gemini-2.5-pro` the LoCoMo gain does not reproduce (−2.0 pp [−9.6, +5.6], $p = .602$), and SWE-QA-Pro turns into a significant loss (−5.4 pp [−8.8, −2.1], $p = .002$). We do not claim that a bounded state improves on a native harness at frontier scale. Our evidence is one policy on two domains, and the second policy contradicts it on the domain where the first gained most. At `gpt-4o-mini` scale the same addition gains 2.0 pp ([−0.5, +4.6], $p = .122$) at 30.9% fewer tokens, which is also not a significant accuracy gain (§B.9). Both frontier policies agree on the contrast the theorems speak to, recovery from a deleted history, and it is significant on every task where the deletion costs anything at all. Our reading is that a stronger policy is better at *using* a well-formed state and no better at manufacturing one from a raw trajectory. Where the state binds, the gap between what the policy could do and what the harness lets it do therefore widens. On SWE-QA-Pro it does not bind. Recovery falls 3.3 pp [−6.4, −0.1] short of native under a stronger policy, which is the extractor loss the limitations already disclose.

The split does not follow ownership. All three score about the same under `gpt-5.6-terra`. Model-managed is highest on LoCoMo (76.0 against 74.2 for OMS) and on SWE-QA-Pro (64.6 against 60.4) and lowest on MuSiQue (64.8 against 68.5). None of these differences is significant at $n = 100$. Under `gemini-2.5-pro` MuSiQue behaves the same way, and the three are within 0.2 pp of one another. LoCoMo does not: model-managed scores 56.5 against 65.2 for OMS, which is -9.5 pp $[-18.2, -0.8]$ against the history-removed harness and -9.2 pp $[-17.9, -0.6]$ against native. This is the only significant loss a policy-owned state takes anywhere in the paper, and it falls the way the ownership argument would predict. However, it is one task at $n = 100$, and the same comparison under `gpt-5.6-terra` runs $+1.8$ pp the other way, so we rest nothing on it. SWE-QA-Pro separates too, along a different axis. Generic summarization falls behind both of the others, by 4.9 pp against OMS (CI $[+1.2, +8.7]$, $p = .010$) and 7.8 pp against model-managed ($[+4.0, +11.6]$, $p < .001$), while the ownership contrast on that task stays null ($+2.6$ pp, CI $[-0.7, +5.9]$, $p = .121$). All three are paired within question, so they will not reproduce by subtracting the column means, which are taken over every episode the judge scored. In the repository domain what gets punished is a representation that compresses without being told what to keep. That result is about the state format, and it says nothing about who writes it. §6.2 reports the same null at $n \geq 500$ per cell on `gpt-4o-mini`, $+0.9$ pp for model-managed and $+1.0$ pp for generic summarization, neither of them significant. Accuracy was never where the body located the cost of giving the update away, so the frontier arm agreeing with it is a replication. Accuracy ties here and price does not. In `gemini-2.5-pro`'s MuSiQue cells the generic summary spends 28.1k tokens per question against 24.0k for OMS and 17.7k for model-managed, for the same score to within 0.2 pp. Across both policies the coarser statement is what replicates: every bounded state beats the history-removed harness wherever deleting the history costs anything at all.

The faithfulness cost replicates. The price of giving the update away is paid on faithfulness. Notes that the agent marks as verbatim quotations are absent from the retrieved corpus 34.1% of the time under model-managed state, against 8.9% under OMS and 5.5% under generic summarization (§6.2, Appendix B.5). Scoring the frontier states with the same check gives 37.8%, 12.9% and 18.6%. This is a factor of three between an owned state and a model-written one, against nearly four in the body, and model-managed is again the worst of the three on each task taken separately (0.378 LoCoMo, 0.315 MuSiQue, 0.406 SWE-QA-Pro). Two things do not carry over. Generic summarization moves from best to middle, which is what we would expect of a representation whose grounding is asked for but never enforced. Model-managed also no longer writes about a third as many quotations as the others (328 against 264 for OMS and 376 for summarization), so at this policy strength the unverified rate is not an artifact of a small denominator. Written to `rebuttal/state_quality_t6.json`.

The judge holds on frontier outputs. Re-running the gold-agreement check of Appendix F.3 against this arm's LoCoMo and MuSiQue rollouts gives AUROC 0.883 on both tasks' short-gold strata ($\kappa_{0.75} = 0.72$ LoCoMo, 0.65 MuSiQue), against 0.895 and 0.945 on the `gpt-4o-mini` rollouts it was validated on. This check needs a short string gold, so it is unavailable on SWE-QA-Pro, as it is in the body. The drop is one-sided. Its share of cover-EM-wrong answers credited rises from 0.17 to 0.31 on MuSiQue and 0.29 to 0.31 on LoCoMo, while the share of cover-EM-right answers it penalizes stays at 0.03 and 0.00. Frontier answers are longer and more hedged, and the deterministic gold label stops matching them. Since that error is shared across conditions within a task, it does not move the paired contrasts. Written to `rebuttal/judge_validation_t6.json`.

C Framework Scope

Table 21 lists the applications whose information-gathering phase is a CGDP. The harness interface map is Table 1 in §4.

D Experimental Setup and Reproducibility

Datasets and splits. We use three datasets: LoCoMo (1,275 conversational-memory tasks), MuSiQue (500 answerable multi-hop Wikipedia tasks) and SWE-QA-Pro (260 repository-grounded questions over 26 long-tail GitHub checkouts, 10 per repository). We take the standard test

Table 21: Information-gathering phases of diverse agentic applications. Each is a CGDP when the source is static or approximately static over an episode. Agents whose action spaces also modify the environment (rows 1 and 4) are covered only during their read-only phase. General side-effecting action requires a state-transition kernel (§3.1).

Application	**Latent corpus c**	**Query actions / observation kernel**
Code assistants	Repository snapshot	Read-only file and shell queries
Agentic memory	Conversational history	Vector-database retrieval
Support agents / RAG	Knowledge base	SQL / BM25 / embedding queries
Recursive LMs	Python data objects	Read-only REPL inspection
Deep research	Web snapshot	Search-engine APIs

and validation splits that each benchmark's authors provide. SWE-QA-Pro is the `test` split of `TIGER-Lab/SWE-QA-Pro-Bench` (Cai et al., 2026), whose construction pipeline already drops questions answerable without repository interaction, so that benchmark comes with a closed-book screen applied by its authors. LoCoMo and MuSiQue carry no such screen, so we ran one ourselves and report it in §B.11.

Compute resources and LLM usage. We ran all agent trajectories, modular extractions and rubric evaluations through hosted APIs reached over an internal gateway, using the `gpt-4o-mini` and `gpt-4o` endpoints for the main grid. The whole set of experiments — baseline runs, history-removal sweeps, gate ablations and the frontier arms of §6.3 — cost roughly $\$1{,}500$ in API credits.

Licenses. We use LoCoMo (**CC BY-NC 4.0**), MuSiQue (**CC BY 4.0**), and SWE-QA-Pro Bench (**MIT**) as evaluation datasets, and `all-MiniLM-L6-v2` (**Apache-2.0**) for dense retrieval embeddings. We accessed `gpt-4o-mini` and `gpt-4o` over an internal gateway to the OpenAI API, under the applicable OpenAI Services Agreement and Service Terms, and the frontier models of §6.3 through their respective hosted endpoints under the same kind of terms.

E Orchestrator and Judge Prompts

We list the key prompts designed for our interventions. All prompts are in the experiment codebase under `src/orchestrator/`. Harness-specific prompts and rubric judge prompts are in `src/methods/prompts/` and `src/scoring/`.

E.1 Structured Extraction Prompt

The structured prompt separates output into facts, resolved questions, and new questions. The instruction to "state precisely what evidence is still missing" produces the "no evidence" artifacts analyzed in §6.

```
New retrieved passages:
{observation}

What we already know (DO NOT repeat any of these):
{established_facts}

<scratchpad>
First, what do the passages actually state?  Extract the key
claims exactly as the evidence presents them -- preserve who
did what, which entity is involved, what values are
mentioned.  Do not paraphrase in a way that changes
attribution or meaning.
Then, given the question "{question}", what is new here
compared to what we already know?  Do any of these claims
resolve our open questions?  If a fact is already listed
above, skip it entirely.
</scratchpad>

Open questions we are still investigating:
{open_questions}
```

```
Output ONLY genuinely new facts not already listed above.
If the passages contain nothing new beyond what we already
know, write "Nothing relevant."

New facts:
- The claim exactly as the evidence states it (source:
document or passage identifier)

Resolved questions:
- Which open questions are now answered by the evidence?

New questions:
- What specific evidence is still missing?  State precisely
what has NOT been found.
```

E.2 FREEFORM EXTRACTION PROMPT

The freeform prompt produces notes and memories without fact/question separation.

```
New retrieved passages:
{observation}

Your current notes (DO NOT repeat any of these):
{existing_notes}

<scratchpad>
First, what do the passages actually state?  Extract claims
exactly as presented -- preserve who did what, which
entities are involved.  Do not paraphrase in a way that
changes attribution.
Then, how do these relate to the question "{question}" and
your current notes?  What is genuinely new?  If a note
already covers this information, skip it entirely.
</scratchpad>

Write ONLY genuinely new notes not already covered above.
If the passages contain nothing new beyond your existing
notes, write "Nothing relevant."

Notes:
- A finding exactly as the evidence presents it

Memories to keep:
- Verbatim quote or key passage worth preserving
```

Both prompts see only $z_t + o_t$, roughly 500 tokens of context, and never the full history.

E.3 REORGANIZATION PROMPT

To strictly bound the size of the information state, the orchestrator allows the state to grow up to $K_{\text{trigger}} = 10$ items before pausing to execute a reorganization call. This call curates the state back down to a target size of $K_{\text{target}} = 6$ (the capacity limit reported in the main text).

```
You are curating an investigation state.  Given the original
question and all facts and questions gathered so far,
produce a compact, prioritized state.

Original question:  {question}

Current facts:  {facts}

Open questions:  {questions}

Instructions:
- Keep at most {k_target} facts and {n_questions} open
questions
- Merge redundant facts into single comprehensive claims
- Drop facts irrelevant to the question
- PRESERVE facts that form multi-hop reasoning chains, even
if individually they seem tangential
- Each fact must retain its source attribution
```

```
- Rewrite open questions to reflect what is actually still
unknown
- Remove questions that have been answered by the facts
- Order facts by importance (most important first)
```

E.4 LLM-BASED STAGNATION GATE PROMPTS

We evaluated two LLM-based stagnation detectors against the programmatic gate. Both see the current investigation state and the recent retrieval rounds.

Conservative (v3). Defaults to CONTINUE, and asks for concrete evidence of stagnation before it will recommend stopping.

```
You are deciding whether further retrieval will meaningfully
improve the answer to this question.

QUESTION: {question}

CURRENT INVESTIGATION STATE: {current_state}

RECENT RETRIEVAL (last {window} rounds):  {recent_rounds}

Your DEFAULT is to CONTINUE retrieval.  Only recommend
stopping if you can point to CONCRETE evidence of
stagnation:
- The same passages or near-paraphrases are appearing across
multiple rounds
- Search queries have covered the obvious angles and a
meaningfully different direction is hard to identify
- The current answer already addresses the question and
further evidence is unlikely to change it

Do NOT recommend stopping just because:
- The answer seems plausible (it may still be incomplete)
- A few passages overlap (some overlap is normal)
- You are uncertain about the answer quality (uncertainty
means more retrieval could help)

VERDICT: PRODUCTIVE / QUERY_STALE / EXHAUSTED
REASON: [explanation]
```

Neutral (v3_neutral). No default direction, and the three verdicts are presented symmetrically.

```
You are evaluating whether an information retrieval
investigation is making progress or has stagnated.

QUESTION: {question}

CURRENT INVESTIGATION STATE: {current_state}

RECENT RETRIEVAL (last {window} rounds):  {recent_rounds}

Based on the evidence above, choose one of the following
verdicts:

VERDICT: PRODUCTIVE -- if new, relevant information is still
being discovered each round.
VERDICT: QUERY_STALE -- if the current search direction is
exhausted but a specific untried angle could yield new
information.
VERDICT: EXHAUSTED -- if retrieval has stalled and further
rounds are unlikely to surface new relevant content.

VERDICT:
REASON:
```

F FULL RESULTS AND ADDITIONAL METRICS

Every table in this appendix labels the four conditions the same way. **B** is the native harness as published, and **L** is the same harness with its cross-round carrier deleted, which the body calls

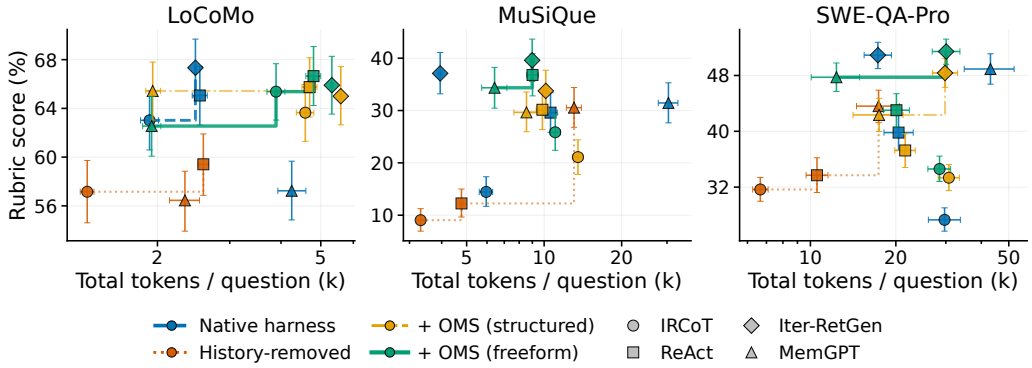


Figure 3: Accuracy against mean total tokens per question (policy + orchestrator + tool calls; log scale), with 95% question-level bootstrap CIs on both axes. Staircases are the accuracy-per-token frontier of each condition class. The OMS arms are history-removed + OMS for IRCoT, ReAct and MemGPT, and native + OMS for Iter-RetGen, which has no removable history. Ownership raises accuracy in $10/12$ cells, but the token change has no fixed sign: strongly negative where the native harness carries the most history (MemGPT: -2.3k on LoCoMo, -24.0k on MuSiQue, -30.7k on SWE-QA-Pro) and positive in the other six, where the native prompt was already small ($+2.0$k to $+12.8$k).

history-removed. z^{s} and z^{f} add the orchestrator-owned state on top of $\mathbf{L}$, filling a fixed predicate schema in the first case and writing prose in the second, and the body calls these *struct* and *free*. A column headed B$-$L reports the first condition minus the second in accuracy points, so a positive entry means the left condition scores higher.

F.1 Rubric Judge Scores and Statistical Tests

Tables 22 and 23 give the per-cell paired t-tests and the split-condition scores behind the pooled numbers of §6.

Table 22: Full paired t-tests for persistent information state (z_t). Each cell reports the left condition minus the right condition in pp; positive means the left condition scores higher. Iter-RetGen has no history-removed condition. **Bold blue** = $p < .05$.

		Hist.-rem.	Native − OMS		z_t recovery		Format
Task	**Harness**	**B−L**	**B−z^{s}**	**B−z^{f}**	**L−z^{s}**	**L−z^{f}**	**z^{s}−z^{f}**
LoCoMo	IRCoT	**+5.9**	−0.6	**−2.4**	**−6.5**	**−8.2**	−1.7
	ReAct	**+5.6**	−0.7	−1.6	**−6.3**	**−7.2**	−0.9
	Iter-RetGen	—	**+2.3**	+1.4	—	—	−0.9
	MemGPT	+0.8	**−8.2**	**−5.3**	**−9.0**	**−6.1**	**+2.9**
MuSiQue	IRCoT	**+5.4**	**−6.7**	**−11.4**	**−12.0**	**−16.8**	**−4.8**
	ReAct	**+17.3**	−0.6	**−7.2**	**−17.9**	**−24.6**	**−6.7**
	Iter-RetGen	—	+3.3	−2.5	—	—	**−5.8**
	MemGPT	+0.9	+1.8	−2.9	+0.9	**−3.8**	**−4.7**
SWE-QA	IRCoT	**−4.4**	**−6.0**	**−7.3**	−1.7	**−2.9**	−1.3
	ReAct	**+6.1**	+2.6	**−3.2**	**−3.6**	**−9.3**	**−5.8**
	Iter-RetGen	—	**+2.6**	−0.5	—	—	**−3.1**
	MemGPT	**+5.3**	**+6.6**	+1.2	+1.3	**−4.1**	**−5.4**
Sig. pos / neg		7 / 1	3 / 3	0 / 6	0 / 6	0 / 9	1 / 7

F.2 Lexical and Embedding-based Metrics

String metrics (Token F1, Exact Match, ROUGE-1, METEOR, SentenceBERT cosine similarity) broadly agree with the rubric judge rankings, so the gains are not an artifact of the judge preferring

Table 23: LoCoMo z_t effect (pp) split by answerable and unanswerable (LoCoMo's category-5 adversarial questions), pooling 835 answerable and 440 unanswerable episodes across the four harnesses. Each cell reports the z_t condition minus the comparison condition, so positive means z_t scores higher. On the answerable stratum the freeform state is ahead. On the unanswerable stratum the ordering reverses: the structured state gains in three of the four harnesses and the freeform state loses ground in three of the four. Both arms here are orchestrator-owned, so this contrast is about the shape of the state; the $+16.0$ pp of §B.6 compares against a policy-owned arm instead. Iter-RetGen has no history-removed condition.

	Answerable (835q)				Unanswerable (440q)			
Harness	$\mathbf{z^s-B}$	$\mathbf{z^f-B}$	$\mathbf{z^s-L}$	$\mathbf{z^f-L}$	$\mathbf{z^s-B}$	$\mathbf{z^f-B}$	$\mathbf{z^s-L}$	$\mathbf{z^f-L}$
IRCoT	+1.0	+5.8	+12.1	+16.9	−0.1	−4.0	−4.2	−8.1
ReAct	−1.4	+5.8	−4.0	+3.2	+4.6	−6.4	+25.8	+14.8
Iter-RetGen	−8.0	+0.1	—	—	+8.3	−4.4	—	—
MemGPT	−5.0	−1.6	−5.3	−1.9	+33.2	+18.4	+36.1	+21.3

z_t-conditioned answers. On MuSiQue, z^f is the best condition for every method on every string metric.

The few divergences between string metrics and the rubric judge are why we make the rubric judge primary (Table 22). A perfectly correct abstention (e.g. answering "UNANSWERABLE") has zero token overlap with the gold answer, which deflates F1 on LoCoMo even though abstaining is the optimal behavior. SWE-QA-Pro string metrics likewise favor history-removed models, because longer and more verbose code explanations happen to overlap more with the ground truth. The rubric judge captures the task-specific quality dimensions that surface overlap cannot, namely abstention, correctness and entity attribution.

Table 24: Token F1 and Exact Match: z_t effect (pp). Iter-RetGen (memoryless by design) has no history-removed condition. **Bold blue** = $p < .05$.

		Token F1				Exact Match			
Task	**Harness**	$\mathbf{B\to z^s}$	$\mathbf{B\to z^f}$	$\mathbf{L\to z^s}$	$\mathbf{L\to z^f}$	$\mathbf{B\to z^s}$	$\mathbf{B\to z^f}$	$\mathbf{L\to z^s}$	$\mathbf{L\to z^f}$
LoCoMo	IRCoT	−0.1	**+2.2**	**+5.9**	**+8.1**	+0.4	**+1.4**	**+1.1**	**+2.1**
	ReAct	−0.6	**+3.2**	**−5.5**	**−1.7**	+0.1	**+1.3**	**−2.8**	**−1.6**
	Iter-RetGen	**−7.9**	+0.6	—	—	**−3.6**	+0.7	—	—
	MemGPT	**−4.1**	**−1.5**	**−4.2**	**−1.6**	−0.1	−0.1	0.0	0.0
MuSiQue	IRCoT	+1.7	**+5.4**	**+6.2**	**+9.9**	+1.4	**+3.4**	**+5.0**	**+7.0**
	ReAct	+2.7	**+7.7**	**+9.9**	**+14.9**	+1.2	**+4.2**	**+5.4**	**+8.4**
	Iter-RetGen	**−5.1**	+2.0	—	—	**−3.0**	**+2.8**	—	—
	MemGPT	**+1.3**	**+3.7**	+0.3	**+2.7**	0.0	+0.2	0.0	+0.2
SWE	IRCoT	**+3.5**	**+3.6**	**+1.6**	**+1.7**	0.0	0.0	0.0	0.0
	ReAct	−0.7	**+2.5**	+0.3	**+4.7**	0.0	0.0	0.0	0.0

Table 25: ROUGE-1 and METEOR: z_t effect (pp). Iter-RetGen has no history-removed condition. **Bold blue** = $p < .05$.

		ROUGE-1				METEOR			
Task	**Harness**	$\mathbf{B\to z^s}$	$\mathbf{B\to z^f}$	$\mathbf{L\to z^s}$	$\mathbf{L\to z^f}$	$\mathbf{B\to z^s}$	$\mathbf{B\to z^f}$	$\mathbf{L\to z^s}$	$\mathbf{L\to z^f}$
LoCoMo	IRCoT	−0.7	+1.9	+7.2	+9.8	−1.1	+1.4	+8.7	+11.1
	ReAct	−0.6	+3.2	−5.7	−0.7	−1.9	+2.6	−4.8	−1.4
	Iter-RetGen	−8.3	+0.5	—	—	−6.9	+0.6	—	—
	MemGPT	−4.1	−1.5	−4.1	−1.6	−5.6	−2.6	−6.0	−3.0
MuSiQue	IRCoT	+1.6	+5.4	+6.7	+10.5	+2.6	+5.8	+6.0	+9.3
	ReAct	+2.9	+7.4	+9.7	+14.2	+2.0	+9.8	+6.1	+13.9
	Iter-RetGen	−5.1	+2.0	—	—	−2.4	+2.5	—	—
	MemGPT	+1.3	+3.6	+0.4	+0.9	+2.6	+1.0	+2.6	+2.7
SWE	IRCoT	+4.1	+4.3	+3.2	+3.3	+1.8	+0.2	+0.5	+0.4
	ReAct	−1.4	+1.4	+1.4	+5.2	−0.4	+2.0	+0.7	+2.1

Table 26: SBERT and Rubric Judge: z_t effect (pp). Iter-RetGen has no history-removed condition. Rubric judge significance is from Table 22; SBERT significance is not shown.

		SBERT				Rubric Judge			
Task	**Harness**	**B**$\rightarrow z^{\mathrm{s}}$	**B**$\rightarrow z^{\mathrm{f}}$	**L**$\rightarrow z^{\mathrm{s}}$	**L**$\rightarrow z^{\mathrm{f}}$	**B**$\rightarrow z^{\mathrm{s}}$	**B**$\rightarrow z^{\mathrm{f}}$	**L**$\rightarrow z^{\mathrm{s}}$	**L**$\rightarrow z^{\mathrm{f}}$
LoCoMo	IRCoT	−0.8	+1.5	+4.7	+7.0	+0.6	+2.4	+6.4	+8.2
	ReAct	−0.4	+3.4	−5.7	−1.9	+0.6	+1.5	+6.3	+7.2
	Iter-RetGen	−7.8	−0.1	—	—	−2.3	−1.4	—	—
	MemGPT	−5.3	−3.0	−5.4	−3.1	+8.2	+5.3	+8.9	+6.0
MuSiQue	IRCoT	+2.4	+6.7	+5.7	+10.0	+6.7	+11.5	+12.1	+16.9
	ReAct	+2.7	+8.1	+9.2	+14.6	+0.5	+7.2	+17.9	+24.6
	Iter-RetGen	−6.3	+2.1	—	—	−3.3	−2.5	—	—
	MemGPT	+1.7	+3.8	+0.8	+2.9	−1.8	+3.0	−0.9	+3.9
SWE	IRCoT	+4.4	+1.9	+13.5	+2.7	+6.0	+7.3	+1.6	+2.9
	ReAct	+3.5	+5.2	−2.9	+7.1	−2.5	+3.2	+3.6	+9.3

Table 27: Stagnation gate effect on string metrics (pp): the blended answer, which is the gate's answer where the gate fired and the harness's own answer otherwise, minus the harness alone. Positive = gate improves the metric. **Bold blue** = $p < .05$. IRCoT benefits most on LoCoMo/MuSiQue; SWE shows negative F1 diffs because early stopping truncates verbose code answers that have high token overlap.

		Token F1				ROUGE-1			
Task	**Harness**	**B**	**L**	z^{s}	z^{f}	**B**	**L**	z^{s}	z^{f}
LoCoMo	IRCoT	+0.1	+2.8	+0.7	+0.2	+0.8	+5.6	+1.7	+0.7
	ReAct	+0.1	−1.5	+0.7	+0.3	+0.1	−1.5	+0.7	+0.3
	Iter-RetGen	−5.5	—	0.0	0.0	−5.7	—	0.0	0.0
	MemGPT	0.0	−0.1	−0.1	−0.1	0.0	−0.1	0.0	0.0
MuSiQue	IRCoT	+5.3	+2.1	+1.8	+0.9	+6.1	−0.3	0.0	+0.4
	ReAct	+0.8	−0.3	−0.3	+0.1	+1.4	+2.3	+3.2	+0.8
	Iter-RetGen	−1.9	—	0.0	−1.9	−0.3	—	−0.3	+0.4
	MemGPT	+2.7	−2.4	+1.0	+0.1	+2.7	−2.2	+1.2	+0.2
SWE	IRCoT	−5.4	−8.4	−5.3	−5.2	−8.2	−9.2	−6.5	−6.6
	ReAct	+1.1	−2.2	−0.1	−1.2	+0.8	−1.8	−0.2	−1.0

F.3 Validating the rubric judge against the datasets' own labels

Accuracy in this paper is a `gpt-4o` rubric score, so it is fair to ask how much of the result is the judge. We cannot answer that with human annotation, and we do not have to. Every rollout we stored carries the dataset's own gold answer and its answerability flag. That gives us a labeled set of 3,600 items over the 36 `t1_control` cells at no annotation cost, and `rebuttal/judge_validation.py` scores the judge against it.

The gold answer supports a deterministic label only where it is short. On MuSiQue the gold is a span of 2 words at the median and 9 at the longest, so cover-exact-match (the normalized gold string appears somewhere in the normalized prediction) is a reliable label. On LoCoMo the gold is a short free-text sentence, 4 words at the median, where cover-EM starts to miss correct paraphrases. We report the subset whose golds run to at most 6 words separately. On SWE-QA-Pro the gold is a reference passage of roughly 200 words, and cover-EM against it would measure verbosity. We build no label there, and the judge stays unvalidated on that task.

Table 28 gives the agreement. On MuSiQue, where the label is trustworthy, the judge separates correct from incorrect answers at $\mathrm{AUROC} = 0.947$ and agrees with the label at $\kappa = 0.744$. Against token F1 in place of cover-EM the rank correlation is 0.851. On LoCoMo the agreement is lower, $\mathrm{AUROC} = 0.826$ over all answerable questions. But restricting to the golds short enough for cover-EM to be fair recovers most of the difference (0.895, $\kappa = 0.750$), which points at the label and away from the judge.

On the unanswerable stratum the rubric is never shown a gold answer, and the dataset's own flag fixes the correct behavior. There the judge and a regular-expression abstention detector agree on all 420

Table 28: Judge agreement with a deterministic label derived from the gold answers the datasets already carry, over the 36 `t1_control` cells. The label is cover-EM for the answerable strata and abstention for the unanswerable stratum, where the rubric is not shown the gold answer at all. κ thresholds the judge's normalized score at 0.75. CIs are 2,000 bootstrap resamples.

Stratum	n	**Base**	ρ	**AUROC [95% CI]**	κ
LoCoMo, answerable	780	0.514	0.636	0.826 [0.801, 0.852]	0.540
golds ≤ 6 words	576	0.642	0.753	0.895 [0.864, 0.922]	0.750
MuSiQue, answerable	1,200	0.336	0.803	0.947 [0.935, 0.957]	0.744
LoCoMo, unanswerable	420	0.660	0.987	1.000 [1.000, 1.000]	1.000
SWE-QA-Pro	1,184	—	no defensible deterministic label		

items. Thus the stratum carrying our largest reported effect can be rescored without a model at all, and none of it rests on judge preference.

Which way the judge errs. The judge almost never penalizes an answer the gold label calls correct: 0 of 401 such answers on LoCoMo and 9 of 403 on MuSiQue score below 0.25. It errs in the other direction, giving at least 0.75 to 22.4% of LoCoMo and 11.2% of MuSiQue answers that cover-EM rejects. Most of those are the label's fault. Reading a sample of them turns up correct paraphrases (gold "Car mod workshop", prediction "Dave got picked for a car modification workshop"), correct but rounded values (gold "260,897", prediction "approximately 260,000"), and 12 MuSiQue items whose gold string is the literal placeholder "TBD". Token overlap is itself lexical, so it cannot settle these cases either. We report the share with low overlap as an upper bound on the judge's false-positive rate, 15.9% on LoCoMo and 8.3% on MuSiQue, and we do not read it as an estimate of that rate.

A separate check runs the other way. Agents abstain on 10.0% of answerable LoCoMo questions, 28.4% of answerable MuSiQue questions and 10.6% of SWE-QA-Pro questions, so a judge that rewarded abstention would inflate every condition that abstains more. Those answers score 0.00, 0.03 and 0.10 on average, and at most 2.9% of them clear 0.25, which leaves no such inflation available.

Blank answers. A harness sometimes ends an episode without emitting a final answer, and our judging script skipped those rollouts without scoring them. That pairs the affected cell on fewer questions and lets the failure drop out of the mean, where it should have counted against the arm that produced it. Blanks are rare, and they concentrate almost entirely in ReAct on SWE-QA-Pro, which can exhaust its step budget in the middle of a trajectory: 63 of the 30,525 rollouts in the full-scale run, 16 of 3,600 in `t1_control`, 16 of 3,300 in `e1_endpoints` and 5 of 2,400 in `b2_native`. We score a blank 0 in the two same-window arms of §B.9 and §B.10, which are the ones the replication claim rests on, by setting `EMPTY_AS_ZERO=1` for the analysis scripts. Everywhere else the numbers are as the judge returned them. We also checked what the choice would cost if we applied it everywhere. No pooled contrast in the paper moves by more than 0.7 pp, the +3.0 pp of Table 2 does not move at all, and none of them changes sign. One reported p-value moves, the generic-summarization comparison on the tool interface in §B.14, from $p < .001$ to $p = .005$.

Does the judge prefer a policy model? Section 6.3 compares `gpt-4o-mini` against Llama-3.3-70B-Instruct under a `gpt-4o` judge, so the judge shares a family with one of the two policies. The two tags cover the same questions under the same judge, so we can condition on the gold label and ask whether the judge still separates the models once correctness is held fixed. On MuSiQue it does not. Among answers cover-EM calls correct the judge gives `gpt-4o-mini` 0.931 and Llama 0.950, and among answers it calls wrong, 0.201 and 0.208. Paired within question, method and condition ($n = 360$), the judge puts `gpt-4o-mini` ahead by 8.5 points where the gold label puts it ahead by 7.8. The gap the judge reports is thus the gap the gold answers report. The same paired comparison on LoCoMo gives 13.0 points against 8.5. However, LoCoMo is the task whose label we have just shown to be unreliable, so we read the MuSiQue comparison as the test.

Do the reported contrasts depend on the judge? Rescoring all 36 cells by cover-EM alone, the spread between cells is no larger than sampling noise under both scorers. On MuSiQue the standard deviation of the cell means is 0.020 under gold scoring against a typical within-cell standard error of 0.047, and on LoCoMo 0.053 against 0.062. Pooled over cells, the freeform, summary and model-managed regimes score 0.345, 0.338 and 0.325 by cover-EM on MuSiQue, against 0.445, 0.453 and 0.455 by the judge. Neither scorer resolves a difference between the three regimes on

pooled accuracy, which is what §6.2 reports. The ordering flips between the two scorers inside a band narrower than the standard error either way. That is what we would expect when the underlying difference is near zero, and it is why we rest the ownership claim on state fidelity and calibration instead.

A labeled set whose labels are constructed rather than lexical. Gold-anchored agreement conflates two failure modes, a judge that rewards length and a judge that misses a paraphrase, because cover-EM is wrong in both directions on exactly those cases. `rebuttal/judge_probe.py perturb` separates them. It builds five answers per question whose correctness is fixed by construction: the gold verbatim, the gold wrapped in 17 words of filler, an LLM paraphrase of the gold, another question's gold of the same answer shape, and an abstention. The wrapper and the distractor each differ from the gold in one controlled way, so a judge that scores them wrongly is making the corresponding error and is not reacting to something else in the answer. The distractor is type-matched (number, date, short, medium, long) and checked for substring containment in both directions, since cover-EM would score a prediction that contains the gold as correct. Over 300 questions and $1{,}422$ judgements the judge credits $300/300$ verbatim golds, $300/300$ wrapped golds and $221/222$ paraphrases, and credits $0/300$ distractors and $0/300$ abstentions. That is a miss rate of 0.1% on constructed-correct answers and a false credit rate of 0.0% on constructed-wrong ones. Padding an answer with filler moves the score by -0.005 (95% CI $[-0.009, -0.002]$) and paraphrasing it by -0.017 ($[-0.027, -0.009]$), both against the direction a verbosity-rewarding judge would move. Hence the 22.4% and 11.2% of judge-high, cover-EM-wrong answers above are mostly the label's doing, which is what reading them suggested.

A judge from a different family. Sharing a vendor between judge and policy is the other standard worry. `rebuttal/judge_probe.py crossjudge` re-scores a stratified subsample of $1{,}595$ stored answers with `gemini-2.5-pro` under the identical rubric. The two judges rank answers almost identically ($\rho = 0.923$, Pearson 0.921) and agree on 93.8% of binary decisions ($\kappa = 0.861$). `gemini-2.5-pro` is the more generous of the two by 4.3 points overall ($[3.5, 5.0]$), and that difference is concentrated in one task. The means are 0.490 against 0.502 on LoCoMo and 0.481 against 0.481 on MuSiQue, but 0.247 against 0.428 on SWE-QA-Pro, which is the task where we have no gold label to adjudicate between them. The overall figure pools a subsample that is not balanced across tasks, at 7 to 79 rows per cell, so it falls between the near-zero gaps on LoCoMo and MuSiQue and the 18-point gap on SWE-QA-Pro, and matches none of the three. The shift is close to constant within a task, so it moves both arms of a contrast together. The mean cell-level contrast from the history-removed condition to z^{f} is $+1.3$ pp under `gpt-4o` against $+0.9$ pp under `gemini-2.5-pro`, agreeing in sign in 10 of the 12 cells. We read that as agreement about levels and ordering; it is not a replication of the paper's effects, since 7 to 79 rows is far too few to resolve per-cell differences of the size §6.1 reports.

Limits of the probe. We build the constructed-label probe from LoCoMo and MuSiQue golds, so the judge is still unvalidated on SWE-QA-Pro, where no string label is defensible and where the two judges also disagree most about the level. The probe establishes something narrower there. An independent judge under the same rubric ranks the same answers the same way, so the SWE-QA-Pro numbers are comparable across conditions even though their absolute level is not pinned down. Fixing that level would need reference answers written for the purpose, which is the one piece of this analysis that would still need human annotation.

F.4 Additional Statistics

Tables 29 and 30 give the distribution of retrieval rounds, and the effect of task complexity (hop count) on each intervention.

F.5 Schema artifacts in the structured state

Table 31 puts a number on the schema artifact of §5.1 and §6.1. We count how often the structured extractor fills a required field with a literal `no evidence` placeholder.

Table 29: Mean and median retrieval rounds per episode. Max rounds: IRCoT 10, ReAct 7, Iter-RetGen 4 (fixed), MemGPT 12. Iter-RetGen always uses all 4 rounds (omitted).

		Mean rounds				Median rounds			
Task	**Harness**	**B**	**L**	$\mathbf{z}^{\mathrm{s}}$	$\mathbf{z}^{\mathrm{f}}$	**B**	**L**	$\mathbf{z}^{\mathrm{s}}$	$\mathbf{z}^{\mathrm{f}}$
LoCoMo	IRCoT	3.6	4.6	4.3	4.0	2	3	2	2
	ReAct	3.4	5.2	4.1	4.4	3	7	3	3
	MemGPT	1.6	2.1	1.1	1.2	1	1	1	1
MuSiQue	IRCoT	5.9	7.6	6.9	6.9	5	10	8	8
	ReAct	5.1	6.9	4.7	5.2	5	7	4	6
	MemGPT	5.9	7.4	3.0	2.6	3	12	2	2
SWE	IRCoT	8.0	8.0	8.1	8.0	10	10	10	10
	ReAct	4.6	6.9	5.8	5.6	4	7	7	7
	MemGPT	4.2	4.8	3.3	2.5	1	1	1	1

Table 30: MuSiQue z_t effect (pp) by hop count. Iter-RetGen has no history-removed condition. **Bold blue** = $p < .05$.

Harness	**Hops**	**N**	**B**$\rightarrow z^{\mathrm{s}}$	**B**$\rightarrow z^{\mathrm{f}}$	**L**$\rightarrow z^{\mathrm{s}}$	**L**$\rightarrow z^{\mathrm{f}}$
IRCoT	2-hop	70	+1.1	**+15.0**	+4.4	**+18.3**
	3-hop (t1)	97	+3.5	+5.7	**+9.6**	**+11.8**
	3-hop (t2)	33	−6.2	+2.3	−2.0	+6.5
	4-hop (t1)	185	+2.1	+2.7	**+10.2**	**+10.8**
	4-hop (t2)	49	+2.0	+0.7	+2.5	+1.2
	4-hop (t3)	66	0.0	+5.5	+0.5	**+6.0**
ReAct	2-hop	70	−1.7	+6.0	+9.6	**+17.3**
	3-hop (t1)	97	+2.1	**+9.1**	**+10.5**	**+17.5**
	3-hop (t2)	33	**+17.0**	**+20.8**	**+16.4**	**+20.2**
	4-hop (t1)	185	+3.7	**+8.0**	**+8.6**	**+12.9**
	4-hop (t2)	49	+1.7	**+9.2**	**+6.1**	**+13.5**
	4-hop (t3)	66	−1.5	−2.6	**+10.3**	**+9.2**
Iter-RetGen	2-hop	70	−0.8	+4.0	—	—
	3-hop (t1)	97	−9.4	+0.4	—	—
	3-hop (t2)	33	−0.8	+5.9	—	—
	4-hop (t1)	185	−5.6	+4.5	—	—
	4-hop (t2)	49	−5.6	−4.3	—	—
	4-hop (t3)	66	−4.5	−2.2	—	—
MemGPT	2-hop	70	0.0	+0.1	+1.1	+1.2
	3-hop (t1)	97	−1.1	+4.0	−2.5	+2.6
	3-hop (t2)	33	+0.4	+2.4	+1.6	+5.0
	4-hop (t1)	185	+0.2	+1.2	+1.4	+3.8
	4-hop (t2)	49	+4.1	+7.7	+0.4	+3.9
	4-hop (t3)	66	+2.1	+1.7	+1.0	+1.5

G Stagnation Gate Configurations

The tables below name a gate configuration by its thresholds in Eq. 7, so `f_j0.6_u0.3_p2` sets $\tau_J = 0.6$, $\tau_U = 0.3$ and $p = 2$, with τ_D left at 1.0 throughout. The leading `f` is the trigger mode `full`, meaning the gate requires all three conjuncts rather than the two query metrics alone. We call a configuration the *global best* when it has the highest mean accuracy change over all 45 harness $\times$ dataset $\times$ condition cells at once, with no per-cell tuning, and `f_j0.6_u0.3_p2` is both that configuration and the one the body freezes.

G.1 Per-method gate results

Table 33 reports the stagnation gate effect for each method separately, across all tasks and conditions. **Bold blue** = $p < .05$.

Table 31: Schema artifacts. Percentage of episodes in which the structured state contains at least one literal `no evidence` placeholder, and percentage of rounds so affected. Freeform states never produce the artifact.

Harness	Domain	Episodes (%)	Rounds (%)
IRCoT	MuSiQue	37.7	63.9
	LoCoMo	56.5	93.7
	SWE-QA-Pro	41.2	72.3
ReAct	MuSiQue	40.7	72.6
	LoCoMo	56.2	97.0
	SWE-QA-Pro	42.3	76.9
Freeform (all cells)		0.0	0.0

Table 32: Stagnation Gate with thresholds frozen from the `gpt-4o-mini` sweep and applied unchanged to Llama-3.3-70B-Instruct. Δ is the accuracy change from adding the gate; non-inferiority is assessed against $\varepsilon = 2$ points. "Fire" is the share of episodes in which the gate interrupts, "Save" the net token change including orchestrator calls.

UPDATE owner	n	Δ (pp)	95% CI	Fire (%)	Save (%)
OMS (orchestrator)	539	$+0.3$	$[-1.1, +1.7]$	30	16.7
None (history-removed)	540	-1.3	$[-3.2, +0.6]$	43	12.6

Table 33: Stagnation gate (global best configuration, $f_j0.6_u0.3_p2$). Δ: score difference in pp. Fire: % of episodes where the gate triggers. Save: net token savings (%). **Bold blue** = $p < .05$. †Iter-RetGen (memoryless by design) has no history-removed condition.

		Δ (pp)				Fire (%)				Save (%)			
Harness	Task	B	L	z^s	z^f	B	L	z^s	z^f	B	L	z^s	z^f
IRCoT	LoC	**+3.3**	**+2.2**	**+3.8**	**+2.9**	20	24	21	15	13	18	19	15
	MuS	**+5.7**	+0.4	+0.3	+0.4	33	42	28	22	9	15	11	9
	SWE	**+16.2**	**+1.9**	**+4.4**	**+4.9**	80	47	39	38	39	17	15	17
ReAct	LoC	+0.2	**+3.0**	−0.5	−0.2	4	49	21	25	−2	12	9	11
	MuS	+0.8	−0.1	−0.2	+0.1	18	39	14	18	−2	−18	4	3
	SWE	**+2.3**	−0.9	+0.3	−1.2	14	37	27	30	−3	−20	3	8
Iter-RetGen†	LoC	**−5.2**†	—	**+0.7**	+0.4	60†	—	16	15	−8†	—	−1	−1
	MuS	**−2.6**†	—	+0.5	−0.2	11†	—	2	3	−1†	—	0	0
	SWE	**−0.7**†	—	+0.0	−0.3	4†	—	0	2	0†	—	0	0
MemGPT	LoC	−0.1	−0.1	+0.0	−0.1	3	10	1	1	6	37	5	8
	MuS	**+1.8**	**−9.0**	+0.7	−0.2	35	57	13	10	31	52	26	24
	SWE	**+2.2**	**−1.8**	**+2.3**	+0.5	28	35	20	13	45	56	42	38

IRCoT benefits most because it has the most retrieval rounds (up to 10), and so the most opportunity for stagnation to develop. ReAct (up to 7 rounds) shows moderate benefits on the baseline and history-removed conditions. MemGPT has variable round counts and less predictable stagnation patterns. The gate hurts it on MuSiQue history-removed (-9.0pp) because it triggers during productive search episodes. Iter-RetGen's 4 fixed rounds mean the gate fires at the last round, where there is nothing left to stop early, so it produces 1/9 significant positives and 3/9 significant negatives.

G.2 SMOOTH VS. DISCRETE CONFIGURATIONS

We swept 16 configurations: 7 discrete (hard thresholds on Jaccard and UPR) and 9 smooth (exponentially-weighted moving averages). On IRCoT the smooth configurations give a slightly better mean improvement (+7.2pp) than the discrete ones (+6.4pp), which fits the intuition that stagnation is gradual.

G.3 ANCHORING: WHY THE GATE HELPS STATEFUL METHODS

Table 34 shows the pooled condition-level impact. The gate significantly helps the native, z^{s} and z^{f} conditions ($p < 0.001$), and is neutral on history-removed ($p = 0.248$). Table 35 supports the anchoring hypothesis. In the history-removed-vs-z_t comparisons shown below, the per-question variance of the gate's effect is consistently higher without persistent state. We read this as the information state anchoring the method's reasoning, so that the early-stop answer is closer to what the method would have produced on its own. Memoryless methods, given similar passages, produce higher-variance answers.

Table 34: Pooled condition-level stagnation gate impact (global best config, all methods). The gate helps the native, z^{s}, and z^{f} conditions and is neutral on history-removed, where the state-change conjunct of Eq. 7 carries no information.

Condition	**Diff (pp)**	p	**Sig**
Native	$+0.7$	.0004	***
History-removed	$+0.3$	.248	
z^{s}	$+0.9$	$<.0001$	***
z^{f}	$+0.6$	$<.0001$	***

Table 35: Per-question standard deviation of the gate's effect on the rubric score, by harness and dataset. The history-removed condition varies more than either z_t condition in all six rows, which is what we would expect if a state that persists across rounds makes the decision to stop more reliable.

Harness	**Dataset**	**Native**	**Hist.-rem.**	$\mathbf{z}^{\mathrm{s}}$	$\mathbf{z}^{\mathrm{f}}$
IRCoT	LoCoMo	.352	**.458**	.345	.343
IRCoT	MuSiQue	**.397**	.305	.185	.205
IRCoT	SWE	.150	**.203**	.173	.170
ReAct	LoCoMo	.277	**.441**	.324	.297
ReAct	MuSiQue	**.254**	.230	.204	.199
ReAct	SWE	.212	**.214**	.167	.202

G.4 THE STATE-DIFF TERM

We compare `full` mode (checks Jaccard + UPR + state diff) against `query_and_full` mode (checks Jaccard + UPR only) at matched thresholds. 194 of 225 cells are identical (0pp difference). The diff check only matters for the discrete $U{=}0.3$ configs, where it cuts the fire rate by 1.8–5.8pp. For smooth configs ($U < 0.3$) the diff threshold never binds at all. The mean delta between modes is -0.04pp. Within this sweep, then, Jaccard and UPR carry the stagnation signal, and dropping the diff term changes little.

This does not mean the term is inert. Every run in the sweep above has an UPDATE owner, so Δz_t is a real quantity. It is satisfied in 57.7% of $2{,}848$ stateful rounds and fails — and so can bind — in the other 42.3%. Removing the term while keeping the state moves the fire rate by only a few points, which is what this sweep measures. Removing the *state* is a different manipulation. With no UPDATE owner the conjunct is vacuously true in 100.0% of $3{,}014$ rounds, the gate degenerates to a lexical loop detector, and it fires on 13.1 pp more episodes ($p = 4.6 \times 10^{-7}$; §B.8). The two findings are consistent and should be read together. As a threshold among thresholds the state diff is close to redundant. As the coupling between STOP and UPDATE it is what keeps the monitor from stopping on repetition alone.

Token overhead estimation methodology. We estimate final-answer (FA) call tokens the same way for every gate variant, programmatic and LLM alike:

- *Prompt tokens:* exact, via tiktoken on the actual evidence, question, system prompt and template. We validated these against the API's own prompt-token counts in an FA token ablation, and the ratio stays within 0.93–1.06× across all 45 method/task/condition cells.
- *Completion tokens:* the per-method/task/condition average from the same ablation (100 randomly chosen triggered questions per cell, with real `gpt-4o-mini` calls).

Table 36: Stagnation gate fire round distribution (global best config). Iter-RetGen always fires at round 4 (its maximum). †Natively memoryless.

		Mean fire round				Median fire round			
Harness	**Task**	**B**	**L**	$\mathbf{z^s}$	$\mathbf{z^f}$	**B**	**L**	$\mathbf{z^s}$	$\mathbf{z^f}$
IRCoT	LoC	4.4	4.5	5.5	5.0	4	4	5	4
	MuS	5.8	5.9	6.5	6.0	6	6	6	6
	SWE	4.2	6.2	5.8	6.0	4	6	6	6
ReAct	LoC	3.8	4.5	5.0	4.9	3	4	5	5
	MuS	5.5	5.9	5.3	5.3	6	6	5	5
	SWE	5.1	6.1	5.3	5.2	5	7	5	5
Iter-RetGen†	LoC	4.0	—	4.0	4.0	4	—	4	4
	MuS	4.0	—	4.0	4.0	4	—	4	4
	SWE	4.0	—	4.0	4.0	4	—	4	4
MemGPT	LoC	5.4	3.4	3.3	3.3	5	3	3	3
	MuS	7.1	4.9	5.3	4.5	7	5	5	4
	SWE	5.7	3.6	4.1	3.7	6	3	4	4

Programmatic gates add no decision overhead at all, since they are metric-based and make no LLM call. LLM gates add the logged `gate_tokens`, one per-round "should I stop?" call. The total overhead is the gate decision plus the FA call estimate. We use the same FA estimator for every gate, so the only thing that differs between variants is how often and where they fire.

Configuration clustering. Clustering the configurations by L_1 distance on their 45-cell mean-diff vectors gives two groups:

1. *Conservative discrete* (U=0.3, p=2–3): moderate fire rates and fewer negatives. The global best ($f_j0.6_u0.3_p2$) is in this band.
2. *Aggressive smooth* (U=0.10–0.15, p=2): high fire rates, with more negatives on the non-IRCoT methods but larger gains on IRCoT.

Three pairs of configurations are functionally identical ($L_1 = 0$pp), because the `full` and `query_and_full` trigger modes coincide whenever the diff threshold does not bind. The global config is close to optimal. Its L_1 distance to the per-method-and-condition diff vectors averages only 0.64pp per cell.

H Ablations

H.1 Retriever type (alpha)

We ask whether the z_t pattern holds under extreme retriever settings: ReAct with α=1.0 (pure BM25, keyword only) and MemGPT with α=0.0 (pure dense embeddings, semantics only). Table 37 reports the key comparisons.

Table 37: z_t effect (pp) under extreme retriever alpha. The recovery-from-history-removal pattern persists across retriever types, although baseline comparisons vary by task and state format. **Bold blue** = $p < .05$.

Harness (α)	**Task**	**B**→ z^s	**B**→ z^f	**L**→ z^s	**L**→ z^f
ReAct (α=1.0)	LoCoMo	**+3.5**	**+2.3**	**+7.9**	**+6.8**
	MuSiQue	**+7.5**	**+9.6**	**+21.5**	**+23.6**
	SWE-QA-Pro	**−2.8**	+2.2	+1.7	**+6.7**
MemGPT (α=0.0)	LoCoMo	**+4.7**	**+2.8**	**+5.2**	**+3.2**
	MuSiQue	−1.1	+2.7	+1.4	**+5.2**
	SWE-QA-Pro	**−8.4**	−1.1	**−3.8**	**+3.5**

The recovery-from-history-removal pattern is strongest for the freeform z_t, which improves over the history-removed condition in all six rows and is significant in five. Structured z_t is more mixed under

these extreme retriever settings, especially on SWE-QA-Pro. The recovery is not an artifact of one retriever.

H.2 Information state capacity, $K \in \{3, 6, 12\}$ (arm e4a-ksweep)

The capacity bound K is the one free parameter of Update that Observation 1 speaks to directly, so we sweep it over a factor of four — $K = 3$, the default $K = 6$, and $K = 12$ — on IRCoT and ReAct across all three domains, with the free-form state and everything else held. We ran all three points in one window against one model alias, so the contrasts below are not exposed to the drift of Appendix B.3. An earlier two-point version of this ablation, run several months before on the structured state, reported the same null.

Table 38: Capacity sweep against the default $K = 6$, free-form state, paired within question. Halving the bound and doubling it move accuracy by at most 6.6 pp in any cell: none reaches $p < .05$, and both pooled intervals contain zero.

Harness	Task	n	K=3	K=6	K=12	Δ_{3-12}
IRCoT	LoCoMo	100	75.2	78.5	80.0	−4.8
ReAct	LoCoMo	100	69.8	72.8	72.5	−2.7
IRCoT	MuSiQue	100	43.8	40.0	37.2	+6.6
ReAct	MuSiQue	100	50.5	47.2	50.0	+0.5
IRCoT	SWE-QA-Pro	100	34.5	33.4	32.0	+2.5
ReAct	SWE-QA-Pro	84–88	43.9	42.9	42.9	+1.0
pooled K=3 vs. K=6			+0.5 pp, CI $[-1.8, +2.7]$, $p = .692$, −2.1% tokens ($n = 588$)			
pooled K=12 vs. K=6			−0.1 pp, CI $[-2.3, +2.0]$, $p = .893$, +2.4% tokens ($n = 584$)			

K does not matter on these corpora, in either direction, and the token footprint barely moves with it either. The extractor writes about the same amount at $K = 3$ as at $K = 12$, because on questions of this shape it has fewer than three things worth keeping. This is a limit of the corpora. It does not refute Observation 1, which is about chain length: LoCoMo and MuSiQue questions resolve in two or three hops, so $k \log_2 N$ is small and any of these bounds is slack. Repeating the sweep where depth is varied on purpose gives the same answer at every depth — $K = 3$ and $K = 12$ are within 5.0 pp of $K = 6$ at each k — while the recovery itself falls from +13.3 pp at $k = 2$ to nothing at $k = 6$. As depth grows the extractor captures less, and its capacity is not the binding constraint (§6.3, Appendix B.13).

Reproduction: `bash rebuttal/e4a_ksweep.sh`, then `judge.sh -tag e4a_ksweep`, then `contrast.py -tag e4a_ksweep -methods ircot,react -ref '{b}_st_freeform' -arms '{b}_st_freeform_k3,{b}_st_freeform_k12'`.

H.3 Retrieval depth (k=3, k=5, k=10)

We test whether the z_t pattern holds at different retrieval depths on IRCoT × LoCoMo.

Table 39: z_t effect (pp) at different retrieval depths (IRCoT × LoCoMo). Recovery from history-removed holds at all depths, while comparisons to baseline narrow as retrieval depth increases. **Bold blue** = $p < .05$.

Depth	B→ z^s	B→ z^f	L→ z^s	L→ z^f
k=3	+1.7	**+2.0**	**+6.0**	**+6.2**
k=5 (canonical)	+0.6	**+2.4**	**+6.4**	**+8.2**
k=10	−0.8	+0.1	**+4.7**	**+5.7**

Both z_t formats significantly recover from history-removed at every k. A higher retrieval depth (k=10) improves all conditions, which narrows the gap between baseline and z_t.

H.4 Information State Reorganization – Token Overhead Analysis

The orchestrator's reorganization step fires when the information state exceeds $K_{\text{trigger}} = 10$ items, and curates it down to $K_{\text{target}} = 6$. To price it, we measured token usage across all 16,280 episodes

run under the explicit information state (z_t) conditions. The step is cheap, and it only fires on the long-horizon episodes:

- **Trigger Frequency:** 20.8% of episodes ($3{,}385/16{,}280$) triggered the reorganization step at least once.
- **Call Overhead:** For episodes that did require reorganization, the orchestrator made an average of 1.86 reorganization calls, with each call consuming on average 979 tokens.
- **Relative Cost:** The total mean overhead for reorganization was $1{,}824$ tokens per triggered episode. This accounted for 11.4% of the total token consumption in those episodes.

Episodes that triggered reorganization did cost substantially more in total ($16{,}010$ tokens against $5{,}887$), but that difference is largely confounded by task complexity. Episodes that exceed the $K = 10$ threshold are the longer-running multi-hop searches, which take more environment steps anyway. The reorganization step itself stays light ($\sim 1{,}824$ tokens), so the agent's context window is kept bounded at a cost that does not dominate the episode.

I Observed Failure Modes and Example Traces

I.1 Observed Failure Modes

Appendix I.2 gives an annotated trace for each of the seven failure modes (G1–G7) our evaluation tasks surfaced. We summarize here several more that the CGDP framework suggests may arise in longer episodes, larger corpora, or harder reasoning settings.

Coverage unawareness. The agent cannot tell "I did not find it" apart from "it is not in the corpus". Current methods carry little notion of coverage, so they cannot say how much of the search space they have already seen. Explicit coverage tracking would be the natural fix.

Aggregation breakdown. Some questions ask the agent to combine many weak signals across passages, e.g. counting entities, summarizing a trend, or comparing sources. LLMs get less reliable as the number of relevant passages grows (Schnabel et al., 2025), so the state update can fail even once the evidence is in hand.

Error compounding. A small extraction mistake in an early round becomes a premise for later reasoning. Persistent state helps, because it keeps each extraction step short and lets a later round revise it, but it does not remove the problem.

A wrong action-observation model. The agent searches the wrong part of the corpus because its picture of how actions map to observations is wrong. The actions it chooses are locally sensible, but they do not match the structure of the corpus. A lightweight predict-before-retrieve step would be the natural fix.

Unjustified confidence. The agent may stop with high confidence because the answer sounds plausible, even though the evidence is incomplete. This happens when stopping depends on fluent output more than on whether the key predicates have actually been resolved. A natural fix would be evidence-grounded stopping or calibrated answer generation.

I.2 Annotated Agent Traces

Every trace below comes from a baseline condition of our experiments, with no z_t injection and no stagnation gate.

G5/G6: Retrieval Stagnation (IRCoT, MuSiQue). Retrieval stagnates when the agent issues the same, or nearly the same, search action round after round, gets back the same passages, and makes no progress. We count an episode as stagnant when the token-level Jaccard similarity between consecutive actions exceeds 0.5 for at least two consecutive pairs.
Example: User query: *"What city shares a border with the place where the person who went to the*

state known for its Mediterranean climate during the gold rush worked?" Ground truth: Rio Linda. IRCoT runs 10 rounds (maximum) and answers incorrectly.

> R1: `To answer the question, we need to identify a city that shares a border with a location associated with the Mediterranea...`
> R2: `The Black Hills Gold Rush took place in Dakota Territory, which is not associated with a Mediterranean climate.`
> R3: `To answer the question, we need to identify a city that shares a border...`
> R5–R10: Repeat of R1/R3 pattern with minor variations.

Annotation. The agent alternates between two framings and never advances the multi-hop chain. It has no mechanism for noticing that it is stuck, or for trying a new direction, so it stays in the loop. The stagnation gate would flag this early, on action overlap alone.

G4: Premature Stopping (MemGPT, LoCoMo). The agent stops prematurely when it answers after one or two rounds, wrongly, on evidence that was never enough.
Example. User query: *"Who performed at the concert at Melanie's daughter's birthday?"* Ground truth: Matt Patterson. MemGPT runs 1 round, retrieves 5 passages with action "`concert Melanie's daughter's birthday`", and answers UNANSWERABLE.
Annotation. The first retrieval misses, and the agent stops instead of refining its search. With a persistent information state the unresolved question "who performed?" would still be written down, and stopping early would look less attractive.

G2: Goal Displacement (ReAct, MuSiQue). Goals get displaced when the agent's actions drift off the original query and onto a tangential lead.
Example. User query: *"How were people from whom new coins were a proclamation of independence by the Somali Muslim Ajuran Empire expelled from the natural boundary between Thailand and A Magne's country?"* Ground truth: The dynasty regrouped and defeated the Portuguese. ReAct runs 5 rounds.

> R1 action: `Ajuran Empire proclamation of independence and expulsion of people`
> R5 action: `Ajuran Empire Southeast Asia history`

Annotation. The agent follows a salient thread and drifts off the intermediate sub-questions the original query needs. Nothing tracks what is still unresolved, so a tangent costs it nothing to follow.

G1: Evidence Not Persisted (IRCoT, MuSiQue). Evidence is forgotten when relevant information shows up in an early round and then disappears from the later reasoning.
Example. User query: *"Who is the mother of the screenwriter of WarGames?"* Ground truth: Jane Greer. IRCoT runs 10 rounds and answers incorrectly. Round 2 passage contains: *"...He is the son of actress Jane Greer and producer Edward Lasker..."* By rounds 8–10, the chain-of-thought discusses the screenwriters without mentioning their mothers.
Annotation. The evidence arrives early, nothing durable holds it, and it stops shaping the reasoning. A good persistent information state would have written "Jane Greer" down the moment it appeared.

G3: Parametric Dominance (ReAct, MuSiQue). Parametric knowledge dominates when the agent grounds its answer in what it learned during pretraining, ignoring the retrieved evidence.
Example. User query: *"How many mandatory transmitters of the Canadian Broadcasting Centre's owner were updated before the deadline?"* Ground truth: only about half. ReAct runs 3 rounds.

> R2 Thought: `The observations indicate that the CBC did not convert all of its mandatory transmitters to digital by the original deadline of August 31, 2011...`
> Answer: `15 mandatory transmitters were updated before the deadline`

Annotation. The final answer names a specific number that the retrieved evidence does not support. The model's prior wins over the passages, because nothing keeps that evidence in an explicit, persistent state.

G7: Evidence-Action Misalignment (LoCoMo). Agents fold retrieved evidence into the answer without checking that it is about the entities the user asked about.
Example. User query: *"What was grandma's gift to Melanie?"* (entity-swapped; the original conversation discusses a gift to a different person). IRCoT retrieves passages about the necklace gift and answers "A necklace" in 2 rounds without verifying that the evidence pertains to Melanie specifically.
Annotation. The agent finds a passage about a gift, but never checks that the gift belongs to the entity in the query. Entity-grounded extraction, or a lightweight alignment check, would catch the mismatch before the answer is written.

Table 40 counts how often each of these patterns appears across the native harnesses. Stagnation is the most common of the three in eight of the twelve harness $\times$ dataset cells, so the gate targets it, and the remaining patterns would each need a module we did not build (Appendix J).

Table 40: How often each failure pattern appears in the native conditions, with no z_t and no gate. Cells count the episodes in which a lexical detector fired, for four of the patterns of §I.2: G4 answers after one or two rounds and gets it wrong, G5/G6 reissues a near-identical search and gets back the same passages, G2 drifts off the sub-questions the query still needs, and G7 answers from a passage about a different entity than the one asked about. G7 is annotated only on LoCoMo's unanswerable stratum, where the denominator is a fixed 440 episodes. Iter-RetGen stops on a fixed round count, so the premature-stopping detector cannot fire on it. These prevalences are descriptive, computed from string-overlap heuristics, and the labels behind them are not validated (Appendix L).

Gap	Task	IRCoT	ReAct	MemGPT	Iter-RetGen
G4 (premature stop)	MuSiQue	74	4	62	0
	LoCoMo	68	10	143	0
	SWE-QA-Pro	15	0	3	0
G5/G6 (stagnation)	MuSiQue	115	146	173	139
	LoCoMo	248	70	24	986
	SWE-QA-Pro	197	51	71	37
G2 (goal displacement)	MuSiQue	114	236	180	0
	LoCoMo	76	54	6	0
	SWE-QA-Pro	9	61	20	0
G7 (evidence-action misalign.)	LoCoMo una.	125/440	73/440	229/440	—

J Additional Interventions Suggested by the Framework

The gap analysis suggests several further orchestrator-level interventions. They are outside the scope of this paper, but they follow naturally from the framework.

Gap-aware stopping. Instead of asking only whether search has stalled, the orchestrator could make a focused LLM call and ask which predicates are still unresolved. That targets premature stopping, the failure mode opposite to the one the stagnation gate catches, so the two would complement each other.

Entity-grounded extraction and alignment checking. We could require the extractor to ground every finding against the entities and relations in the user query. A separate verification step would then catch any mismatches that remain, before the answer is written.

Triggered resampling. When stagnation is detected, the orchestrator could branch to one or more new hypotheses that are still consistent with the accumulated observations, rather than terminating immediately. Changes of direction would then be explicit, and no longer left to the model's chain-of-thought.

Observation filtering. The orchestrator could filter or compress the raw observation before extraction, so that clearly irrelevant text never reaches the information-state update.

Parallel hypothesis aggregation. Another way to explore multiple hypotheses is to run several retrieval trajectories in parallel and aggregate them afterward (Lee et al., 2026). Hypothesis diversity then becomes an explicit orchestrator decision, no longer an accidental byproduct of a single trajectory.

K Extended Related Work

K.1 Relation to an earlier version of this work

An earlier version of this work called the loop Predicate-Based Adaptive Identification and defined the information state through logical predicates. We retire that framing. The best-performing implementation uses no predicate formalism, and no result depends on one. Structured fact/open-question fields remain only as a representation ablation (§5.1), where they underperform freeform text. The stopping rule was likewise called the "Exhaustion Gate". We renamed it the Stagnation Gate, because the old name invited the stronger reading that the corpus had been exhausted, when what is exhausted is the trajectory.

K.2 Our additions to prior art

Running summaries, evidence checklists and external memory tools are prior art (Rozanov & Rei, 2025; Aghajani Asl et al., 2025; Packer et al., 2023; Yen et al., 2026). Textual memory is not our idea. We add three things: (i) a portable update interface *enforced by the orchestrator* and separated from both the policy and the stopping logic, (ii) matched controls that isolate that separation from representation and from query conditioning, and (iii) a stopping rule whose input is the state that update produces, so it behaves differently when no module owns the update.

K.3 Prior work by architectural layer

The main text (§2) groups prior work by the architectural layer it modifies. We expand those groups here. **Policy and action selection.** IRCoT (Trivedi et al., 2023) interleaves retrieval with chain-of-thought, Iter-RetGen (Shao et al., 2023) conditions the next query on the previous generation, and ReAct (Yao et al., 2023) structures trajectories as thought–action–observation triples. Search-R1 (Jin et al., 2025), Search-o1 (Li et al., 2025a) and WebThinker (Li et al., 2025b) train or prompt the policy to interleave search with reasoning. All of these improve *which* query is issued, and leave state management to context accumulation. **Offline corpus organization.** A-MEM (Xu et al., 2025), HippoRAG (Gutiérrez et al., 2024), HopRAG (Liu et al., 2025) and GraphRAG (Edge et al., 2025) restructure the corpus into graphs before any query arrives, which changes $P(o \mid a, c)$. We leave the corpus untouched and organize online what the agent has learned from it. The two compose. **Stopping and retrieval initiation.** FLARE (Jiang et al., 2023) and DRAGIN (Su et al., 2024) decide *whether to retrieve* from token confidence or attention. Our gate addresses the opposite end of the round, namely whether a trajectory that has already retrieved repeatedly has stopped producing new evidence.

Table 41 organizes the systems discussed in §2 and below by the architectural layer they modify.

K.4 Agentic RAG Systems

Search-R1 (Jin et al., 2025) and Search-o1 (Li et al., 2025a) train the retrieval policy with reinforcement learning, and WebThinker (Li et al., 2025b) carries that to web scale. SLIM (Yen et al., 2026) leaves the policy alone and compresses the trajectory every few rounds. All three change what the policy does or what it reads. We change who owns the update and who decides to stop, and leave the policy as its authors shipped it. Any of these can run inside our harness; we do not set them against it.

K.5 LLM Metacognition and Self-Assessment

Our finding that programmatic stagnation detection is more token-efficient than LLM-judged stopping agrees with a growing body of evidence on the metacognitive limits of LLMs. Feng et al. (2024) show that multi-LLM probing improves abstention by 19.3% over single-model self-assessment, and Chen et al. (2023) show that verbalized confidence is poorly calibrated across model families. This is

Table 41: Prior work by architectural layer. **Owner**: who writes the state. **Freq.**: how often it is written. **Bound**: enforced size limit. **Repl.**: replaces the raw observation history rather than supplementing it. **Prov.**: records where a fact came from. **Neg.**: records searched-but-not-found or coverage. **Stop input**: what the stopping decision reads. **Train**: requires training. **Env.**: assumed environment. For OMS, *novelty* abbreviates the action-Jaccard and unique-passage-rate signals of §5.2. "?" marks a property we did not evaluate directly and that the cited work does not state, so such cells support no comparative claim.

Method	Owner	Freq.	Bound	Repl.	Prov.	Neg.	Stop input	Train	Env.
Policy / action selection									
IRCoT, ReAct	— (context)	every step	ctx. window	no	no	no	policy tokens	no	static
Iter-RetGen	— (last gen.)	every step	one gen.	yes	no	no	fixed rounds	no	static
Search-R1, WebThinker	— (context)	every step	ctx. window	no	no	?	policy tokens	yes	static
Online state / trajectory management									
StateAct	policy	every step	prompt slots	no	no	?	policy tokens	no	static/dyn.
FAIR-RAG	policy (LLM)	per round	checklist	no	partial	gaps	checklist	no	static
MemGPT	policy (tools)	when called	paged blocks	partial	no	no	policy tokens	no	static/dyn.
SLIM	framework	periodic	summary	partial	?	?	policy tokens	no	static
Running summary	policy	per round	summary	yes	no	no	policy tokens	no	static
Offline corpus organization									
A-MEM, HippoRAG, GraphRAG	— (corpus)	offline	n/a	n/a	yes	no	policy tokens	no	static
Stopping / retrieval initiation									
FLARE, DRAGIN	—	n/a	n/a	n/a	n/a	no	token conf. / attn.	no	static
This work									
OMS + Stagnation Gate	orchestrator	every obs.	$K \leq 6$ items	yes	yes	partial*	z_t + novelty	no	static

* The update prompt preserves unresolved sub-questions, but does not guarantee a record of every attempted-and-failed query. See §5.1 for the exact invariant we can claim.

what motivates our design choice. The stagnation signals we measure (action Jaccard and UPR) are observable facts about the retrieval process, so the LLM never has to assess its own search progress.

K.6 Consistency and Contradiction Detection

Evidence-action misalignment connects to a literature on contradiction detection in LLM outputs. Mündler et al. (2024) report roughly 80% F1 for pairwise contradiction detection and substantially less at the document level. Detection works when the two facts appear together in a short context, and degrades when one of them has to be recovered from a long history. A bounded z_t puts them together, since each new extraction is compared against a short state in a focused pairwise call. A harness whose only record is the raw trajectory cannot do that.

K.7 Generative Agents and Long-Term Memory

Park et al. (2023) introduced generative agents with long-term memory for simulated social environments. Their memory architecture (observation $\rightarrow$ reflection $\rightarrow$ planning) is structurally similar to our information state management (observation $\rightarrow$ extraction $\rightarrow$ state update), but the setting differs. Their agents interact with a changing social world (a full POMDP), while ours search a static database (sequential identification). The MAKER framework (Meyerson et al., 2025) extends agentic memory to long-horizon creative tasks.

L Extended Limitations

This section expands the limitations summarized in §7. **Static corpus.** Settings with write actions need the transition kernel $P(c' \mid c, a)$ that Eq. 2 excludes. Definition 1 does not cover them, and our results say nothing about them. **Model coverage.** Llama-3.3-70B-Instruct is not a frontier model, and the recovery effect was 4.8 points smaller there — the direction in which a frontier model would likely move it further. We read the cross-family claim as a transfer demonstration on two mid-tier models. It is not a scaling result. **Gate thresholds.** We establish non-inferiority for one frozen threshold transfer on one model. The margin $\varepsilon = 2$ accuracy points is a judgment about acceptable loss, and we do not derive it, so a different margin can reverse the conclusion. **Detector validity.** The

trajectory-pattern counts in Appendix I come from lexical detectors (Jaccard overlap, novelty ratios). We report them as descriptive prevalence, and they are not validated labels. **Judge validity.** Accuracy is a `gpt-4o` rubric score. Appendix F.3 validates it against the gold answers that LoCoMo and MuSiQue already carry, and finds no preference for either policy model once correctness is held fixed. However, we cannot run the same check on SWE-QA-Pro, where the gold is a 200-word reference and no string label is defensible. **Extractor loss.** The extractor can drop evidence that the policy would have used. We see this in the cell where recovery falls short of native (SWE-QA-Pro/MemGPT). **Missing baselines.** We port the StateAct, FAIR-RAG and SLIM update rules into our own UPDATE step (Appendix B.12), and every other comparison is against a matched control. We never run the published systems end to end, so the numbers isolate the architectural variable but do not position the method against the literature's best implementations.