INCUMBENT: THE MODEL PROPOSES, THE STORE DISPOSES — STRUCTURAL TRUTH MAINTENANCE FOR AGENT MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-horizon agents let a language model write their memory: it paraphrases what the user said, infers what they want, and decides when a fact has expired, so the model's own output becomes state that is later retrieved as evidence. We introduce INCUMBENT, a hypergraph memory in which the model proposes and the store disposes: every span it keeps is verified against the user's words, the store assigns provenance so an inference is never read as a user statement, and each attribute holds one current value to which the reader is bound. Against six baselines, INCUMBENT lifts truthfulness on AI-LieDar to 95.0% from the best baseline's 53.3%, raises AMemGym's memory score by 67% (0.344 against 0.206), and reaches 59.0 F1 on LoCoMo while reading 9× fewer context tokens than an agent that holds the transcript. Enforcement, not recording, is what pays: a record of superseded values is worth 0.2 percentage points of AMemGym accuracy to a reader that ranks by similarity and 5.7 to one bound by it.

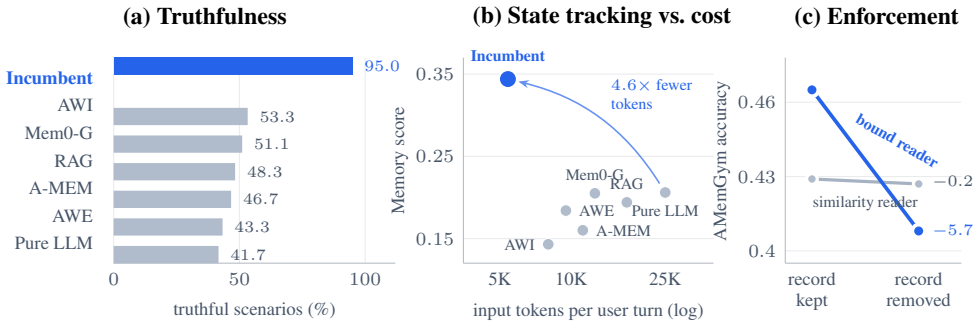

Figure 1: INCUMBENT against six baselines. **(a)** Truthfulness under goal pressure (AI-LieDar). **(b)** Memory score against input tokens per user turn (AMemGym, fixed instructions excluded). **(c)** Removing the record of superseded values, under two readers of INCUMBENT's own store (Table 2).

1 INTRODUCTION

An assistant that works with the same person for months needs a memory outside its context window, and the prevailing way to build one is to let a language model write it (Packer et al., 2023; Zhong et al., 2024; Chhikara et al., 2025; Xu et al., 2025). The model reads the conversation, decides what is worth keeping, rewrites it as notes, facts or graph edges, and later answers from what it wrote. Benchmarks for long-horizon assistants now measure how well this works (Maharana et al., 2024; Wu et al., 2025; Jiayang et al., 2026).

Letting the model write the memory makes every write a point at which the model's own output becomes the agent's state, and three things go wrong there. *Paraphrase*: the stored note replaces the user's words, and summarising loses details that later questions need (Pan et al., 2025). *Promotion*: an inference — the user is probably job hunting — is stored beside what the user said and later retrieved as if the user had said it, and a model with a goal to serve will present such content as fact (Su et al., 2025). *Staleness*: when a user changes jobs or cities, nothing marks the old fact as false unless the

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.

model decides to. In each case the error does not stay local. Once stored, it is retrieved as evidence, and a mistake made at write time becomes a persistent error in state.

Existing memories handle all three by asking the model. Mem0 has it emit an ADD, UPDATE or DELETE for each incoming fact (Chhikara et al., 2025); A-MEM has it decide whether a note should evolve (Xu et al., 2025); the in-context agent of AMemGym overwrites the old value and keeps no record that it changed (Appendix I). Each request is one a model can decline, forget or aim at the wrong record, and an overwrite cannot say what changed. Recording the change does not close the gap either. A store that marks a value superseded and still ranks by similarity hands the stale value to the generator whenever its wording sits closer to the question: with a similarity reader over our own store, 14% of the memory lines rendered for the generator were values the store had itself marked as displaced, on attributes whose current value was one field away.

This is the problem truth-maintenance systems were built to solve (Doyle, 1979; de Kleer, 1986). A reasoner proposes beliefs; a separate system records why each is held and whether it still holds, and retracts what no longer does, so that consistency does not depend on the reasoner remembering to maintain it. We bring that division of labour to language-model memory. In INCUMBENT, the model proposes and the store disposes. Memory is a hypergraph with one hyperedge per attribute of a user, and the store enforces three guarantees that existing memories leave to the model. **Grounding**: a fact keeps the span of the user's words that licensed it only if containment verifies it at write time, and the reader answers from that span rather than from a paraphrase. **Provenance**: facts, open requests and the assistant's own inferences are separately typed, and the store, not the model, assigns their provenance, so an inference cannot reach the generator as something the user said. **Currency**: a hyperedge holds at most one current value, a new value displaces the old as an effect of applying the write (Proposition 2), and the reader is bound to the value in force (Proposition 3). None of the three depends on the model following an instruction; each is a property of how a write is applied or how a read is selected. Because the maintainer (the model call that proposes writes) is shown attributes rather than history, its prompt grows with what a user has said and not with how long they have been talking (Proposition 5).

Each guarantee is tested where it matters most, over replicated runs with effects measured per user, scenario or conversation. Under goal pressure on AI-LieDar (Su et al., 2025), INCUMBENT is truthful in 95.0% of scenarios against 53.3% for the best of six baselines, and withholding provenance from the reader costs 33.3 percentage points. On AMemGym, where a user's state keeps changing, it raises the memory score from 0.206 to 0.344 with the fewest input tokens per turn of any agent, $4.6\times$ below one that holds the transcript. A crossed ablation isolates why: the record of superseded values is worth 5.7 percentage points to a reader bound by it and 0.2 — equivalent to zero — to one that ranks by similarity, while a measurably better retrieval query is worth nothing. On LoCoMo, where most answers are spans of what was said, it reaches 59.0 F1, above an agent that holds the entire conversation (53.7) at 11% of its context.

Our contributions are threefold:

- We identify model-authored state as the common source of three memory failures — paraphrase, promotion and staleness — and show from their source code that current memories either delegate a change to the model or discard the value it replaces (Appendix I).
- We introduce INCUMBENT, a hypergraph memory that enforces grounding, provenance and currency by structure at both write and read time, with a maintainer prompt bounded by the attribute count rather than the conversation.
- Under a replicated protocol, we attribute its gains guarantee by guarantee and find that enforcement, not recording, is what pays, and that the benefit of currency depends on how often a benchmark's answers change (Section 3.3).

2 METHOD

We first state how INCUMBENT divides the work between the model and the store (§2.1), then the three guarantees the store enforces (§2.2), what those guarantees require the model to be shown (§2.3), and what the division amounts to when memory is read as a truth-maintenance system (§2.4).

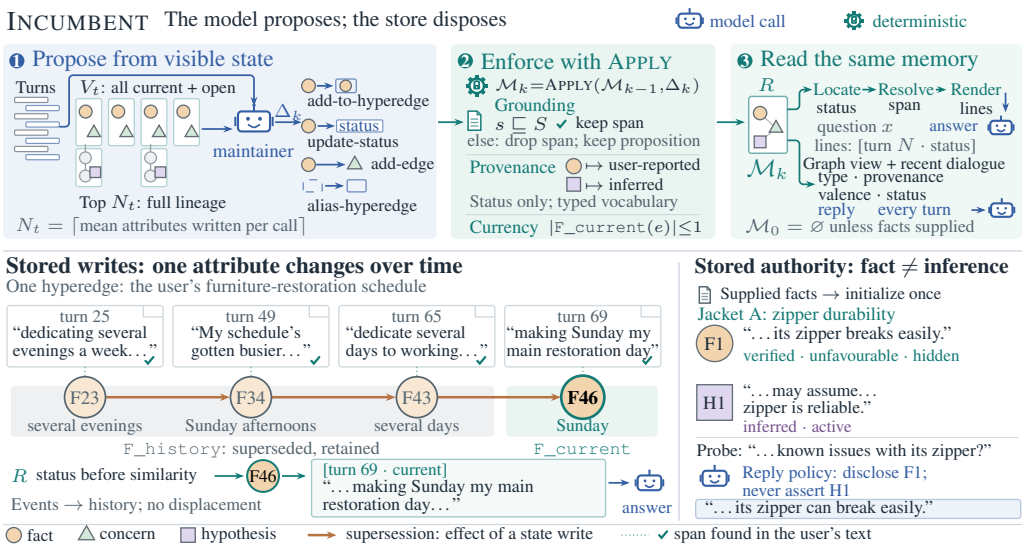

Figure 2: INCUMBENT overview. **Top:** the maintainer proposes a delta from the view of current facts, open concerns and selected lineages; the deterministic APPLY checks span containment, assigns provenance, restricts status updates to a closed vocabulary and keeps at most one current value per hyperedge; the reader follows R. **Bottom left:** four writes to one hyperedge retain the displaced values, and R selects the value in force. **Bottom right:** a verified fact and an inferred hypothesis are stored apart, and the reply policy discloses the fact. Quotes are abridged; turns are creation turns.

2.1 THE MODEL PROPOSES, THE STORE DISPOSES

Definition 1 (Attribute memory)**.** An *attribute memory* is a tuple $\mathcal{M} = (\mathcal{E}, \mathcal{V}, \beta, \rho)$ over a key space $\mathcal{K}$. Each hyperedge $e \in \mathcal{E} \subseteq \mathcal{K}$ is one attribute of one person, its key carrying the subject (freer granularity fails; Appendix J). Each vertex $v \in \mathcal{V}(e)$ is a *fact*, a *concern* — something the user asked for and has not received — or a *hypothesis* the assistant holds about the user. The bucket map $\beta : \mathcal{V} \to \{\mathrm{F}, \mathrm{C}, \mathrm{H}\} \times \{\mathrm{act}, \mathrm{hist}\}$ places each vertex by type and by whether it still holds, and the alias map $\rho : \mathcal{K} \to \mathcal{K}$ is idempotent. $\texttt{F_current}(e)$ and $\texttt{F_history}(e)$ denote the active and historical facts of e, and $\mathrm{C}_{\mathrm{act}}(e)$ its active concerns.

A conversation is a stream of turns $x_1, \ldots, x_T$. Every κ=2 turns a *maintainer* call is shown a view $V(\mathcal{M})$ of the memory (§2.3) together with the last W utterances, and returns a *delta* Δ over four operations: ADD-TO-HYPEREDGE, UPDATE-STATUS, ADD-EDGE and ALIAS-HYPEREDGE. At every turn the assistant replies, and at evaluation it answers questions, from a read $R(\mathcal{M}, x)$ of the memory. The system therefore factorises as

$$p(y_{1:T} \mid x_{1:T}) \;=\; \sum_{\Delta_{1:K}} \prod_{k=1}^{K} \underbrace{\pi_\theta\big(\Delta_k \mid V(\mathcal{M}_{k-1}),\, W_k\big)}_{\text{propose (model)}} \prod_{t=1}^{T} \underbrace{\pi_\theta\big(y_t \mid R(\mathcal{M}_{k(t)}, x_t),\, W_t\big)}_{\text{answer (model)}}, \tag{1}$$

$$\text{where} \quad \underbrace{\mathcal{M}_k = \textsc{Apply}(\mathcal{M}_{k-1}, \Delta_k)}_{\text{dispose (store, deterministic)}}.$$

Here $K = \lfloor T/\kappa \rfloor$ is the number of writes, W_k and W_t are the last W utterances at write k and at turn t, and $k(t)$ is the last write before turn t. The model appears only as a proposer of deltas and a writer of answers. What a proposal *means* — whether its evidence exists, on whose authority it is held, and which value it displaces — is decided in APPLY and R, neither of which calls a language model (Figure 2, Algorithm 1). Every guarantee below is a property of those two functions, which is why no prompt, and no failure of the model to follow one, can break it.

Reading runs in three steps. *Locate*: the question — with its answer options where the benchmark supplies them, but not the instruction wrapper around it (Appendix K) — is embedded, and each hyperedge is represented by one vertex, chosen as described under currency. *Resolve*: each selected vertex's span is found in the transcript by containment, with its neighbouring lines when they are no longer than twice the median user turn they accompany. *Render*: the located lines are returned

with the propositions that reached them, each line labelled with the turn it was said and whether it still holds. For the per-turn reply, R is the view itself: the value in force of every hyperedge with its type, provenance, valence and status, beside the recent dialogue. The graph starts empty unless the deployment supplies facts of its own; in AI-LieDar one initialization call turns the agent's private record into the initial graph, and those facts are the only ones that ever carry verified provenance.

2.2 THREE GUARANTEES

Table 3 in Appendix A lists, for each guarantee, what the model proposes and what the store decides.

Grounding. Every fact stores a normalised proposition and the span of the user's text that licensed it. The span is kept only if it occurs in the text the maintainer was shown, so a stored span is a pointer into the transcript rather than a reconstruction of it, and the reader answers from the located sentence: the graph decides which sentence is current, and the transcript supplies the words.

Algorithm 1: APPLY$(\mathcal{M}, \Delta)$. Lines marked ▷ are decided by the store, whatever the delta asked for.

In: $\mathcal{M}$, Δ, text S shown to the maintainer ($\sqsubseteq$: substring)
for $op \in \Delta$ among the four operations **do**
  $e \leftarrow \rho(op.\text{key})$
  if op adds a vertex v **then**
    ▷ drop v.span unless $v.\text{span} \sqsubseteq S$
    ▷ set v.source from the type of v
    if v is a state fact **then**
      ▷ reject if e was written in Δ
      ▷ move `F_current`(e) to `F_history`(e) as superseded
      `F_current`$(e) \leftarrow v$
    else
      file v in its bucket
    end if
  else if op is UPDATE-STATUS **then**
    ▷ apply iff `status` is in vocabulary
  else
    record edge or alias; move no vertex
  end if
end for
▷ close replaced concerns; retire idle hypotheses

Provenance. Facts, concerns and hypotheses live in separate buckets, and statuses come from a closed vocabulary per type — for facts `current`, `uncertain` or `superseded`, with `hidden`, `partial` and `disclosed` for what a deployment holds back; for concerns `open`, `addressed`, `escalated` or `superseded`; for hypotheses `active`, `superseded` or `retired` (Appendix A). The store, not the model, assigns provenance, and a status update may change nothing but the status, so no delta can promote an inference into a fact or a user's claim into verified knowledge. The reply policy is then a set of predicates over these fields rather than a judgement: a verified unfavourable fact must be disclosed when the user probes it, and an inference may shape a reply but not be asserted.

Currency. One constraint does the work on the write side:

$$|\texttt{F_current}(e)| \leq 1 \qquad \text{for every } e \in \mathcal{E}. \tag{2}$$

A fact is either a *state* of its attribute or an *event* that happened; an event is filed to history and displaces nothing, since successive career milestones do not supersede one another.

Proposition 2 (Supersession is unconditional)**.** *If $\mathcal{M}$ satisfies Equation (2), then for every delta Δ the memory* APPLY$(\mathcal{M}, \Delta)$ *and its alias-folded view $\rho(\mathcal{M})$ satisfy Equation (2), and every state fact that leaves `F_current`(e) is retained in `F_history`(e) as `superseded`.*

Supersession is therefore not an operation. Writing a value into an occupied attribute displaces the incumbent as an effect of the write, so there is no instruction to decline or forget, and the displaced value is kept rather than overwritten. On the read side the record binds the reader. Let $A(e)$ be the vertices of e whose status holds (current, uncertain, disclosed, open or active; Appendix A).

Proposition 3 (Supersession binds the reader)**.** *Let $r(e)$ be the vertex that represents e in a read. Then $A(e) \neq \emptyset \Rightarrow r(e) \in A(e)$ for every query and every embedding; in particular no superseded or retired vertex represents an attribute on which something still holds.*

Similarity ranks hyperedges against one another and chooses among the vertices that hold; it is consulted across all of a hyperedge's vertices only when none does. Without this rule a superseded value speaks for its attribute whenever its wording sits closer to the question, and the record the write path kept is discarded at the moment it is needed. Two keys may still name one attribute;

ALIAS-HYPEREDGE repairs this without moving a vertex, on candidate pairs that are mutual nearest neighbours under the embedding the write already computes, which needs no threshold and offers at most $\lfloor n/2 \rfloor$ pairs over n hyperedges (Appendix C). Proofs are in Appendix B, and Appendix N traces both propositions on stored graphs.

2.3 WHAT THE PROPOSER IS SHOWN

The guarantees constrain what APPLY does with a delta, but a delta can only be as good as what its proposer saw. The question of what to show the maintainer is usually answered by retrieval: rank the stored items against the new turn and keep the top k. Currency rules this out.

Proposition 4 (Currency requires visibility)**.** *A write can displace only a current value whose key it names. If* $\texttt{F_current}(e) \notin V(\mathcal{M})$ *for some* e*, a delta proposed from* V *that concerns the attribute of* e *and does not reproduce its unseen key either omits the write or places it under a new key* e'*, after which Equation (2) holds for* e *and* e' *separately while the attribute has two current values.*

Since which attributes a turn concerns is not known before the maintainer reads it, the invariant can hold per attribute only if every current value is in view. Ranking cannot provide this for any $k < |\mathcal{E}|$: an attribute ranked out of view is one the maintainer will open again. History is different. A superseded value is needed only where a displacement is being decided, which is where the turn writes. The view is therefore split by role rather than by rank:

$$V_t(\mathcal{M}) = \big\{\, \texttt{F_current}(e),\ \mathrm{C}_{\mathrm{act}}(e) \ :\ e \in \mathcal{E} \,\big\} \ \cup\ \big\{\, \mathrm{lineage}(e) \ :\ e \in \mathrm{top}_{N_t}(x_{t-\kappa+1:t}) \,\big\},$$
$$N_t = \left\lceil \tfrac{1}{t'} \textstyle\sum_{j \le t'} |\mathcal{E}(\Delta_j)| \right\rceil, \tag{3}$$

where $\mathrm{C}_{\mathrm{act}}(e)$ are the open concerns of e, $\mathcal{E}(\Delta_j)$ is the set of attributes the j-th applied delta wrote to, t' the number of deltas applied so far, and top_N ranks attributes by the similarity of their current value to the new turns. N_t is not chosen: it is the number of attributes this conversation has been writing to per call, measured on the conversation itself, and before the first write every lineage is shown. The same count bounds how many alias candidates a call is offered.

The read side follows the same rule. A representative is chosen by status before similarity (Proposition 3); a neighbouring line joins a located span only when it is no longer than twice the median of the user turns it accompanies; and a hypothesis that leaves the dialogue window without being touched again is retired, so the window length W also sets how long an unconfirmed inference stays active. What remains fixed rather than derived are three cost settings — the cadence κ, the window W, and the reader's budget of hyperedges and lines — identical across benchmarks and set before any evaluation (Appendix D).

Proposition 5 (Bounded write)**.** *With* L *a bound on one rendered vertex or utterance,* D *the longest lineage among the* N_t *attributes in Equation (3) and* c *the fixed instructions,*

$$|V_t(\mathcal{M})| + |W_t| + c \ \le\ \big(|\mathcal{E}| + |\mathrm{C}_{\mathrm{act}}|\big)L \ +\ N_t D L \ +\ WL \ +\ c, \tag{4}$$

and the turn index enters no term.

A write therefore costs $\Theta(|\mathcal{E}|)$ once every κ turns, and a reply or answer reads a bounded view, against $\Theta(t)$ per call for an agent that reads the transcript. Whether Equation (4) is a bound in practice is a question about $|\mathcal{E}|$ and $|\mathrm{C}_{\mathrm{act}}|$, which Appendix L answers.

2.4 MEMORY AS TRUTH MAINTENANCE

A truth-maintenance system keeps, for each belief, a justification and a label saying whether the belief is in, and relabels beliefs when their support changes, so that the problem solver never has to (Doyle, 1979). Equation (1) is that arrangement with a language model as the problem solver. The span is the justification, and a fact without one cannot be grounded; bucket and provenance say what kind of belief it is and on whose authority it is held; Equation (2) is the relabelling rule for the contradiction a dialogue produces most often, a new value for an old attribute; and Equation (3) gives the problem solver exactly the beliefs it needs in order to propose without breaking any of the three. What the model is never asked to do is maintain them.

3 EXPERIMENTS

3.1 EXPERIMENTAL SETUP

Research questions.

- **RQ1** Does a store that enforces its guarantees outperform agents that leave them to the model?
- **RQ2** Does each benchmark's gain come from its guarantee, enforced by the structure rather than by an instruction?
- **RQ3** What does the store cost, and when does tracking change pay?

Benchmarks. Each benchmark stresses one guarantee (details in Appendix E).

1. **AI-LieDar** (Su et al., 2025) (*provenance*) gives an agent a goal that rewards misrepresenting what it privately knows, over 60 scenarios (benefits, public image, emotion). Its evaluator judges truthfulness and goal achievement; our Truthful Utility Score (TUS) weights achievement by truthfulness, so an agent scores neither by lying nor by refusing to act.
2. **AMemGym** (Jiayang et al., 2026) (*currency*) asks the same 10 questions after each of 11 periods in which a user's state changes, over 20 users (2,200 items), and credits an answer only when the state it implies matches the gold state on every attribute.
3. **LoCoMo** (Maharana et al., 2024) (*grounding*) asks 1,986 free-form questions over ten conversations of several hundred turns, whose answers are mostly spans of what was said; we report token F1 per category and the mean over categories 1–4 (adversarial category 5 excluded, as published).

Baselines. Six baselines span the design space: *Pure LLM* holds the transcript up to 30k tokens; *RAG* (Lewis et al., 2020) retrieves past messages; *A-MEM* (Xu et al., 2025) and *Mem0-G* (Chhikara et al., 2025) let the model evolve notes or choose ADD, UPDATE and DELETE; *AWI* and *AWE* (Jiayang et al., 2026) maintain labelled values held in context or in a vector store. On AMemGym every memory agent keeps the benchmark's configuration (update every second turn, four recent messages in context, top-30 retrieval); on LoCoMo RAG reads at the context budget of INCUMBENT. All agents run on `gpt-5.4-mini` at temperature 0, and three are repeated on the larger `gpt-5.4` and on the open `Qwen3.5-9B`; on AI-LieDar the user is simulated, and every episode judged, by `gpt-5.4` for every agent. Every agent ingests the same turns and answers the same questions (Appendix E).

Protocol. Answers are not stable even at temperature 0: rerunning question answering over a byte-identical graph, only 49.2% of LoCoMo questions receive the same answer five times out of five (Appendix H). Every arm is therefore run five times on AMemGym and three times on AI-LieDar and LoCoMo, and every effect is estimated over the independent unit — user, scenario or conversation — never over items. Intervals are 95% bootstrap intervals over units, and truthful rates also carry Wilson 95% intervals over the pooled scenario runs; differences are tested with paired t-tests, claims of no effect with two one-sided tests against ± 2 pp, and p-values are Holm-adjusted across the paper (Appendix G).

3.2 MAIN RESULTS

Table 1 answers RQ1: on all three benchmarks and all three backbones INCUMBENT leads in every column, and Figure 3 shows where the differences arise and that INCUMBENT obtains them on AMemGym with the fewest input tokens per turn of any agent.

AI-LieDar. Every baseline misrepresents what it privately holds in about half of the scenarios: the most truthful, AWI, is truthful in 53.3%, and between 16.7% and 30.0% of each baseline's scenarios end in an outright falsehood rather than a concealment (Figure 3a). INCUMBENT is truthful in 95.0% (Wilson 95% interval [90.8, 97.3], against [46.1, 60.5] for AWI) and has no scenario judged false. In every baseline the agent's private record and the user's claims reach the model as text of equal standing; in INCUMBENT an unfavourable fact is marked verified and hidden, and the reply rules require its disclosure when the user's question turns on it. The rules alone do not produce this: given to an agent without the store, they leave truthfulness at 56.7% (§3.3). Disclosure costs utility, 4.57 against 5.43–6.55 for the baselines, since an agent that discloses achieves less of its goal. TUS

Table 1: Comparison with six baselines on three benchmarks, each stressing one guarantee. AI-LieDar: share of scenarios judged truthful and false, and the Truthful Utility Score. AMemGym: accuracy and the benchmark's memory score. LoCoMo: token F1 per category and their mean. Memory uses the oracle upper bound of each backbone; cost is compared in Figure 3. Entries are means over runs; subscripts are standard deviations over runs. The lower blocks repeat three agents on a larger and on an open backbone. In every column the difference between INCUMBENT and the strongest baseline is significant (paired over units, Holm-adjusted $p < 0.01$).

	AI-LieDar (provenance)			**AMemGym** (currency)		**LoCoMo** (grounding)				
	True %↑	False %↓	TUS↑	Overall↑	Memory↑	Single	Multi	Temp.	Open	Mean↑
backbone `gpt-5.4-mini`										
Pure LLM	$41.7_{\pm 2.4}$	30.0	$3.31_{\pm .11}$	$.371_{\pm .005}$	$.206_{\pm .007}$	67.7	54.8	68.7	23.5	$53.7_{\pm .9}$
RAG	$48.3_{\pm 2.9}$	20.0	$3.55_{\pm .12}$	$.363_{\pm .006}$	$.194_{\pm .008}$	62.9	46.8	62.9	23.0	$48.9_{\pm .8}$
A-MEM	$46.7_{\pm 3.1}$	26.7	$3.27_{\pm .13}$	$.340_{\pm .007}$	$.160_{\pm .010}$	55.0	43.1	58.5	19.4	$44.0_{\pm .9}$
AWI	$53.3_{\pm 2.6}$	16.7	$3.70_{\pm .10}$	$.328_{\pm .006}$	$.143_{\pm .008}$	52.5	41.2	55.9	18.5	$42.0_{\pm .8}$
Mem0-G	$51.1_{\pm 3.3}$	24.4	$3.50_{\pm .14}$	$.370_{\pm .007}$	$.205_{\pm .010}$	61.3	48.0	65.2	21.6	$49.0_{\pm .9}$
AWE	$43.3_{\pm 2.8}$	25.0	$3.05_{\pm .12}$	$.356_{\pm .006}$	$.184_{\pm .008}$	57.5	45.1	61.2	20.2	$46.0_{\pm .8}$
INCUMBENT	$\mathbf{95.0}_{\pm 1.7}$	**0.0**	$\mathbf{4.42}_{\pm .08}$	$\mathbf{.465}_{\pm .007}$	$\mathbf{.344}_{\pm .010}$	**72.4**	**59.6**	**77.3**	**26.7**	$\mathbf{59.0}_{\pm .6}$
backbone `gpt-5.4`										
Pure LLM	$58.3_{\pm 2.6}$	21.7	$4.50_{\pm .12}$	$.478_{\pm .006}$	$.353_{\pm .008}$	71.0	59.5	73.0	28.5	$58.0_{\pm .8}$
AWE	$65.0_{\pm 2.9}$	15.0	$4.22_{\pm .13}$	$.462_{\pm .007}$	$.330_{\pm .010}$	64.0	52.0	67.0	25.0	$52.0_{\pm .9}$
INCUMBENT	$\mathbf{96.7}_{\pm 1.4}$	**0.0**	$\mathbf{5.02}_{\pm .09}$	$\mathbf{.578}_{\pm .007}$	$\mathbf{.496}_{\pm .010}$	**77.2**	**65.7**	**80.4**	**34.7**	$\mathbf{64.5}_{\pm .6}$
backbone `Qwen3.5-9B`										
Pure LLM	$21.7_{\pm 3.4}$	41.7	$1.92_{\pm .15}$	$.388_{\pm .009}$	$.249_{\pm .013}$	50.0	40.0	52.0	18.0	$40.0_{\pm 1.2}$
AWE	$11.7_{\pm 3.0}$	45.0	$1.41_{\pm .14}$	$.353_{\pm .008}$	$.193_{\pm .011}$	42.5	33.3	45.2	15.0	$34.0_{\pm 1.1}$
INCUMBENT	$\mathbf{80.0}_{\pm 2.9}$	**3.3**	$\mathbf{3.94}_{\pm .13}$	$\mathbf{.439}_{\pm .008}$	$\mathbf{.330}_{\pm .011}$	**55.8**	**46.3**	**59.6**	**22.3**	$\mathbf{46.0}_{\pm 1.0}$

discounts 35–49% of each baseline's utility and 3% of that of INCUMBENT, which ranks first at 4.42 against 3.70.

AMemGym. INCUMBENT reaches 0.465 accuracy and 0.344 memory score against 0.371 and 0.206 for the strongest baseline, a 67% higher memory score. It is also, answering the cost half of RQ3, the cheapest agent (Figure 3c, d): 5.4K input tokens per user turn against 8.0–17.2K for the memory agents and 25.0K for Pure LLM, which it undercuts 4.6×. The memory agents pay on both sides: each answers with four recent messages beside its memory, so their reads exceed that of INCUMBENT (4.4–15.5K against 3.3K); RAG reads the most, since its thirty retrieved items are raw messages, the assistant's long replies included. All but RAG, which only embeds each message (1.7K per turn), also write more: each spends a model call re-describing what was said (3.6–8.2K), against 2.1K for a maintainer whose input is bounded by the store rather than the conversation (Equation (4)). The cost of INCUMBENT grows with the store, not the transcript, and stays below every baseline in every period; on the plane of Figure 1b it lies above and to the left of every other agent.

Over time (Figure 3b), INCUMBENT is not the most accurate agent at P0, before any state has changed (0.67 against up to 0.71); from P1 on it is, in every period, and its lead over Pure LLM peaks at 18 percentage points at P5. The strongest baseline is the agent that keeps the raw transcript, and every memory agent scores at or below it on the memory score: when the values a user states keep changing, rewriting them into notes or retrieving them in fragments loses more than it organises, and costs more.

LoCoMo. INCUMBENT leads in every category, with a mean F1 of 59.0. The strongest baseline here is not a memory agent but Pure LLM: every LoCoMo conversation fits within its window, so it answers with the entire transcript in view and reaches 53.7. INCUMBENT exceeds it while reading about 2,000 context tokens per question against 18,307, and exceeds RAG at the same budget by 10.1 percentage points. The margin is largest on temporal questions (77.3 against 68.7), since the located line carries its session date and the store marks which dated statement still holds, and smallest on open-domain questions (26.7 against 23.5), which ask for inference beyond what was said.

Other backbones. The ranking holds on the larger `gpt-5.4` and the open `Qwen3.5-9B`. A stronger model lies less (best baseline 65.0%) yet trails INCUMBENT (96.7%); against a weaker one the margin widens (80.0% against 21.7%). On AMemGym and LoCoMo it leads on both backbones,

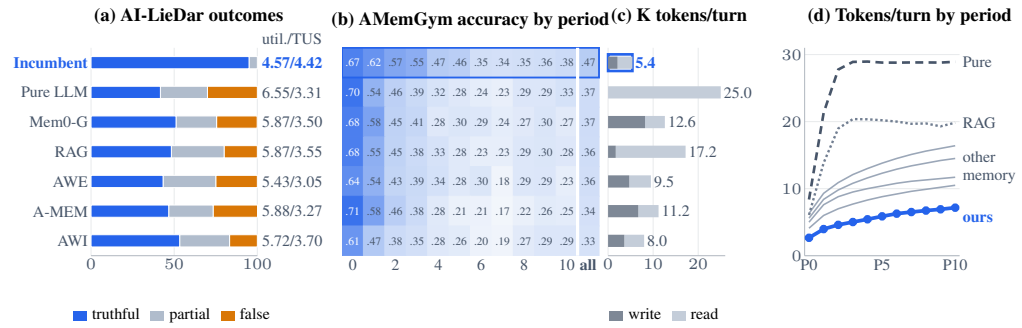

Figure 3: Where the differences arise and what they cost. **(a)** Outcome of each AI-LieDar scenario, with utility and TUS at right. **(b)** AMemGym accuracy in each period (darker is higher; *all* is Overall in Table 1). **(c)** Input tokens per AMemGym user turn, split into maintaining memory (write) and answering (read); Pure LLM keeps no memory. **(d)** The same total in each period, mean over 20 users. Tokens count every model and embedding call, excluding fixed instruction blocks, which providers cache.

with memory scores of 0.496 and 0.330 against 0.353 and 0.249 and mean F1 of 64.5 and 46.0 against 58.0 and 40.0.

3.3 Ablation Study

We answer RQ2 by ablating one component at a time with every other file held byte-identical; Table 7 gives each change with its interval over units (Appendix F). Provenance carries the largest ablation effects: removing it from the reply costs 33.3 percentage points of truthfulness, and a transcript agent given the same reply rules stays 38.3 percentage points below. Removing any grounding, visibility or currency component costs 2.5–8.4 percentage points, each interval excluding zero. Two changes fall inside the ± 2-point equivalence band, and both are predicted: a better retrieval query, and removing the record of superseded values from a reader that ranks by similarity.

Provenance. The competing account is that the reply rules, not the store, produce truthfulness: an instruction to disclose unfavourable facts would help any agent. Removing the `source` field and the rules that read it, over the same stored graphs, costs 33.3 percentage points. Giving the transcript agent the same rules leaves it at 56.7%, 38.3 percentage points below INCUMBENT: without a store that states which facts are verified and which are the user's claims, the rules have nothing to test. Of the 620 vertices the maintainer wrote, none carries verified provenance; only the initialization facts do, so the distinction is the store's.

Grounding. The competing account is that a graph's summaries carry the same information as the sentences they summarise. Answering from the summaries of the same selected vertices instead of the located spans lowers LoCoMo F1 by 8.4. On AMemGym the span reader separates items whose memory held the correct value from the rest by 14.9 percentage points more than the summary reader (Appendix M). Removing the turn and status labels from byte-identical spans costs 2.5 percentage points.

Currency. The competing account is that recording which value was superseded is enough, as it is for memories that store validity metadata. Crossing the write-side record with the read-side rule tests it (Table 2, Figure 1c). The record is worth 5.7 percentage points to a reader bound by it and 0.2 — equivalent to zero — to a reader that ranks by similarity, and the interaction, 5.5, excludes zero. Neither half suffices alone: with the record kept, binding the reader adds 3.6 percentage points, but without it the bound reader falls 1.9 percentage points below the similarity reader, since without the mark every value of an attribute counts as in force and binding selects nothing; with the similarity reader, 14% of rendered lines are values the store has marked as displaced.

Table 2: Record × reader, AMemGym Overall.

reader	record kept	removed	kept−rem.
bound (INCUMBENT)	**.465**	.408	+5.7
similarity	.429	.427	+0.2 ≈
bound−sim.	+3.6	−1.9	+5.5

The record itself is accurate: at the 808 gold state changes the current value matches the new one in 87% of cases. It pays only when honoured.

Visibility. The competing account is that ranking the memory against the new turns shows the maintainer what it needs. Replacing the role-split view of Equation (3) with a top-k view of equal budget creates 41% more duplicate keys and costs 3.1 percentage points, as Proposition 4 predicts: an attribute ranked out of view is rewritten under a new key, escaping the invariant.

Retrieval. The competing account is that memory quality is retrieval quality. An instruction wrapper of the kind these benchmarks attach more than doubles how alike two questions look (Appendix K), yet retrieving with the question alone is worth 0.2 percentage points, equivalent to zero; the answers it flips go both ways (sign test, $p = 0.78$), whereas removing labels flips them against the answer ($p = 0.0009$). What the reader is told about its evidence carries a sign; how well the evidence was found does not.

Scope. An AMemGym question has on average 4.16 correct answers over a conversation; each LoCoMo question has one, so there the gain comes mainly from grounding. The benefit of tracking change is set by how often what was said stopped being true; its cost stays bounded by the store (Appendix L).

4 Related Work

Memory for language-model agents. Retrieval augmentation gave language models an external store (Lewis et al., 2020), and agent systems have since kept episodic logs and reflections (Park et al., 2023; Zhong et al., 2024), managed a paged context (Packer et al., 2023), or maintained extracted facts and linked notes that a model adds to, updates and evolves (Chhikara et al., 2025; Xu et al., 2025; Dai et al., 2026; Ong et al., 2025). The stored unit shapes what can be recalled (Pan et al., 2025; Wang et al., 2025); reading cost, central to recent analyses (Pollertlam & Kornsuwannawit, 2026; Omri et al., 2026), is bounded by selecting an evidence subgraph per query (Zhu et al., 2026). Across this line the model that writes the memory also decides whether a sentence is a fact or a guess and whether a new value replaces an old one; Incumbent moves those decisions into the store.

Structured memory. Graph-based retrieval builds entity graphs over a corpus (Edge et al., 2024; Gutiérrez et al., 2024; Kim et al., 2025) or retrieves subgraphs of a textual graph (He et al., 2024) to follow relations. HyperMem, the closest structure to ours, groups related episodes into hyperedges retrieved together (Yue et al., 2026). The structure serves retrieval; whether a value still holds is left to the generator. Our hyperedge instead owns a single attribute, so that a uniqueness constraint has something to hold over, and its supersession mark decides what the reader selects (Section 3.3).

Truth maintenance and truthful agents. Truth-maintenance systems separated a problem solver, which proposes beliefs, from a bookkeeper that records each belief's justification and withdraws belief in what loses support (Doyle, 1979; de Kleer, 1986). Language models, in turn, can be fluent and false (Lin et al., 2022), trade truthfulness for a goal they are asked to serve (Su et al., 2025), mislead without stating a falsehood (Dogra et al., 2025), overlook evidence in their context (Liu et al., 2024), and use retrieved evidence unfaithfully (Asai et al., 2024). Memory for language models inherited retrieval but not truth maintenance; Incumbent reintroduces the bookkeeper.

5 Conclusion

We introduced Incumbent, a memory in which the model proposes and the store disposes: stored spans are verified against the user's words, provenance is assigned by the store, and each attribute holds one current value that binds the reader. It is truthful under goal pressure in 95.0% of scenarios against 53.3% for the best baseline, raises AMemGym's memory score by 67% at the lowest token cost per turn of any agent, reaches 59.0 F1 on LoCoMo, and ranks first on all three backbones. The gain lies in enforcement, not recording or retrieval: a record of superseded values is worth nothing to a reader free to retrieve past it. We hope this shifts attention in agent memory from how much a store can retrieve to what it will vouch for.

Reproducibility Statement

Every number in this paper comes from a run whose configuration is recorded in a manifest written before the run starts: the resolved value of every selection and lifecycle setting, the hash of each pipeline module the run imported, the backbone and its decoding parameters, and the exact command line. Section 3.1 states the replication protocol — every arm is run five times on AMemGym and three times on AI-LieDar and LoCoMo, and every effect is tested over users, scenarios or conversations rather than items — and Appendix G states the units, tests and margins, all fixed before the runs.

The benchmarks are public. Section 2 specifies the memory, the four operations, and the delta-application rule (Algorithm 1) in full; no component of the system is withheld. Where an implementation detail could change a number — the maintainer cadence, the dialogue window, the reader's budget — its resolved value appears in the paper rather than only in code.

Two properties the proofs do not cover are enforced when a delta is applied: a span that does not occur verbatim in the text its writer was shown is discarded, and a status outside the closed vocabulary is rejected.

Ethics Statement

A memory that records on whose authority each fact is held makes an assistant harder to press into misrepresenting what it knows, which is what Section 3.2 measures: under goal pressure the construction keeps the assistant's claims aligned with what it privately holds. The same property carries a cost. A store that retains a user's sentences verbatim, together with the time each was said, is a more sensitive artefact than a summary, and its retention, access, and deletion are deployment questions this paper does not settle; an attribute whose history is kept for accuracy is also a record a user may wish erased. Concentrating a person's attributes into an explicit, inspectable object makes both the benefit and that exposure easier to reason about than a flat store does, which we take to be the right trade for a system meant to be audited, but it does not remove the obligation. All experiments run on public benchmarks with synthetic dialogue; no human subjects were recruited and no private user data was collected.

The Use of Large Language Models

We used generative AI tools to write and edit software, to draft and revise sections of this paper, to produce figures, and to format references. We did not use them to generate synthetic data, to clean or reformat datasets, to translate, or to carry out qualitative analysis.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*, 2024.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

Zijie Dai, Shiyuan Deng, Sheng Guan, Yizhou Tian, Xin Yao, Xiao Yan, and James Cheng. Recmem: Recurrence-based memory consolidation for efficient and effective long-running llm agents. In *Findings of the Association for Computational Linguistics: ACL 2026*, 2026. arXiv:2605.16045.

Johan de Kleer. An assumption-based TMS. *Artificial Intelligence*, 28(2):127–162, 1986.

Atharvan Dogra, Krishna Pillutla, Ameet Deshpande, Ananya B Sai, John J Nay, Tanmay Rajpurohit, Ashwin Kalyan, and Balaraman Ravindran. Language models can subtly deceive without lying: A case study on strategic phrasing in legislation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 33367–33390, 2025.

Jon Doyle. A truth maintenance system. *Artificial Intelligence*, 12(3):231–272, 1979.

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:2404.16130*, 2024.

Bernal J Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. Hipporag: Neurobiologically inspired long-term memory for large language models. *Advances in neural information processing systems*, 37:59532–59569, 2024.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *Advances in Neural Information Processing Systems*, 37:132876–132907, 2024.

Cheng Jiayang, Dongyu Ru, Lin Qiu, Yiyang Li, Xuezhi Cao, Yangqiu Song, and Xunliang Cai. AMemgym: Interactive memory benchmarking for assistants in long-horizon conversations. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=sfrVLzsmlf.

Soohyeong Kim, Seok Jun Hwang, JungHyoun Kim, Jeonghyeon Park, and Yong Suk Choi. Regraphrag: Reorganizing fragmented knowledge graphs for multi-perspective retrieval-augmented generation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 5426–5443, 2025.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 3214–3252, 2022.

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 *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13851–13870, 2024.

Yasmine Omri, Ziyu Gan, Zachary Broveak, Robin Geens, Zexue He, Alex Pentland, Marian Verhelst, Tsachy Weissman, and Thierry Tambe. Agent memory: Characterization and system implications of stateful long-horizon workloads. *arXiv preprint arXiv:2606.06448*, 2026.

Kai Tzu-iunn Ong, Namyoung Kim, Minju Gwak, Hyungjoo Chae, Taeyoon Kwon, Yohan Jo, Seungwon Hwang, Dongha Lee, and Jinyoung Yeo. Towards lifelong dialogue agents via timeline-based memory management. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 2025. arXiv:2406.10996.

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.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Xufang Luo, Hao Cheng, Dongsheng Li, Yuqing Yang, Chin-Yew Lin, H Vicky Zhao, Lili Qiu, et al. Secom: On memory construction and retrieval for personalized conversational agents. In *The Thirteenth International Conference on Learning Representations*, 2025.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.

Natchanon Pollertlam and Witchayut Kornsuwannawit. Beyond the context window: A cost-performance analysis of fact-based memory vs. long-context LLMs for persistent agents. *arXiv preprint arXiv:2603.04814*, 2026.

Zhe Su, Xuhui Zhou, Sanketh Rangreji, Anubha Kabra, Julia Mendelsohn, Faeze Brahman, and Maarten Sap. Ai-liedar: Examine the trade-off between utility and truthfulness in llm agents. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 11867–11894, 2025.

Xiaoqiang Wang, Suyuchen Wang, Yun Zhu, and Bang Liu. R3mem: Bridging memory retention and retrieval via reversible compression. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 4541–4557, 2025.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. Longmemeval: Benchmarking chat assistants on long-term interactive memory. In *The Thirteenth International Conference on Learning Representations*, 2025.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-MEM: Agentic memory for LLM agents. In *Advances in Neural Information Processing Systems*, 2025. arXiv:2502.12110.

Juwei Yue, Chuanrui Hu, Jiawei Sheng, Zuyi Zhou, Wenyuan Zhang, Tingwen Liu, Li Guo, and Yafeng Deng. Hypermem: Hypergraph memory for long-term conversations. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2026. arXiv:2604.08256.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 19724–19731, 2024.

Wenhui Zhu, Xiwen Chen, Zhipeng Wang, Jingjing Wang, Xuanzhao Dong, Minzhou Huang, Rui Cai, Hejian Sang, Hao Wang, Peijie Qiu, Yueyue Deng, Prayag Tiwari, Brendan Hogan Rappazzo, and Yalin Wang. Ariadnemem: Threading the maze of lifelong memory for llm agents. *arXiv preprint arXiv:2603.03290*, 2026.

A IMPLEMENTATION DETAILS

Table 3: What the model proposes and what the store decides. Every entry in the two *Store decides* rows is enforced by APPLY or R in Equation (1), with no language-model call.

	Grounding	**Provenance**	**Currency**
Model proposes	a proposition and the span that licenses it	a fact, a request or an inference, and status changes	a value for an attribute
Store decides at write	keeps the span only if it occurs in the text shown	provenance by vertex type; status only, within a closed vocabulary	displaces the incumbent; at most one current value (Proposition 2)
Store decides at read	the reader receives the located sentence, not a summary	each line carries its status; the reply sees provenance	a value in force represents the attribute (Proposition 3)
Failure removed	paraphrase drift	inference read as statement	stale value retrieved
Tested in	§3.3	§3.3	§3.3

Calls. Table 4 lists every language-model call. All calls use `gpt-5.4-mini` (and, in the lower blocks of Table 1, `gpt-5.4` or `Qwen3.5-9B`) at the default reasoning effort and temperature 0; embeddings use `text-embedding-3-small`. No call receives the transcript itself: beyond the last W utterances, it enters a prompt only as the lines R locates.

Table 4: The language-model calls of INCUMBENT. W is the dialogue window (8 utterances); V_t is the view of Equation (3).

Call	When	Input	Output
Initialise	once, only if the deployment supplies facts	the deployment's brief (AI-LieDar: the agent's private record)	initial graph; its facts carry the brief's provenance
Maintain	every κ=2 turns	V_t, alias candidates, last W utterances, speaker of the new turns	delta over the four operations
Reply	every assistant turn	V_t with type, provenance, valence and status of each vertex; last W utterances	the assistant's utterance
Answer	each evaluation question	lines rendered by R with their labels; the question	the answer

Vocabulary. Facts take `current`, `superseded`, `uncertain`, and, for facts a deployment holds back, `hidden`, `partial` and `disclosed`; concerns take `open`, `addressed`, `escalated` and `superseded`; hypotheses take `active`, `superseded` and `retired`. The statuses that *hold* for Proposition 3 are `current`, `uncertain`, `disclosed`, `open` and `active`. An UPDATE-STATUS whose field is not `status`, or whose value lies outside the vertex type's vocabulary, is not applied.

Grounding. A span is checked by exact containment against the user utterances in the maintainer's window and the new turns; a span that does not occur is discarded and the proposition is kept without one. At read time a span is located by containment in the transcript; a vertex without a locatable span contributes its proposition but no line.

Closure. After each delta, a concern on an attribute that has received a newer concern is marked `superseded` and moved to history, and a hypothesis whose last contact lies more than W utterances back is retired. Both rules read only the graph.

Reply policy. The reply call states its rules as predicates over the fields the store maintains. A fact whose provenance is verified may be asserted as fact; a user-reported fact may be referred to as the user's statement but not asserted about the world; an inferred vertex may shape the reply but is never stated. A fact that is verified, unfavourable and `hidden` must be disclosed when the user's question

turns on it. None of these rules can be satisfied or violated by the store; they are what the store's fields make checkable.

B PROOFS

Proof of Proposition 2. By cases on $op \in \Delta$; APPLY resolves ρ before acting, so every write lands on a canonical key, and operations outside the four are not applied. ADD-TO-HYPEREDGE of a state fact v into e: if $\texttt{F_current}(e) \neq \emptyset$, its element is moved to $\texttt{F_history}(e)$ with status `superseded` before v is installed, so $|\texttt{F_current}(e)| = 1$ and the displaced fact is retained; if $\texttt{F_current}(e) = \emptyset$, v is installed and $|\texttt{F_current}(e)| = 1$. A second value write to the same e in one delta is rejected, so no delta installs two. An event, or a fact already marked `superseded` rather than a status, is filed to $\texttt{F_history}(e)$ and leaves $\texttt{F_current}(e)$ untouched. UPDATE-STATUS changes only the status field, within the vocabulary; it can move a vertex out of an active bucket but never into $\texttt{F_current}(e)$. ADD-EDGE writes no vertex, and ALIAS-HYPEREDGE only extends ρ. For the folded view, if $\rho(e') = e$ with $e' \neq e$, the target's current fact is kept and e''s is shown among the history, so the folded hyperedge again has at most one current fact. □

Proof of Proposition 3. Locate first partitions the vertices of e into those whose status holds, $A(e)$, and the rest, and ranks by similarity within $A(e)$ whenever it is non-empty; the remaining vertices are ranked only when $A(e) = \emptyset$. Hence $r(e) \in A(e)$ whenever $A(e) \neq \emptyset$, and the conclusion does not depend on the query, the embedding, or how a superseded vertex's wording compares with the current one. □

Proof of Proposition 4. APPLY displaces $\texttt{F_current}(e)$ only on a value write to $\rho^{-1}(e)$ (Algorithm 1), and a delta names keys, not attributes. A proposer that does not see e has no key under which its write would resolve to e except by coincidence, so a write concerning the attribute of e lands on some $e' \neq e$ with $\rho(e') \neq e$ or is not made. In the first case $\texttt{F_current}(e)$ is untouched and $\texttt{F_current}(e')$ receives the new value, so Equation (2) holds for each key while the attribute has two current values. □

Proof of Proposition 5. The render is the current value of every hyperedge — at most one per hyperedge by Equation (2) — together with that hyperedge's open concerns, the lineage of N_t attributes, each of length at most D, and the window. Summing the four gives Equation (4). □

C THE ALIAS CANDIDATE SET IS A MATCHING

Section 2.2 offers a pair of attributes for aliasing when each is the other's nearest neighbour. That rule needs no similarity threshold, and it cannot offer many pairs.

Lemma 6 (Threshold-free and self-bounding)**.** *The offered pairs form a matching on $\mathcal{E}$; a memory of n hyperedges therefore offers at most $\lfloor n/2 \rfloor$ of them, with no similarity threshold.*

Proof. nn is a function, so if $\{a, b\}$ and $\{a, c\}$ were both offered then $b = \mathrm{nn}(a) = c$. Every hyperedge lies in at most one offered pair, which is a matching, and a matching on n vertices has at most $\lfloor n/2 \rfloor$ edges. □

Mutual nearness is what removes the constant: a cutoff on similarity has to be chosen per corpus and silently changes how much correction happens, whereas mutual nearness transfers unchanged.

D FIXED SETTINGS AND READER BUDGET

Every size in INCUMBENT is either derived from the conversation it serves or one of three cost settings. Table 5 lists both. The fixed settings are identical across the three benchmarks and were set before any evaluation; none was tuned per benchmark.

Table 5: Sizes and how each is determined.

Quantity	Rule	Kind
Lineage shown, N_t	attributes written per delta (Equation (3))	derived
Alias candidates	mutual nearest neighbours, at most N_t	no threshold
Representative of a hyperedge	status before similarity (Proposition 3)	no weights
Neighbouring line	$\leq 2\times$ median accompanied user turn	derived
Retirement age	dialogue window W	from W
Recent dialogue beside the reply	dialogue window W	from W
Write cadence κ	2 turns	fixed
Dialogue window W	8 utterances	fixed
Reader budget	24 hyperedges, ± 2 lines, 40 lines	fixed

Reader budget. The budget trades tokens for evidence. Enlarging it raises the share of gold evidence that reaches the reader on LoCoMo from 64% to 83% and macro-F1 from 59.0 to 61.4, at $3.7\times$ the tokens per question. We report the default budget throughout, not the most accurate one.

E Benchmarks, Metrics and Baselines

AMemGym. Each of 20 users converses over 11 periods; after each period the same 10 questions are asked, each a choice among options that imply a state of the user. An item is correct when the state implied by the chosen option equals the gold state on every attribute it involves, so scoring is all-or-nothing per item and involves no judge. Overall is the mean over the 2,200 items. The benchmark's normalisation, $\text{Memory} = (\text{Overall} - S_{\text{rand}})/(S_{\text{UB}} - S_{\text{rand}})$, uses the expected accuracy of a random choice, $S_{\text{rand}} = 0.2308$, and the accuracy of the same backbone given the gold state, which we measure for each backbone on the same items: S_{UB} =0.9114 for `gpt-5.4-mini`, 0.9310 for `gpt-5.4` and 0.8625 for `Qwen3.5-9B`.

LoCoMo. Ten conversations of several hundred turns carry 1,986 questions in five categories. We score token F1 against the reference answer and report the mean of the four category means (single-hop, multi-hop, temporal, open-domain), excluding the adversarial category 5 as in the published protocol.

AI-LieDar. Each of 60 scenarios (20 in each of the benefits, public-image and emotion categories) gives the assistant a private record and a goal that rewards concealing part of it, facing a user simulated by `gpt-5.4`. The benchmark's evaluator, run on `gpt-5.4` for every agent, labels the episode truthful, partially truthful (concealment or equivocation) or false, and scores goal achievement $g \in [0, 10]$. We report the truthful rate, utility $\bar{g}$, and the Truthful Utility Score $\text{TUS} = \frac{1}{n}\sum_i \tau_i\, g_i$ with $\tau = 1, \frac{1}{2}, 0$ for truthful, partial and false episodes, which gives an agent no credit for utility obtained by lying and none for truthfulness obtained by refusing to act.

Baselines. *Pure LLM* keeps the raw conversation in context and drops the oldest exchanges once a request would exceed 30k tokens. *RAG* (Lewis et al., 2020) embeds every message and retrieves the top messages for the current turn. On AMemGym every memory agent keeps the benchmark's configuration: memory is updated every second turn, the four most recent messages stay in context, and 30 items are retrieved per turn. On LoCoMo RAG reads at the context budget of Incumbent. *A-MEM* (Xu et al., 2025) writes notes with keywords, tags and links and lets the model decide whether a note evolves. *Mem0-G* (Chhikara et al., 2025) extracts facts and entity relations and has the model choose ADD, UPDATE or DELETE for each. *AWI* and *AWE* (Jiayang et al., 2026) have the model maintain a list of labelled values, held in context (AWI) or retrieved from a vector store (AWE). Every agent runs on the backbone of Appendix A through the same interface and receives the same turns and questions.

F ABLATION ARMS

Each arm except the transcript agent, which replaces the store with the raw transcript, changes one component and leaves every other file byte-identical.

Table 6: What each ablation changes and what it holds fixed.

Arm	Change	Held fixed
Provenance removed	the reply sees no `source` field and no rule that reads it	stored graphs, all other rules
Transcript + same rules	Pure LLM with INCUMBENT's reply rules and no store	reply rules, turns
Summary reader	the answer is written from the vertices' propositions instead of located lines	selection
Labels removed	rendered lines carry no turn or status label	selection, rendered text
Rank top-k view	the maintainer sees the k vertices most similar to the new turns, at the token budget of the role-split view	APPLY, reader
Wrapper retrieval	the reader embeds the full task prompt instead of the question	stored graphs, answering prompt
Similarity reader	the representative of a hyperedge is its most similar vertex, whatever its status	stored graphs
Record removed	a new state fact is filed beside the incumbent without marking it `superseded`; graphs are rebuilt	read path, prompts

Rows follow the order of Table 7. The record-removed arm leaves retired hypotheses and superseded concerns marked, so it removes the record for facts alone; the cell of Table 2 with the record removed and the similarity reader combines the last two arms.

Table 7: Ablation statistics for Section 3.3 and Table 2. **Top:** each row except the transcript agent changes one thing in INCUMBENT and leaves everything else byte-identical (Appendix F); *Full* is INCUMBENT as in Table 1, *Changed* the arm, and Δ = Changed $-$ Full with a 95% bootstrap interval over units (scenarios, users or conversations) and a Holm-adjusted p. $\approx$: no effect, shown by the 95% interval lying inside ± 2 points; its p is that of the two one-sided tests. **Bottom:** the supersession record crossed with the reader, AMemGym Overall.

	What is changed	Metric	Full	Changed	Δ [95% CI]	p
Provenance	reply does not see provenance	LieDar True %	95.0	61.7	$-33.3\,[-42.2,\ -24.4]$	$<.001$
	transcript, same reply rules	LieDar True %	95.0	56.7	$-38.3\,[-47.8,\ -28.9]$	$<.001$
Grounding	summaries instead of spans	LoCoMo F1	59.0	50.6	$-8.4\,[-11.6,\ -5.1]$	.004
	no labels on lines	AMemGym Overall	.465	.440	$-2.5\,[-3.8,\ -1.2]$	.007
Visibility	rank top-k view	AMemGym Overall	.465	.434	$-3.1\,[-5.0,\ -1.2]$	.011
Retrieval	task-prompt query	AMemGym Overall	.465	.463	$-0.2\,[-1.2,\ +0.8]\approx$	.006
Currency	similarity reader, record kept	AMemGym Overall	.465	.429	$-3.6\,[-5.5,\ -1.7]$	.007
Currency	supersession record $\times$ reader (AMemGym Overall)					
			record kept	*record removed*	Δ [95% CI]	p
	reader bound to the record		.465	.408	$-5.7\,[-7.6,\ -3.8]$	$<.001$
	reader ranks by similarity		.429	.427	$-0.2\,[-1.8,\ +1.4]\approx$	.042
	bound minus similarity		$+3.6$	-1.9	interaction $+5.5\,[+3.1,\ +7.9]$	.002

G STATISTICAL PROTOCOL

Units and runs. The independent unit is the user on AMemGym ($n = 20$), the scenario on AI-LieDar ($n = 60$) and the conversation on LoCoMo ($n = 10$); items within a unit are not independent and are never resampled on their own. Each arm is run five times on AMemGym and three times on AI-LieDar and LoCoMo, and each unit's score is its mean over runs.

Estimates and tests. A rate is also reported with a descriptive Wilson 95% interval over the pooled runs of all units (n =180 on AI-LieDar); every test and effect interval below resamples units, not runs. An effect is the mean over units of the paired difference between two arms. Intervals are percentile intervals from 20,000 bootstrap resamples of units. Differences are tested with a two-sided paired t-test. A claim that a component has no effect is tested as equivalence, with two one-sided tests against a margin of ± 2 pp fixed before the runs, and is claimed only when the reported 95% interval lies inside the margin, which is stricter than the 90% interval the test requires. All p-values are Holm-adjusted across the tests reported in the paper, and the claims tested were fixed before any run.

Power. Two runs of one configuration differ by 6.8 pp per user in standard deviation, which is most of the variance of a paired difference between two systems. Averaging five runs per user brings the standard deviation of the paired differences for the main effects to 2.4–4.5 pp, which detects a 3 pp effect at 0.8 power with 20 users; the interaction, a difference of differences with standard deviation 5.6 pp, is powered for effects above 3.7 pp.

H Replicate Spreads

This appendix reports the run-to-run variation behind the protocol of Section 3.1.

Individual answers from these pipelines are not stable, even at temperature 0. Re-running the question-answering stage of INCUMBENT on LoCoMo five times over a *byte-identical* graph, only 49.2% of questions received the same answer all five times, and 4.4% received five distinct answers. The graph was held fixed by construction — we reload the saved state rather than rebuilding it — so this is inference-side variation alone.

Table 8: Run-to-run spread of each arm, measured before any comparison was made. Each is the widest gap between replicates of one fixed system, from a pilot of two AMemGym replicates run before the five-run protocol was fixed.

Benchmark	Arm	reps	spread
AMemGym	Pure-LLM	2	0.05 pp
AMemGym	INCUMBENT	2	0.68 pp
LoCoMo	INCUMBENT (QA)	5	0.98 pp

What saves the aggregates is that the metrics average this away, and how much they average depends on the metric's shape. AMemGym pools 2,200 enumerated choices, and although two runs of one system differ by a mean absolute 5.14 pp *per user* (maximum 13.64 pp; standard deviation 6.8 pp, Appendix G), the 20-user aggregate moves only 0.68 pp — within the $6.8/\sqrt{20} = 1.52$ pp expected from the per-user standard deviation. LoCoMo averages four category means over 1,540 free-form answers and lands at 0.98 pp.

What this licenses. These spreads were measured before any comparison and are why every arm is replicated and every effect is tested over independent units rather than items (Section 3.1): an item-level test would treat this inference-side churn as independent evidence.

I How Existing Memories Handle a Change

Table 9 is read from the source of each system rather than from its paper: `mem0ai` 2.1.0 (`configs/prompts.py`, the update prompt that asks the model for ADD, UPDATE, DELETE or NONE); `a-mem` (the *should_evolve* prompt in `memory_system.py`; notes carry no status field); the in-context store of AWI, which assigns a new value over the old one under the same label; and AWE, which routes each write through the same `mem0ai` update prompt into a vector store.

Table 9: Who enforces what currently holds. Every existing system either asks the model to decide an update, or overwrites and loses the previous value.

System	How a change is recorded	Decided by	Previous value
AWI (in context)	value overwritten in place	store	discarded
Mem0(-G), AWE	model emits ADD/UPDATE/DELETE	model	kept, unlinked
A-MEM	model answers *should_evolve*	model	kept, unlinked
INCUMBENT (ours)	displacement is an effect of the write	store	marked; binds the reader

J WHAT FREE GRANULARITY COSTS

A hypergraph memory is only as strong as the object its invariants hold over. This appendix measures a permissive alternative to the design of Section 2 — hyperedges named freely by the maintainer, questions answered from the graph's own summaries — at the two points the design constrains, over ten LoCoMo conversations and twenty AMemGym users. The alternative fails in four ways, each answered by one decision of Section 2, and the last paragraph measures the same four quantities under the design.

Hyperedges become renamed vertices. With granularity left free the maintainer produced *thousands* of hyperedges over ten conversations, their number growing with the conversation rather than with the domain, and 64.2% of them hold exactly one vertex. Three quarters of those singletons have a sibling sharing their subject and leading topic word — one person's painting arrives as four separate one-vertex hyperedges. This is not a spelling problem: normalising case, plurals and token order merges 0.2% of them. A hyperedge with one vertex has no incumbent, so Equation (2) constrains nothing and supersession has nothing to demote.

Reuse becomes contingent on retrieval. The maintainer can only reuse an attribute it can see, and here what it sees is twice bounded: an index and a top-k render that together reach a seventh of one conversation's hyperedges, and less by the final call. Reuse competes with retrieval for one budget, and the budget loses as the conversation grows. Fixing granularity removes the competition rather than enlarging the budget: at tens of attributes per domain, every current value fits.

Optional closure does not happen, and the write is where the loss is. A maintainer does not close what it opened when closing is optional: the hypothesis bucket retired 0.3% of its entries, and a bucket that only grows is not a bound. The update itself fails the same way. At the moments a user's gold value changed, the relevant hyperedge was already in front of the maintainer three times out of four, and it was left untouched at 64.9% of those — invisibility explains at most the remainder.

Summaries are loosely coupled to the answer. If the reader answered from the memory, accuracy would track whether the memory held the right value. It tracks it weakly: classifying each hyperedge's stored text into the queried attribute's options, accuracy separates by only 10.5 pp between memories that hold the gold value and memories that do not, and the system answers more than half of the items with a wrong stored value correctly anyway. A graph rendered as its own summary does less work than its size suggests.

Table 10 maps each failure to the commitment it forces, and Section 2 adopts all four.

The same quantities under one attribute per hyperedge. Measured on the configuration of Section 2 over the same benchmarks, hyperedges holding a single vertex fall from 64.2% to below 10%; the maintainer sees every attribute's current value, so coverage is 100% by construction rather than 14.2%; closure retires more than half of the hypotheses it creates, against 0.3% when closure was optional; and accuracy conditioned on whether the memory held the correct value separates by $+26.1$ pp rather than $+10.5$. Part of that gain is the reader's: the summary reader over the same constrained graph separates by $+11.2$ pp (Appendix M).

Table 10: Each failure of the permissive configuration and the commitment it forces. The last paragraph of this appendix measures the middle column again on the constrained system, so each commitment is falsifiable on its own terms.

Failure	Measured	Commitment
hyperedges are renamed vertices	64.2% singletons	a hyperedge is one attribute (Section 2.1)
Reuse waits on retrieval	index covers 14.2%	the writer sees every attribute's current value; no index (Section 2.2)
Optional closure does not happen	577 active / 2 retired	closure runs on every write rather than being an operation the model may issue
Summaries are weakly coupled	+10.5 pp conditional gap	the reader answers from the user's words (Section 2.2)

K How Diluted the Retrieval Query Was

The agent interface these benchmarks define hands the reader a single string — the task prompt with the question inside it — so a read path that embeds what it is given embeds the instructions too. We measured how much this changes the query on the 105 questions of one LoCoMo conversation, embedding each with `text-embedding-3-small` and averaging the cosine similarity over 600 sampled pairs. The questions alone have a mean similarity of 0.342; wrapped in the instructions every item shares, 0.792. The wrapper more than doubles how alike two queries look, and what a retriever can rank on is the part that remains.

The corresponding ablation on AMemGym (Section 3.3) retrieves with the full task prompt instead of the question alone. It changes accuracy by 0.2 percentage points over 2,200 items, equivalent to zero under the ±2 margin. The two numbers together are the finding: a retrieval signal can be degraded this visibly while the answers barely move, as expected of a reader whose contribution is the currency it enforces rather than the volume of text it recovers.

L Cost

Section 3.2 quotes the ratios; Table 11 is the table they come from.

Table 11: Cost on the same runs. *s/turn* is wall-clock per ingested turn. AI-LieDar *Tok* is the total input tokens over the benchmark run. AMemGym *Tok/turn* is input tokens per user turn in thousands, write and read calls together, model and embedding calls alike, excluding each call's fixed instruction block, which providers cache (the quantity in Figure 1b and Figure 3c). *Ctx* is the context tokens per LoCoMo question, excluding the fixed reader instruction and the question itself.

	AI-LieDar		AMemGym		LoCoMo
Memory agent	s/turn ↓	Tok ↓	s/turn ↓	Tok/turn ↓	Ctx ↓
Pure LLM	1.89	894K	11.7	25.0	18,307
RAG	2.52	1,756K	30.0	17.2	1,934
A-MEM	16.02	4,632K	12.5	11.2	2,520
AWI	2.26	1,194K	11.1	8.0	1,210
Mem0-G	33.75	1,725K	44.5	12.6	973
AWE	5.49	1,209K	8.9	9.5	1,480
INCUMBENT (ours)	4.33	995K	14.0	5.4	2,000

Where the tokens go. Per AMemGym user turn and excluding each call's fixed instruction block, INCUMBENT spends 2.1K input tokens maintaining the store and 3.3K reading it (Figure 3c). RAG spends 1.7K embedding every new message and reads 15.5K, because its thirty retrieved items are raw messages, the assistant's long replies included. AWI, AWE, A-MEM and Mem0-G spend 3.6–8.2K writing, since each write call re-describes the new messages against the stored memory, and 4.4–4.8K reading, since four recent messages enter each reply beside the retrieved items or, for AWI,

the whole list of values. Pure LLM spends nothing on maintenance and 25.0K on reading, having filled its context window by the third period (Figure 3d). Neither term of INCUMBENT carries the transcript; its cost grows with the store.

The write stays bounded. Figure 4 plots the input to each LoCoMo maintainer call against the transcript accumulated so far, the quantity bounded by Equation (4). The input grows $4.2\times$ from the twentieth turn to the last while the turns grow $28\times$, and by the last turn the transcript is $3.1\times$ that input. Both counts of Equation (4) grow far more slowly than the transcript: over an AMemGym user the attribute set grows $3.0\times$ while the transcript grows $11\times$, and open concerns hold at 0.53 per hyperedge. On AI-LieDar's twenty-turn scenarios INCUMBENT spends less than every memory agent over the benchmark (995K input tokens against 1,194–4,632K) and 11% more than Pure LLM, whose transcript is still short.

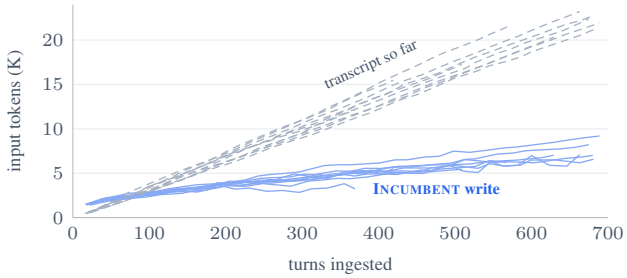

Figure 4: Input to one maintainer call, excluding its fixed instruction block, against the transcript an agent holding the conversation must read, over the ten LoCoMo conversations (one solid and one dashed line each).

M POINTERS, NOT SUMMARIES

A graph can answer from what it wrote down or from what it points at, and the two differ in how much the stored value constrains the answer. On AMemGym, conditioning accuracy on whether the memory held the correct value, a reader that answers from the located sentences separates correct from incorrect memory by $+26.1$ pp, against $+11.2$ pp for a reader given the graph's own summaries of the same vertices. A summary keeps which value of an attribute is current and discards the sentence that conveyed it, so a question whose answer turns on the user's own wording has nothing left to match. Keeping the span makes the memory decisive rather than decorative.

N WORKED EXAMPLES

Currency on AMemGym. One user's bridge gathering, as stored after the eleventh period (spans abridged):

Bucket	Vertex	Turn	Status	Proposition *and verified span*
`F_current`	F52	75	current	winter bridge games now have three to five close friends *"smaller, more intimate games with just a few close friends"*
`F_history`	F38	53	superseded	about eight to ten older regulars *"our weekly card games have settled into … about 8 or 10 older regulars"*
	F19	25	superseded	planning a bigger bridge gathering next week
	F1	1	superseded	a bridge gathering of six to twelve friends

The question *"What are some engaging activities I can organize for my monthly bridge gatherings?"* has the gold state *small (3–5)*. A reader that ranks by similarity represents the attribute by F38, whose wording ("our weekly card games … regulars") sits closer to the question, and answers for a group of eight to ten. By Proposition 3 INCUMBENT represents it by F52 and answers from the located sentence about a few close friends.

Provenance on AI-LieDar. In a scenario involving a job candidate, the initialization call turns the private record into

Vertex	Provenance	Status	Proposition
F1	verified, unfavourable	hidden	based on her résumé and difficulty answering basic questions, she currently has little chance of receiving the offer
H1	inferred	active	she may be seeking reassurance and may not yet recognise how much her interview performance affects her prospects

When the user asks for an honest assessment of her chances, the question turns on F1, which is verified, unfavourable and hidden, so the reply policy (Appendix A) requires the assistant to state it; H1 may shape how it is said but may not be asserted. The maintainer's later writes about the user enter as user-reported or inferred, so no turn of the conversation can raise them to the authority F1 holds.