GROUNDED CONTINUATION: A LINEAR-TIME RUNTIME VERIFIER FOR LLM CONVERSATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

In a long conversation, an LLM may produce a fluent continuation that rests on premises the conversation has already abandoned. Context-manipulation attacks exploit precisely this weakness. We address this problem with a runtime verifier. An LLM Interpreter maps each utterance to one or more of eight epistemic operations, and then a symbolic engine applies these operations to a dependency map that records what every claim rests on and whether it still stands. Based on the dependency map, checking whether a continuation is grounded then reduces to a walk over the map, linear in its size and requiring no LLM call. Retraction propagates through the same map with a conflict-free guarantee and flags exactly the conclusions that lose support. Our experiments with five QA models demonstrate substantial improvements in QA accuracy at a low cost per query. On ReviseQA for belief revision and MemoryAgentBench's FactConsolidation split (MemAB-FC), the verifier outperforms a retrieval baseline and raises MemAB-FC single-hop accuracy from 0.46–0.95 to 0.94–0.98. With the verifier, even the small 7B model overtakes unaided GPT-4o. When a GPT-4o Interpreter extracts every update from raw text rather than taking the benchmarks' structured updates, the verifier still outperforms the retrieval baseline. Moreover, on MemAB-FC, QA prompts remain compact at 97–174 tokens while the full-context baseline reaches 114.5K. Retraction queries take less than a microsecond at 2000 turns.

1 INTRODUCTION

Large language models produce continuations that can be locally fluent and pragmatically plausible yet *ungrounded*: disconnected from the claims, observations, and revisions that the conversation has actually established (Sirdeshmukh et al., 2025; Wilie et al., 2024). The deployment question is concrete. Given the next LLM output, can we check, *at this turn* and within a tight latency budget, whether that output traces back to those prior commitments? Answering "why did we reject approach A?" or "what assumptions does our current decision rest on?" requires tracking dependencies and revisions, even when the relevant utterances remain inside the context window. What is missing is a maintained structure connecting the LLM's continuation to the conversation's prior commitments. Figure 1 illustrates the problem on a fact-revision task: retrieval-augmented generation (RAG) presents superseded and current facts together, whereas our verifier maintains their standing and follows links between active records to supply evidence to the QA model.

This problem is empirically severe. Laban et al. (2026) found a 39% average performance drop in multi-turn versus single-turn settings across 200,000+ simulated conversations, with LLMs failing to revise incorrect assumptions even when later turns contradict them: "when LLMs take a wrong turn in a conversation, they get lost and do not recover." Shaikh et al. (2025) show, on real human-LLM dialogues, that LLMs initiate clarification three times less often than human partners and that early grounding failures predict later breakdowns. The sharpest instance is continued reliance on a premise that the conversation has revised or superseded. The gap is also actively exploited: context-manipulation attacks against deployed agents (Patlan et al., 2025; Dong et al., 2025; Hu et al., 2026a) craft continuations that are locally consistent yet disconnected from the conversation's prior commitments, triggering unauthorised actions.

Retrieval-augmented attribution methods address *external* grounding. A generated claim is accepted if it traces to a cited document (Gao et al., 2023; Bohnet et al., 2022). We focus on *interactional*

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.

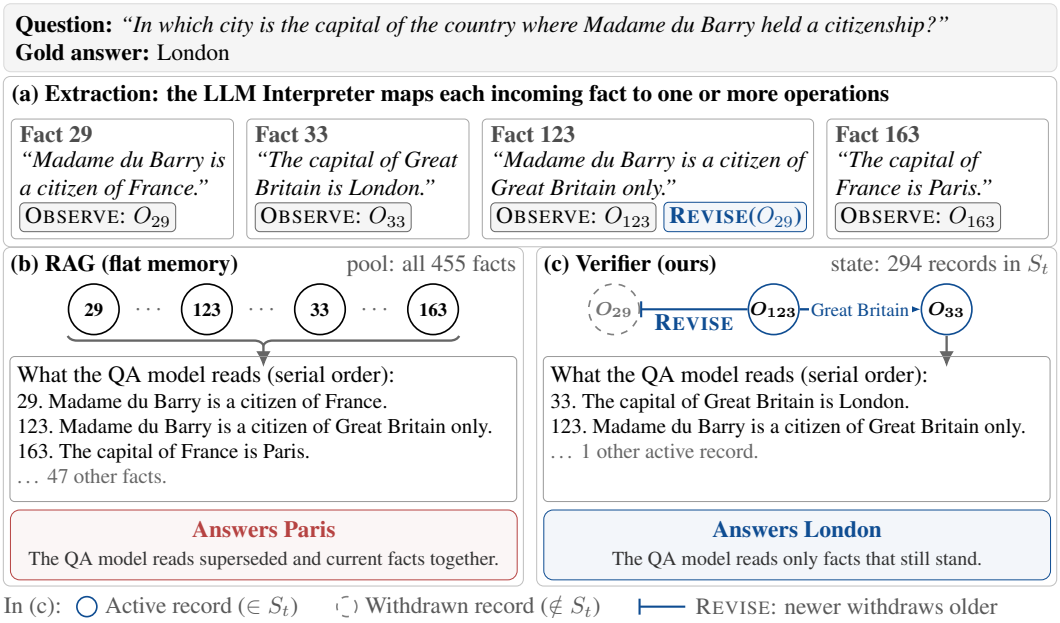

Figure 1: **Maintaining standing changes the evidence available to QA. (a)** The Interpreter maps incoming facts to OBSERVE and REVISE operations. **(b)** RAG presents superseded and current facts together. **(c)** The verifier follows the recorded values among the active records to the answer.

grounding in long-horizon LLM conversations. A continuation is acceptable if it is supported by the claims, observations, hypotheses, and revisions made earlier in the same conversation. Mem0 consolidates and updates conversational facts (Chhikara et al., 2025), while Zep additionally tracks their temporal validity (Rasmussen et al., 2025). Neither explicitly maintains the dependencies from premises to conclusions needed to check a continuation's support and propagate retractions.

A runtime check must inspect either the generating network or the continuation's support in the conversation state. White-box verification of neural networks is established (Huang et al., 2017; 2020) but faces severe complexity barriers for transformers. Even deciding emptiness is EXPSPACE-complete for fixed-precision unique-hard-attention transformers (Bergsträßer et al., 2026). We instead verify a continuation's support against the maintained state in linear time.

We implement this check with a runtime verifier. An LLM Interpreter maps each utterance to one or more of eight operations, and a symbolic engine applies them to a dependency map that records what each claim rests on and whether it still stands. At any turn the check asks whether a candidate continuation rests, through claims still in good standing, on the conversation's prior commitments, and answers in time linear in the map's size. Language understanding stays with the LLM, while dependency tracking stays with the engine, whose retraction guarantee holds by construction. The maintained state also supplies QA models with currently active evidence, connecting the grounding mechanism to downstream question answering.

Our main contributions are as follows:

- **Runtime Verification of Interactional Grounding**: at each turn we check whether a candidate continuation traces back through a maintained dependency map to prior commitments.
- **Better Performance and Versatility**: across three benchmarks, our verifier outperforms RAG on revision and consolidation, remains competitive on open-domain conversations, and lifts the weakest model (7B) above unaided GPT-4o on ReviseQA.
- **Formal Guarantees**: on the maintained dependency map, we prove conflict-free updates, exact identification of a retraction's recorded dependents with unaffected arguments preserved, and linear-time grounding and retraction queries.
- **Efficient Queries at Scale**: prompt tokens remain compact at 97–174 against 6.0K–114.5K for full context, and a retraction query runs in under a microsecond at 2000 turns.

2 THE RUNTIME VERIFIER

The verifier checks candidate continuations against a maintained dependency map. Its state combines an epistemic model of what is known, an argumentation framework of what attacks what and what currently stands, commitment records of who said what publicly, and the dependencies linking each claim to the premises it rests on. Definition 2.1 formalises these four components as $\mathcal{D}_t$.

Preliminaries. An argumentation framework is a set of arguments with an attack relation (Dung, 1995). Two arguments *conflict* if one attacks the other. A set of arguments is *conflict-free* if it contains no conflicting pair, *admissible* if it is conflict-free and attacks every attacker of its members, and a *preferred extension* is a maximal admissible set. The epistemic side uses dynamic epistemic logic (DEL) over plausibility models. Its operators and semantics are recalled in Appendix A, and Appendix B.1 calibrates the world-set machinery on the muddy-children puzzle (Phase 1).

Definition 2.1 (Dependency map)**.** Let $Prop_t$ be the conversation's propositions and Ags its agents. A *dependency map* at turn t is a tuple $\mathcal{D}_t = (\mathcal{M}_t, AF_t, Cm_t, Dep_t)$, where:

- $\mathcal{M}_t$ is an epistemic plausibility model over $Prop_t$ (Definition A.1; Baltag & Smets 2008);
- $AF_t = (Args_t, Att_t, st_t)$ is a labelled Dung argumentation framework with a finite set of arguments $\alpha \in Args_t$ carrying claims $claim(\alpha) \in Prop_t$, attack relation $Att_t \subseteq Args_t \times Args_t$, and standing labels $st_t : Args_t \to \{\mathsf{active}, \mathsf{weakened}, \mathsf{abandoned}, \mathsf{resolved}\}$;
- $Cm_t : Ags \to \mathcal{P}(Args_t)$ records each agent's public commitments (Walton et al., 2008);
- $Dep_t : Args_t \to \mathcal{P}(Args_t)$ maps each argument to its recorded supporting arguments, identified by their unique identifiers.

We write $S_t := \{\alpha \in Args_t : st_t(\alpha) \in \{\mathsf{active}, \mathsf{resolved}\}\}$ for the *active set* of arguments in good standing. Grounding and retraction queries read the current records without changing them. The formal foundation of these four components and the proofs of every statement in this section are in Appendix A. Given an associated record identifier, the dependency walk uses only Dep_t, argument membership, and standing labels.

The runtime check. The verifier tests a single property of a candidate continuation, *groundedness* with respect to the current dependency map.

Definition 2.2 (Grounded continuation)**.** Verification takes a candidate continuation c asserting ϕ_c together with an associated record identifier α_c satisfying $claim(\alpha_c) = \phi_c$. The *dependency walk* follows each link $\alpha \to \beta$ with $\beta \in Dep_t(\alpha)$, starting at α_c. Write $Dep_t^*(\alpha_c)$ for all reachable arguments, including α_c. The continuation is *grounded* under this association iff every argument in this set belongs to S_t. Otherwise, including when no record is associated, it is *ungrounded*.

With the association fixed before the walk (Appendix A), the verifier returns:

$$\mathsf{Verify}((c, \alpha_c), \mathcal{D}_t) = \begin{cases} \langle \mathsf{grounded}, Dep_t^*(\alpha_c) \rangle & \text{if } \alpha_c \text{ is associated and its walk succeeds,} \\ \langle \mathsf{ungrounded}, \emptyset \rangle & \text{otherwise.} \end{cases}$$

A grounded continuation carries the chain of upstream commitments it rests on. The walk stops at the first failing step. Ungrounded continuations are flagged for retry, retraction, or human review.

Lemma 2.1 (Grounding correctness)**.** *For a candidate associated with $\alpha_c \in Args_t$, the walk accepts iff $Dep_t^*(\alpha_c) \subseteq S_t$, and on acceptance returns exactly this dependency closure.*

Threat model. Candidate continuations may arise from adversarial manipulation of the generating model's context or memories (Patlan et al., 2025; Dong et al., 2025), or from ordinary model drift. At verification time, we assume that the adversary may choose c but cannot alter the conversation record supplied to the Interpreter or the maintained $\mathcal{D}_t$. Verification reads this state without updating it. For the associated record, Lemma 2.1 ensures that an accepted continuation has a support chain consisting entirely of arguments in good standing in $\mathcal{D}_t$:

$$\mathsf{Verify}((c, \alpha_c), \mathcal{D}_t) = \langle \mathsf{grounded}, Dep_t^*(\alpha_c) \rangle \implies Dep_t^*(\alpha_c) \subseteq S_t.$$

Retraction queries. The engine also tracks how premise retractions affect existing arguments. It follows dependency links in reverse from the withdrawn premise's recorded argument, distinguishing separate assertions of the same proposition.

Definition 2.3 (Affected set)**.** Let $\beta \in Args_t$ identify the premise record queried for retraction. The one-step and iterated affected sets are

$$\begin{aligned} \mathit{Affected}(\beta) &:= \{\alpha \in Args_t : \beta \in Dep_t(\alpha)\}, \\ X_0 &:= \mathit{Affected}(\beta), \\ X_{k+1} &:= X_k \cup \{\alpha \in Args_t : Dep_t(\alpha) \cap X_k \neq \varnothing\} \quad (k \geq 0), \\ \mathit{Affected}^*(\beta) &:= \textstyle\bigcup_{k\geq 0} X_k. \end{aligned}$$

$\mathit{Affected}^*(\beta)$ reports the recorded dependents for actual or counterfactual retraction, without changing their standing. The sequence X_k stabilises within $|Args_t|$ rounds. Both sets include records already out of standing. Their intersection with S_t identifies the currently active dependents.

Updating $\mathcal{D}_t$. The structure the walk traverses is not a list of mentioned propositions. It is populated by hypotheses generated through abduction (Definition A.2) over an awareness structure (Definition A.3) the conversation expands, and its standing labels, attacks, and dependencies are updated jointly each turn. An LLM Interpreter maps each utterance to one or more of eight operations and extracts their arguments. The engine applies the resulting operation instances in a fixed order, using the per-operation rule in Algorithm 1, to produce the per-turn update $\mathcal{D}_t \mapsto \mathcal{D}_{t+1}$ over $\mathcal{M}$, AF, Cm, and Dep.

The eight operations split into three roles. OBSERVE and RESOLVE commit content. HYPOTHESIZE, SUPPORT, UNDERMINE and REVISE adjust plausibility or attacks without erasing provenance. EXPAND-AWARENESS and QUESTION expand or query without committing claims. The taxonomy collects the update primitives of the four components. Multiple operation instances from one utterance are composed into the same per-turn state transition.

In the table below, ψ denotes an observed formula, γ a hypothesis, p a proposition, χ a query, and $\mathcal{A}_i$ the awareness set of agent $i \in Ags$ (Definition A.3).

The eight operations and their DEL realisations

Operation	DEL realisation	Example
OBSERVE	$[!\psi]$ (hard public announcement)	"I see 401 errors."
HYPOTHESIZE	$[\Uparrow \gamma]$ (soft lexicographic upgrade)	"Could be a retry loop."
SUPPORT	$[\uparrow \gamma]$, or $[\Uparrow \gamma]$ if γ-specific evidence	"Timing is consistent."
UNDERMINE	family of plausibility downgrades of γ	"Wrong error code, 401, not 503."
REVISE	$[!\neg\gamma]$ or Att edge addition	"It's reversed: A causes B."
EXPAND-AWARENESS	add p to the awareness set $\mathcal{A}_i$; world-set refinement on p	"The DB alert is really Redis."
RESOLVE	consensual $[!\gamma]$, or decision recording dissent	"Confirmed: bug is line 42."
QUESTION	no DEL update; χ queued as an open abductive problem	"Why is traffic 3× normal?"

Soundness of selective retraction. The operations maintain one invariant on standing, and the retraction guarantee rests on it alone. For $X \subseteq Args_t$, write $AF_t \restriction X$ for AF_t with the arguments in X removed. We use this restriction to analyse retraction under the semantics of Dung (1995). The maintained map retains the withdrawn records.

Lemma 2.2 (Standing invariant)**.** *Under the operation preconditions of Table 3, no argument of S_t is the target of an attack in Att_t. In particular, S_t is conflict-free in AF_t at every turn.*

Attacks enter only through UNDERMINE and REVISE, which leave their target outside S_t, and RESOLVE applies only to arguments in S_t with no recorded incoming attack. Weakened and abandoned arguments therefore stay outside S_t. A renewed assertion of their claim enters as a new argument with its own support references.

Proposition 2.3 (Conflict-free selective retraction)**.** *Let S_t be the active set after a premise record β is withdrawn, so $\beta \notin S_t$. Write $R := \{\beta\} \cup \mathit{Affected}(\beta)$. Then $S' := S_t \setminus R$ is conflict-free in $AF_t \restriction R$, some preferred extension of this restriction contains S', and every such extension preserves every $\alpha \in S_t$ with $\beta \notin Dep_t(\alpha)$.*

Corollary 2.4 (Exact iterated retraction)**.** *For any $\beta \in Args_t$, $\mathit{Affected}^*(\beta)$ contains exactly the arguments with a nonempty recorded dependency path to β. Under the hypotheses of Proposition 2.3,*

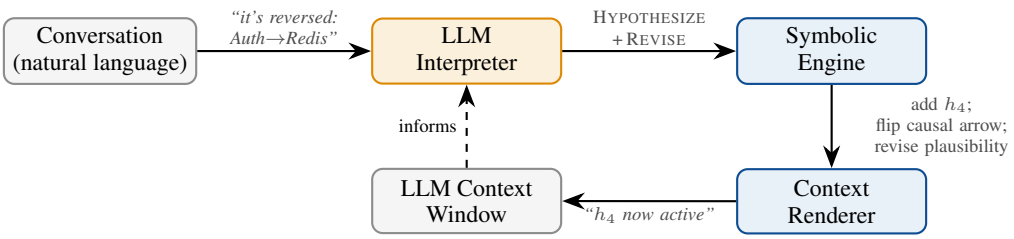

Figure 2: Hybrid architecture, illustrated with Phase 2 T12 (an engineer reverses the causal direction). Solid: utterance flow, classified operations, engine update, summary. Dashed: the rendered state fed back into the LLM's context.

writing $R^* := \{\beta\} \cup \mathit{Affected}^*(\beta)$, $S_t \setminus R^*$ *is conflict-free in* $AF_t \restriction R^*$, *and every* $\alpha \in S_t$ *with no such path is preserved.*

Per-turn tractability. Given the associated record, grounding and retraction reduce to reachability over Dep_t, computable in time linear in $|Args_t| + |Dep_t|$ with no LLM call. Standing is maintained by the update operations (Lemma 2.2), so the walk reads labels without recomputing argumentation extensions. For state updates, we analyse tractability under three structural restrictions satisfied by our scenarios: (A1) at most 12 agents; (A2) an acyclic attack graph, inherited from IBIS-style deliberation (Kunz & Rittel, 1970) in which each new argument supports, attacks, or refines an existing one rather than cycling back; (A3) an explicit finite world set W for $\mathcal{M}_t$ (Definition A.1). The general-case complexity is discussed in Appendix A.

Proposition 2.5 (Tractability under structural restrictions)**.** *Under (A1)–(A3), per-turn computation is polynomial in* $|W|$ *and* $|Args_t|$*:* $\mathcal{O}(|W|)$ *per hard announcement,* $\mathcal{O}(|W| \log |W|)$ *per soft upgrade,* $\mathcal{O}(1)$ *per standing update, and* $\mathcal{O}(|Args_t| + |Dep_t|)$ *per* Verify *walk or* $\mathit{Affected}^*$ *query over a maintained reverse dependency index.*

3 Architecture and Validation Scenarios

The verifier runs as a loop between an LLM Interpreter, a symbolic engine, and a context renderer (Figure 2), validated on two authored multi-agent scenarios, Phases 2 and 3. Throughout, o_i denotes an observation, h_i a hypothesis, a_i an assumption a decision rests on, and Tn the n-th turn.

LLM Interpreter and Symbolic Engine. For each utterance u_t, the Interpreter performs (1) *operation classification* into the eight operations of Section 2 and (2) *proposition extraction* of the relevant content and record identifiers. The engine applies the extracted operations in a fixed order to update standing and dependencies. Table 3 gives the operations' preconditions, and grounding and retraction queries then inspect the resulting state.

Context Renderer. After each engine update, the Renderer summarises the changed parts of $\mathcal{D}_t$ in natural language and writes the summary into the LLM's context window (dashed arrow in Figure 2). The next classification step can then read the *rendered state* of $\mathcal{D}_t$ alongside the conversation, as in the state-augmented condition of Table 2. The Renderer's output is also what users see when the verifier flags an ungrounded continuation: "decision α no longer grounded; affected by retraction of a_3; depends on a_1, a_2."

Phase 2: Naturalistic debugging. Three engineers debug a cascading failure (Appendix B.2). The scenario exercises hypothesis lifecycle tracking and counterfactual retraction: when an observation is invalidated, which conclusions lose grounding? Ground truth: token bug → retry storm → Redis exhaustion → rate-limit bypass → Stripe 429s, plus a monitoring miscategorisation. The conversation generates four hypotheses with non-trivial lifecycles, two of them decisive: h_2 (Redis→Auth) is undermined at T7 by error-code evidence and abandoned at T9, and h_4 at T12 reverses the causal direction, subsuming h_1 and h_3 at T13. The verifier is queried with four retractions ($\mathit{Affected}^*(o_8)$, $\mathit{Affected}^*(o_9)$, $\mathit{Affected}^*(o_6)$, and the three-hop $\mathit{Affected}^*(o_5)$ via reverse traversal $o_5 \to h_3 \to h_4 \to h_1$). Correctness depends on whether the dependency map records *why* h_2 was abandoned and preserves the provenance of h_4's causal reversal: both are recorded in the map, whereas flat history must be replayed to recover them.

Phase 3: Open-ended deliberation. A team chooses a real-time collaboration architecture (Appendix B.3). The scenario exercises commitment tracking and selective revision under assumption change: when an upstream assumption flips, which decisions need re-grounding and which remain intact? The outcome is a *decision with dissent*: Alice commits to Yjs server-relayed under three assumptions (a_1 docs stay short; a_2 editing remains burst-mode; a_3 Q2 long-running documents unconfirmed). Bob records dissent. When a_3 later flips, $\mathit{Affected}^*(a_3)$ must flag exactly the Yjs decision while leaving the access-control resolution intact: the selective re-grounding capability that motivates the verifier.

4 EXPERIMENTS

Our experiments address the following four questions. Does maintaining standing improve QA performance, and on which tasks? Does this improvement persist when every update is interpreted from raw text? What does dependency closure contribute beyond a one-step retraction query? What does maintaining and querying the state cost?

4.1 EXPERIMENTAL SETUP

Benchmarks. Three third-party benchmarks carry the main evaluation (protocols in Appendix C). *ReviseQA* (Helwe et al., 2025) poses a conclusion to re-evaluate after each of seven sequential context edits, and we run all 930 verified scenarios. *MemAB-FC*, the FactConsolidation split of MemoryAgentBench (Hu et al., 2026b), streams numbered facts of which 35–40% are later replaced and asks for *current* values at four stream lengths from 6K to 262K tokens. *BEAM* (Tavakoli et al., 2026) holds, in its 100K split, 20 long open-domain conversations with 20 probing questions each, two per memory ability from information extraction to knowledge update.

Arms. *LLM-only* answers from the raw context. *RAG* (Lewis et al., 2020) retrieves from the conversation itself, following each benchmark's own design where one exists. *Verifier (native)* feeds the engine the benchmark's structured updates, and the QA model reads the recorded premises. *Verifier (end-to-end)* has a GPT-4o Interpreter produce every update from the raw text, and is the only verifier arm on BEAM, which supplies no structured updates. The two verifier arms separate the engine's guarantees (Lemma 2.2 and proposition 2.3) from the Interpreter's faithfulness in recovering $\mathcal{D}_t$. On ReviseQA the verifier reads the premises the conclusion rests on, and RAG retrieves top-22 statements from the post-edit context. On MemAB-FC RAG retrieves top-50 facts, and Verifier (native) follows subject–object links from the entities the question names. On BEAM RAG retrieves turn pairs, and the verifier reads source turns with a retraction list, 3.3K and 3.7K tokens on average, respectively (Appendices C and D.4).

Models and metrics. The QA model varies over Qwen2.5-7B-Instruct and Qwen2.5-14B-Instruct (Yang et al., 2024), Gemma-3-12B-it (Gemma Team et al., 2025), GPT-4o-mini (OpenAI, 2024) and GPT-4o (Hurst et al., 2024), with the Interpreter held at GPT-4o. The classification experiment (Table 2) varies the Interpreter instead, over Claude Sonnet 4 (Anthropic, 2025), GPT-4o, Qwen2.5-7B-Instruct and Llama-3.1-8B-Instruct (Grattafiori et al., 2024). ReviseQA reports its own accuracy@k, the fraction of scenarios whose first k edits are all answered correctly, with $k = 2$ and 7 the benchmark's easy and hard levels. MemAB-FC reports single-hop and multi-hop accuracy pooled over the four stream lengths. Classification reports multi-label F1 and exact match, overall and on each scenario's *key shifts*, the turns at which standing or awareness changes. BEAM scores each answer with an LLM judge (Zheng et al., 2023) against the benchmark's rubric.

4.2 DOWNSTREAM EFFECTIVENESS

Verifier versus retrieval. On ReviseQA the native verifier leads RAG at every revision depth on all five QA models, significantly on four of them at every depth and on Gemma-3-12B through $k \leq 6$ (exact McNemar, $p < 0.05$; Table 1, per-depth curves in Appendix D.1). On MemAB-FC it lifts single-hop accuracy from RAG's 0.46–0.95 to 0.94–0.98 and multi-hop from 0.04–0.13 to 0.64–0.74. On the 336 of 400 multi-hop questions for which the active records contain a chain linking the question to the gold answer, it reaches 0.74–0.85. For 61 of the other 64, the chain to the gold answer passes through a superseded record (Appendix D.3). The gain is nearly constant across QA models, 0.60–0.65 on multi-hop. Under retrieval the strongest model answers three times as many

Table 1: Five QA models under three arms. The verifier arm is native, and Section 4.3 compares it with the end-to-end arm. MemAB-FC pools the four stream lengths. Per-depth and per-length results and paired tests are in Appendices D.1 and D.2.

	ReviseQA (accuracy@2 / @7)			MemAB-FC (single-hop / multi-hop)		
QA model	LLM-only	RAG	Verifier	LLM-only	RAG	Verifier
Qwen2.5-7B	0.127 / 0.008	0.356 / 0.149	**0.544 / 0.202**	0.383 / 0.055	0.463 / 0.040	**0.975 / 0.642**
Qwen2.5-14B	0.222 / 0.010	0.388 / 0.168	**0.574 / 0.230**	0.443 / 0.048	0.760 / 0.083	**0.943 / 0.713**
Gemma-3-12B	0.374 / 0.045	0.512 / 0.283	**0.612 / 0.312**	0.398 / 0.048	0.917 / 0.083	**0.983 / 0.733**
GPT-4o-mini	0.277 / 0.015	0.416 / 0.123	**0.511 / 0.157**	0.395 / 0.052	0.708 / 0.065	**0.970 / 0.715**
GPT-4o	0.476 / 0.161	0.508 / 0.242	**0.628 / 0.348**	0.400 / 0.060†	0.950 / 0.130	**0.978 / 0.738**

† GPT-4o LLM-only was run on the 6K stream only.

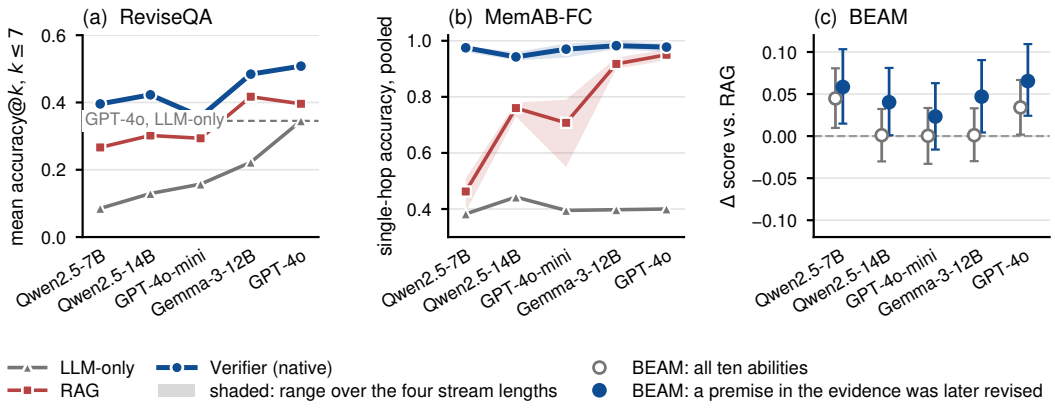

Figure 3: QA models ordered by LLM-only accuracy on ReviseQA. **(a)** Mean accuracy@k for $k \leq 7$. Dashed: GPT-4o, LLM-only. **(b)** MemAB-FC single-hop accuracy, pooled over stream lengths. Shading: range over lengths. **(c)** BEAM: Verifier (end-to-end) score relative to RAG, on all ten abilities and on questions with a later-revised premise. Error bars: 95% bootstrap intervals.

multi-hop questions as the weakest, and under the verifier all five fall within 0.10 of each other. Our RAG baseline is stronger than the benchmark's own retrieval agent (Appendix D.2).

QA model capability. Every QA model improves under the verifier, and the weakest readers gain most: mean accuracy@k for $k \leq 7$ rises from 0.085 to 0.396 on Qwen2.5-7B and from 0.345 to 0.508 on GPT-4o (Figure 3a). Qwen2.5-7B under the verifier, at 0.396, passes GPT-4o reading the raw context, at 0.345, and exceeds GPT-4o with RAG on MemAB-FC single-hop, 0.975 versus 0.950 (Table 1). Gains this uneven reorder the models: on ReviseQA, GPT-4o-mini leads Qwen2.5-7B unaided and trails it under the verifier.

Open-domain conversations. BEAM tests the verifier on long open-domain conversations. We label from the dependency map whether a question's evidence holds a premise the conversation later revised (Appendix D.4). Pooled over the five QA models, the end-to-end verifier scores 0.347 against RAG's 0.353 on the 780 question-model pairs where every premise still stands, and 0.249 against 0.202 on the 995 where one was revised (Figure 3c). Paired Wilcoxon tests put the verifier ahead on three QA models where a premise was revised, and detect no significant difference on any of the five where every premise stands.

4.3 End-to-end interpretation

QA accuracy. On ReviseQA, with a GPT-4o Interpreter producing every update from raw text, the two verifier arms differ by at most 2.26 percentage points of accuracy@2 and 1.94 of accuracy@7 across the five QA models, in both directions, and by at most 0.71 percentage points in the per-model mean over $k \leq 7$. The end-to-end arm stays above RAG at every depth on all five models (Appendix D.1). On MemAB-FC, pooling the 6K and 32K streams, end-to-end single-hop and

Table 2: Phase 2 classification under the Minimal (Min.), Definitions (Def.) and State-augmented (State-aug.) prompt conditions. Each cell is F1 / Exact match in percent. Phase 3 in Table 4.

	Overall (13 turns)			**Key shifts (4 turns)**		
Interpreter	Min.	Def.	State-aug.	Min.	Def.	State-aug.
Claude Sonnet 4	0.66 / 11	0.85 / 54	**0.91 / 63**	0.50 / 5	0.74 / 25	**0.83** / 30
GPT-4o	0.69 / 31	0.74 / 40	0.81 / 52	0.43 / 0	0.50 / 10	0.74 / **50**
Qwen2.5-7B	0.54 / 29	0.57 / 26	0.59 / 29	0.39 / 10	0.47 / 0	0.47 / 0
Llama-3.1-8B	0.54 / 12	0.55 / 20	0.65 / 32	0.37 / 0	0.47 / 20	0.56 / 15

multi-hop accuracy differs from native accuracy by at most 1.5 percentage points across Qwen2.5-7B, Qwen2.5-14B and Gemma-3-12B, and remains above RAG in every comparison.

Faithfulness. On ReviseQA the Interpreter emits a removal on $4{,}355$ of the $4{,}361$ edit steps that retire a premise inside the conclusion's maintained dependency set. On MemAB-FC, the dependency map the Interpreter builds from raw text matches the one built from the benchmark's structured updates at Jaccard 1.000 at 6K and 0.989 at 32K, with the Interpreter deciding which record each new fact supersedes (Appendices C.3 and D.2).

Prompt ablation. Table 2 varies what the Interpreter is told: the operation names only, full definitions with examples, or the definitions plus the rendered engine state at checkpoints. For Claude Sonnet 4, definitions alone lift Phase 2 F1 from 0.66 to 0.85, so the taxonomy is learnable from general definitions, and the rendered state reaches 0.91 overall and lifts the key shifts from 0.74 to 0.83. Interpreters differ widely under the same prompt, from 0.91 to 0.59 in the state-augmented condition. On Phase 3, evaluated under the Definitions condition only, Claude Sonnet 4 reaches 0.85, and every Interpreter matches or exceeds its Phase 2 F1 under the same prompt (Table 4).

4.4 Dependency closure

Closure versus one-step. Given ground-truth dependency structure we compare the one-step and iterated queries of Definition 2.3. Over the four retraction queries of the authored Phase 2 scenario, carrying 16 dependency decisions, the one-step query returns the whole affected set for one of them, and $\mathit{Affected}^*$ for all four. Over 64 counterfactual scenarios from 16 RECON case files (Arya, 2026), the one-step query is exact on 13 and $\mathit{Affected}^*$ on all 64, recovering 198 dependent steps with no false positives (Appendix E.1). This is Corollary 2.4 in practice: given a faithful $\mathcal{D}_t$, the iterated query returns exactly the arguments that lose support.

LLM reference. When GPT-4o at temperature 0 answers the same question, it is exact on all 64 scenarios from the isolated chain of about 550 tokens, and on 61 from RECON's own oracle setting, the full 51K-token skeleton, over-including one step in each of the other three. The iterated query computes the recorded dependency closure deterministically and without an LLM call.

4.5 Efficiency

Token cost. On MemAB-FC, the LLM-only prompt grows with the stream, up to 114.5K tokens. RAG retrieves the top-50 facts, 753–811 tokens per query at every length. Verifier (native) sends only the active records linked to the question, 97–174 tokens, 4.7–$7.7\times$ fewer than RAG and $686\times$ fewer than LLM-only at 262K (Figure 4a,b). It is also the most accurate arm at every length for all five QA models. On BEAM, Verifier (end-to-end) reads 3.7K tokens per question, about 3% of GPT-4o's full-context input (Appendix D.4). Verifier (native) takes the benchmark's structured updates and spends no LLM tokens at write time. Following the write and read accounting of Pollertlam & Kornsuwannawit (2026), Verifier (end-to-end) pays a one-time Interpreter ingestion cost of 96.6K tokens at 6K and 494.7K at 32K, which its per-query savings over LLM-only recover after 16.3 and 15.3 of the 100 queries per stream (Appendix E.2).

Retraction latency. With a reverse dependency index, a retraction query takes 0.25–$0.46\,\mu$s over states of 13–2000 turns, compared with 1.1–$190\,\mu$s for history replay (Figure 4c). The gap widens with the conversation, from $4\times$ at 13 turns to $415\times$ at 2000. Latency scales with the size of the affected set rather than the history length, and Proposition 2.5 bounds the worst case.

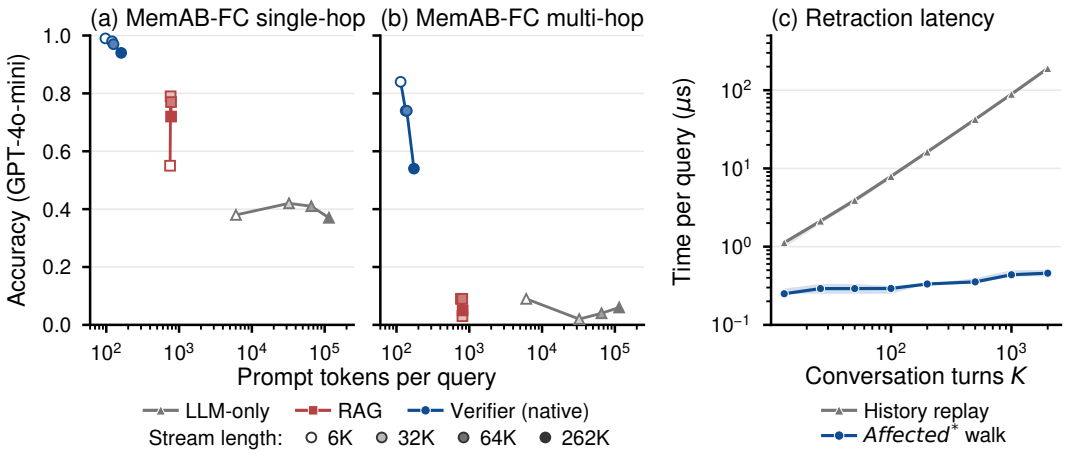

Figure 4: **Query costs. (a, b)** MemAB-FC single-hop and multi-hop accuracy versus prompt tokens per query. **(c)** Median retraction latency versus history replay.

5 Related work

RAG and attribution. RAG retrieves evidence for generation (Lewis et al., 2020), and ALCE and AutoAIS score whether a generated answer is supported by the documents it cites (Gao et al., 2023; Bohnet et al., 2022). Self-RAG adds retrieval and generation self-critique (Asai et al., 2024), GraphRAG organises evidence into entity graphs (Edge et al., 2024), and MTRAG evaluates retrieval and generation over multi-turn dialogue (Katsis et al., 2025). These approaches anchor support in source documents or retrieved passages. The verifier checks interactional grounding instead: whether a continuation rests on commitments that still stand in the conversation.

Conversational memory. LLM agents keep memory in many forms (Zhang et al., 2024). Mem0 consolidates conversational facts (Chhikara et al., 2025), A-MEM links and evolves memory notes (Xu et al., 2025), Zep invalidates temporally conflicting facts (Rasmussen et al., 2025), and D-SMART resolves conflicts in an evolving knowledge graph (Lei et al., 2025). None of them specifies a mechanism for computing the transitive effects of a premise withdrawal on conclusions derived earlier. VISTA reconstructs a reasoning dependency tree from a conversation history for interactive analysis (Zhang et al., 2026), without a runtime check of whether a continuation's support still stands. The verifier maintains dependencies and standing at every turn, answers a support query online, and returns exactly the recorded transitive dependents of a withdrawn premise (Appendix F).

Formal foundations. Truth-maintenance systems revise beliefs through recorded justifications (Doyle, 1979), DEL models epistemic updates (van Ditmarsch et al., 2007), and argumentation frameworks supply conflict semantics (Dung, 1995). Runtime verification checks an execution against a property as it runs (Leucker & Schallhart, 2009), the lightweight counterpart of the model checking over multi-agent models (Huang et al., 2011; Lomuscio et al., 2017). The verifier applies these foundations to one property, groundedness, over a state that an LLM Interpreter extracts from the conversation itself.

6 Conclusion

We presented a runtime verifier that checks, at each turn, whether the next LLM output still rests on what the conversation has established. Lemma 2.1 makes the check exact, and Lemma 2.2, Proposition 2.3 and Proposition 2.5 establish conflict-free retraction and per-turn tractability. Given ground-truth dependencies, the closure query recovers every gold affected set in Phase 2 and RECON, including indirect dependents that one-step queries miss. The verifier leads RAG on ReviseQA and MemAB-FC across five QA models, and on BEAM where evidence was later revised. It reduces per-query token cost on MemAB-FC by 4.7–$7.7\times$ relative to RAG and by up to $686\times$ relative to LLM-only, and a retraction query takes under a microsecond over 13–2000 turns.

AI USE STATEMENT

Large language models play two roles in this work: they are the systems under study, and they were used as research assistance. This statement concerns the second role. As systems under study, GPT-4o, GPT-4o-mini, Claude Sonnet 4, Qwen2.5, Gemma-3 and Llama-3.1 serve as the Interpreter and the QA models described in Section 4.1 and are the object of evaluation.

We used Claude (Anthropic) for two tasks with required disclosure: after the experiments had run, it drafted summaries of the results, which we checked against the result files, and it gave feedback on the experimental protocol and on one definition of the formal framework, which the authors evaluated and decided on. The research problem, the formal framework, the eight-operation taxonomy, and the proofs of all formal statements are the authors' own work. We did not use generative AI to generate synthetic data, to clean or reformat datasets, to translate, or to perform qualitative analysis. For tasks with recommended disclosure, we used Claude to edit the paper for readability and structure, to summarise related literature, and to write and edit code.

All AI-assisted work was verified by the authors: every reported number was checked against the committed result files, every script was tested on smoke subsets before full runs, every proof was checked line by line, and every AI suggestion was accepted or rejected by the authors, who made all changes themselves. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work involves no human subjects. The three conversation scenarios were written and labelled by the authors, and all other data are public benchmarks (ReviseQA, MemoryAgentBench, BEAM, RECON) used under the licenses of their releases. The verifier is a defensive mechanism against context-manipulation attacks on deployed agents. It introduces no new attack surface, and we release code and the recorded Interpreter outputs only.

REPRODUCIBILITY STATEMENT

All results run from the anonymised repository at `https://anonymous.4open.science/r/Epistemic-Conversation-Models-15ED/`. Appendix C states every protocol: benchmark splits, evidence selection and budgets, arm definitions, prompt conditions, model identifiers and serving, and statistical tests; the repository README documents the environment and how each runner is invoked. Proofs of all formal statements are in Appendix A, and the reference engine is deterministic and runs without API access. Third-party benchmarks are used as released by their authors; QA runs use temperature 0, and the classification experiment reports five runs per cell at temperatures 0 and 1.0. The recorded Interpreter outputs of the end-to-end runs are included, so these arms can be re-run without new Interpreter calls.

REFERENCES

Anthropic. System card: Claude Opus 4 & Claude Sonnet 4. `https://www.anthropic.com/claude-4-system-card`, 2025.

Mihir Shriniwas Arya. RECON: Benchmarking agent memory for compositional reasoning over long contexts. *arXiv preprint arXiv:2607.16716*, 2026.

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

Guillaume Aucher and François Schwarzentruber. On the complexity of dynamic epistemic logic. In *Proceedings of the 14th Conference on Theoretical Aspects of Rationality and Knowledge (TARK)*, 2013.

Alexandru Baltag and Sonja Smets. A qualitative theory of dynamic interactive belief revision. In Giacomo Bonanno, Wiebe van der Hoek, and Michael Wooldridge (eds.), *Logic and the Founda-*

tions of Game and Decision Theory (LOFT 7), volume 3 of *Texts in Logic and Games*, pp. 11–58. Amsterdam University Press, 2008.

Alexandru Baltag, Lawrence S. Moss, and Slawomir Solecki. The logic of public announcements, common knowledge, and private suspicions. In *Proceedings of the 7th Conference on Theoretical Aspects of Rationality and Knowledge (TARK)*, pp. 43–56. Morgan Kaufmann, 1998.

Pascal Bergsträßer, Ryan Cotterell, and Anthony W. Lin. Transformers are inherently succinct. In *International Conference on Learning Representations (ICLR)*, 2026.

Bernd Bohnet, Vinh Q. Tran, Pat Verga, Roee Aharoni, Daniel Andor, Livio Baldini Soares, Massimiliano Ciaramita, Jacob Eisenstein, Kuzman Ganchev, Jonathan Herzig, Kai Hui, et al. Attributed question answering: Evaluation and modeling for attributed large language models. *arXiv preprint arXiv:2212.08037*, 2022.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready AI agents with scalable long-term memory. In *Proceedings of the 28th European Conference on Artificial Intelligence (ECAI)*, pp. 2993–3000, 2025. doi: 10.3233/FAIA251160.

Shen Dong, Shaochen Xu, Pengfei He, Yige Li, Jiliang Tang, Tianming Liu, Hui Liu, and Zhen Xiang. Memory injection attacks on LLM agents via query-only interaction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2503.03704.

Sylvie Doutre, Andreas Herzig, and Laurent Perrussel. A dynamic logic framework for abstract argumentation. In *Proceedings of the Fourteenth International Conference on Principles of Knowledge Representation and Reasoning (KR)*, pp. 62–71. AAAI Press, 2014.

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

Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–357, 1995.

Paul E. Dunne. Computational properties of argument systems satisfying graph-theoretic constraints. *Artificial Intelligence*, 171(10–15):701–729, 2007.

Paul E. Dunne and Michael Wooldridge. Complexity of abstract argumentation. In Guillermo R. Simari and Iyad Rahwan (eds.), *Argumentation in Artificial Intelligence*, pp. 85–104. Springer, 2009.

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. URL `https://arxiv.org/abs/2404.16130`.

Ronald Fagin and Joseph Y. Halpern. Belief, awareness, and limited reasoning. *Artificial Intelligence*, 34(1):39–76, 1987.

Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. *Reasoning about Knowledge*. MIT Press, Cambridge, MA, 1995.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. Enabling large language models to generate text with citations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6465–6488, 2023.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Chadi Helwe, Sultan AlRashed, and Francesco Orabona. ReviseQA: A benchmark for belief revision in multi-turn logical reasoning. In *ICML 2025 Workshop on Assessing World Models*, 2025.

Jinwei Hu, Xinmiao Huang, Youcheng Sun, Yi Dong, and Xiaowei Huang. Lying with truths: Open-channel multi-agent collusion for belief manipulation via generative montage. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5979–5996, 2026a.

Yuanzhe Hu, Yu Wang, and Julian McAuley. Evaluating memory in LLM agents via incremental multi-turn interactions. In *International Conference on Learning Representations (ICLR)*, 2026b.

Xiaowei Huang, Cheng Luo, and Ron van der Meyden. Improved bounded model checking for a fair branching-time temporal epistemic logic. In *Model Checking and Artificial Intelligence (MoChArt 2010)*, volume 6572 of *Lecture Notes in Computer Science*, pp. 95–111. Springer, 2011. Feasibility of bounded epistemic model checking.

Xiaowei Huang, Marta Kwiatkowska, Sen Wang, and Min Wu. Safety verification of deep neural networks. In *International Conference on Computer Aided Verification (CAV)*, pp. 3–29, 2017.

Xiaowei Huang, Daniel Kroening, Wenjie Ruan, James Sharp, Youcheng Sun, Emese Thamo, Min Wu, and Xinping Yi. A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defence, and interpretability. *Computer Science Review*, 37:100270, 2020.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. GPT-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Yannis Katsis, Sara Rosenthal, Kshitij Fadnis, Chulaka Gunasekara, Young-Suk Lee, Lucian Popa, Vraj Shah, Huaiyu Zhu, Danish Contractor, and Marina Danilevsky. mtRAG: A multi-turn conversational benchmark for evaluating retrieval-augmented generation systems. *Transactions of the Association for Computational Linguistics*, 13:784–808, 2025.

Werner Kunz and Horst W. J. Rittel. Issues as elements of information systems. Technical Report 131, Institute of Urban and Regional Development, University of California, Berkeley, 1970.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. LLMs get lost in multi-turn conversation. In *International Conference on Learning Representations (ICLR)*, 2026.

Xiang Lei, Qin Li, Min Zhang, and Min Zhang. D-SMART: Enhancing LLM dialogue consistency via dynamic structured memory and reasoning tree. *arXiv preprint arXiv:2510.13363*, 2025.

Martin Leucker and Christian Schallhart. A brief account of runtime verification. *Journal of Logic and Algebraic Programming*, 78(5):293–303, 2009.

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

Alessio Lomuscio, Hongyang Qu, and Franco Raimondi. MCMAS: An open-source model checker for the verification of multi-agent systems. *International Journal on Software Tools for Technology Transfer*, 19(1):9–30, 2017. Earlier tool papers at TACAS 2006 and CAV 2009; often cited with earlier dates.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947.

Timothy Obiso, Kenneth Lai, Abhijnan Nath, Nikhil Krishnaswamy, and James Pustejovsky. Dynamic epistemic friction in dialogue. In *Proceedings of the 29th Conference on Computational Natural Language Learning (CoNLL)*, pp. 323–333, Vienna, Austria, 2025. Association for Computational Linguistics.

OpenAI. GPT-4o mini: Advancing cost-efficient intelligence. OpenAI Blog, `https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/`, 2024.

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 *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23)*, pp. 1–22. ACM, 2023.

Atharv Singh Patlan, Peiyao Sheng, S. Ashwin Hebbar, Prateek Mittal, and Pramod Viswanath. Real AI agents with fake memories: Fatal context manipulation attacks on Web3 agents. *arXiv preprint arXiv:2503.16248*, 2025.

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.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. Zep: A temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*, 2025.

Omar Shaikh, Hussein Mozannar, Gagan Bansal, Adam Fourney, and Eric Horvitz. Navigating rifts in human-LLM grounding: Study and benchmark. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 20832–20847, 2025.

Ved Sirdeshmukh, Kaustubh Deshpande, Johannes Mols, Lifeng Jin, Ed-Yeremai Cardona, Dean Lee, Jeremy Kritz, Willow Primack, Summer Yue, and Chen Xing. MultiChallenge: A realistic multi-turn conversation evaluation benchmark challenging to frontier LLMs. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 18632–18702. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.958. URL https://aclanthology.org/2025.findings-acl.958/.

Mohammad Tavakoli, Alireza Salemi, Carrie Ye, Mohamed Abdalla, Hamed Zamani, and J Ross Mitchell. Beyond a million tokens: Benchmarking and enhancing long-term memory in LLMs. In *International Conference on Learning Representations (ICLR)*, 2026.

Hans van Ditmarsch, Wiebe van der Hoek, and Barteld Kooi. *Dynamic Epistemic Logic*, volume 337 of *Synthese Library*. Springer, Dordrecht, 2007.

Fernando R. Velázquez-Quesada, Fernando Soler-Toscano, and Ángel Nepomuceno-Fernández. An epistemic and dynamic approach to abductive reasoning: Abductive problem and abductive solution. *Journal of Applied Logic*, 11(4):505–522, 2013.

Douglas Walton, Chris Reed, and Fabrizio Macagno. *Argumentation Schemes*. Cambridge University Press, 2008.

Peiran Wang, Yang Liu, Yunfei Lu, Yifeng Cai, Hongbo Chen, Qingyou Yang, Jie Zhang, Jue Hong, and Ye Wu. AgentArmor: Enforcing program analysis on agent runtime trace to defend against prompt injection. *arXiv preprint arXiv:2508.01249*, 2025.

Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.

Bryan Wilie, Samuel Cahyawijaya, Etsuko Ishii, Junxian He, and Pascale Fung. Belief revision: The adaptability of large language models reasoning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 10480–10496. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.586. URL https://aclanthology.org/2024.emnlp-main.586/.

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*, volume 38, 2025. doi: 10.52202/085713-0593. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/19909c36f51abc4856b4560aff3d36d6-Abstract-Conference.html.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Yiran Zhang, Mingyang Lin, Mark Dras, and Usman Naseem. Beyond the black box: Demystifying multi-turn LLM reasoning with VISTA. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 41745–41747, 2026. Demonstration Track.

Zeyu Zhang, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Quanyu Dai, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. A survey on the memory mechanism of large language model based agents. *arXiv preprint arXiv:2404.13501*, 2024.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, volume 36, 2023.

A Formal Framework and Proofs

A.1 Formal Definitions Deferred from Section 2

Definition A.1 (Epistemic Plausibility Model)**.** An *epistemic plausibility model* is a tuple $\mathcal{M} = (W, \{\leq_i\}_{i\in\mathrm{Ags}}, V)$ where:

- W is a non-empty set of possible worlds;
- each $\leq_i \subseteq W \times W$ is a reflexive, transitive, *locally connected* preorder: for all w, w' in the same indistinguishability cell, either $w \leq_i w'$ or $w' \leq_i w$;
- $V : \mathrm{Prop} \to \mathcal{P}(W)$ is a valuation.

Convention. We follow the standard Baltag–Smets reading: $w \leq_i w'$ means "w is at least as plausible as w' for agent i," so $\leq_i$-minimal worlds are most plausible. The indistinguishability relation is $\sim_i := \leq_i \cup \geq_i$. Different agents may have different information cells from the same w. Agent i *knows* φ at w iff φ holds at every w' with $w \sim_i w'$ (Fagin et al., 1995). Agent i *believes* φ at w iff φ holds at every $\leq_i$-minimal world in $\{w' : w \sim_i w'\}$.

Updates. The three dynamic operators in full: *Public announcement* (hard) $[!\psi]$ eliminates every $w \notin [\![\psi]\!]$. *Lexicographic upgrade* (radical, soft) $[\Uparrow \psi]$ makes every ψ-world strictly more plausible than every $\neg\psi$-world while preserving the within-side order. No worlds are eliminated. *Conservative upgrade* (minimal, soft) $[\uparrow \psi]$ promotes the most plausible ψ-worlds to the overall minimum, preserving all other comparisons. Event models (Baltag et al., 1998) generalise these when different agents perceive the same event differently.

Definition A.2 (Abductive Problem and Solution)**.** Given a model $\mathcal{M}$, actual world w, agent i, and observation χ with $\mathcal{M}, w \models \chi$ but $\mathcal{M}, w \not\models \mathbf{B}_i\chi$ (the observation is true but not previously believed, *surprising*), an *abductive problem* is the triple $(\mathcal{M}, i, \chi)$. A formula γ is an *abductive solution* when:

(1) *Consistency*: $\mathcal{M} \not\models \mathbf{B}_i\neg\gamma$;

(2) *Explanatory*: $\mathcal{M}[\Uparrow \gamma], w \models \mathbf{B}_i\chi$;

(3) *Non-triviality*: $\gamma \neq \chi$, $\mathcal{M} \not\models \mathbf{B}_i\gamma$, and $\gamma \neq \top$.

The solution is integrated via $[\Uparrow \gamma]$, so γ enters as belief, not knowledge.

Definition A.3 (Awareness Structure)**.** An *awareness structure* (Fagin & Halpern, 1987) extends an epistemic plausibility model with a function $\mathcal{A}_i : W \to \mathcal{P}(\mathrm{Form})$ for each agent i, where $\mathcal{A}_i(w)$ is the set of formulas i is aware of at w. The awareness modality satisfies $\mathcal{M}, w \models \mathbf{A}_i\varphi \iff \varphi \in \mathcal{A}_i(w)$. Explicit knowledge is $X_i\varphi \iff \mathbf{K}_i\varphi \wedge \mathbf{A}_i\varphi$. Awareness expansion adds formulas to $\mathcal{A}_i$ and refines W accordingly: each world splits into copies that differ on the new proposition.

Recorded support and traversal. Each member of $Dep_t(\alpha)$ is a supporting record identified by its ID. Different IDs remain distinct even when their claims coincide. The associated ID is fixed before traversal, and a candidate with no associated record is ungrounded. The formal check assumes that this association identifies the candidate's claim. The walk follows the recorded IDs and checks their current labels, including those of weakened or abandoned hypotheses. A visited set prevents repeated expansion. Records with no dependencies end a branch. Retraction queries traverse the same relation in reverse and return reports without changing standing. In the query bounds, $|Dep_t| := \sum_{\alpha\in Args_t} |Dep_t(\alpha)|$.

A.2 Proofs

Proof of Lemma 2.1 (Grounding correctness). Fix the candidate's associated record α_c and the finite map $\mathcal{D}_t$. The walk begins at α_c and expands exactly the recorded dependency links, so every visited argument lies in $Dep_t^*(\alpha_c)$. Conversely, if the walk does not fail, induction on dependency-path length shows that it visits every member of this closure. A visited set ensures termination even when dependency links form cycles. Every record is checked against S_t on its first visit. If all closure members lie in S_t, no check fails and the walk returns the complete closure. If some member lies

outside S_t, the walk either fails earlier or reaches such a member and rejects. Therefore acceptance is equivalent to $Dep_t^*(\alpha_c) \subseteq S_t$, and the returned set on acceptance is exactly the closure. □

Proof of Lemma 2.2 (Standing invariant). By induction on turns. Initially $Att_0 = \emptyset$. Inspecting Algorithm 1: OBSERVE, HYPOTHESIZE, SUPPORT, EXPAND-AWARENESS and QUESTION add no attack edge and label any new argument active; UNDERMINE, whose precondition requires an active or weakened target, adds $(\alpha_e, \alpha_\gamma)$ and leaves $st(\alpha_\gamma) =$ weakened; REVISE adds attacks on α_γ and sets $st(\alpha_\gamma) =$ abandoned; RESOLVE sets $st(\alpha_\gamma) =$ resolved only when $\alpha_\gamma \in S_t$ and no attack on it is recorded (Table 3); and no operation moves a weakened or abandoned argument back into S_t. Hence after every turn each attack targets an argument outside S_t, so no two arguments of S_t attack each other. □

Proof of Proposition 2.3 (Conflict-free selective retraction). Let $\mathcal{D}_t$ be a dependency map satisfying Lemma 2.2, with the withdrawn premise record $\beta \notin S_t$. Define $R := \{\beta\} \cup \mathit{Affected}(\beta)$, $S' := S_t \setminus R$, and $AF' := AF_t \restriction R$. Its argument set is $Args_t \setminus R$ and its attack relation is $Att_t \cap (Args_t \setminus R)^2$.

Conflict-freeness and preservation: S_t is conflict-free by Lemma 2.2, and $S' \subseteq S_t$. Restriction cannot create new attacks, so S' is conflict-free in AF'. Since $\beta \notin S_t$, an argument $\alpha \in S_t$ is removed precisely when $\beta \in Dep_t(\alpha)$. Thus the one-step removal preserves exactly those members of S_t without a direct recorded dependency on β.

Admissibility and preferred extensions: no argument of S_t is the target of any attack in AF_t by Lemma 2.2. Hence no argument of S' is attacked in AF', and S' is admissible: it defends itself vacuously. Every admissible set is contained in some preferred extension (Dung, 1995). Such an extension contains every preserved member of S_t and excludes R, whose records are absent from the restricted framework. The engine maintains standing directly, so no extension is computed at runtime. □

Proof of Corollary 2.4 (Exact iterated retraction). Use the dependency-edge direction $\alpha \to \delta$ iff $\delta \in Dep_t(\alpha)$. Here a dependency path is a finite sequence of recorded links, possibly revisiting vertices. We show by induction that X_k consists exactly of arguments having a path of length between 1 and $k+1$ to β. For $k = 0$, this is the definition of $\mathit{Affected}(\beta)$. At the next round, a newly included argument has a dependency in X_k, extending such a path by one edge. Conversely, any path of length at most $k+2$ either has length one or begins with an edge to an argument in X_k, so its starting argument lies in X_{k+1}. Taking the union proves that $\mathit{Affected}^*(\beta)$ is exactly the set with a nonempty recorded path to β. The increasing sequence is bounded by finite $Args_t$ and stabilises within $|Args_t|$ rounds. This argument also covers dependency cycles.

Let $R^* := \{\beta\} \cup \mathit{Affected}^*(\beta)$. For each $\alpha \neq \beta$, membership in R^* is equivalent to having such a path, so removing R^* removes exactly the withdrawn record and its recorded dependents and preserves every other argument. Under the hypotheses of Proposition 2.3, S_t is conflict-free by Lemma 2.2, so its subset $S_t \setminus R^*$ is conflict-free in $AF_t \restriction R^*$, since restricting the attack relation introduces no conflict. The exact-reachability statement holds for any map. □

Proof of Proposition 2.5 (Tractability under structural restrictions). Hard announcements: Each public announcement $[!\psi]$ eliminates worlds where $\neg\psi$ holds, requiring one pass through W: $\mathcal{O}(|W|)$. With k binary propositions, $|W| \leq 2^k$. Over T turns: $\mathcal{O}(T \cdot 2^k)$.

Soft upgrades: A lexicographic upgrade $[\Uparrow \gamma]$ (definition in the *Updates* paragraph above) can be realised by partitioning W on $[\![\gamma]\!]$ and stably sorting on $\leq_i$, in $\mathcal{O}(|W| \log |W|)$ per upgrade. With at most h hypotheses over T turns this bound gives $\mathcal{O}(T \cdot h \cdot |W| \log |W|) = \mathcal{O}(T \cdot h \cdot k \cdot 2^k)$. Awareness expansion adds atomic propositions to $Prop$, potentially doubling $|W|$ each time, so e expansions replace k by $k + e$ in this bound.

Standing and queries: Standing is maintained by the operations (Lemma 2.2) at $\mathcal{O}(1)$ per label update, so no extension is recomputed at runtime. A visited-set traversal for Verify checks each reachable argument and its recorded dependencies at most once. The $\mathit{Affected}^*(\beta)$ traversal does the same over the reverse index, visiting $\{\beta\} \cup \mathit{Affected}^*(\beta)$ and the reverse edges leaving those records. Each traversal therefore takes $\mathcal{O}(|Args| + |Dep|)$ time in the worst case. The reverse traversal takes $\mathcal{O}(1 + |\mathit{Affected}^*(\beta)| + m_\beta)$, where m_β counts the reverse edges inspected. Each visited

identifier is output once. For reference, when the attack graph Att is acyclic, the grounded, complete, preferred, and stable semantics all yield the same unique extension (Dung, 1995), computable in $\mathcal{O}(|Args| + |Att|)$ by topological processing (Dunne, 2007). Without the reverse index, computing $\mathit{Affected}(\beta)$ by scanning the dependency records takes $\mathcal{O}(|Args| + |Dep|)$ time.

In our scenarios, k is 3 (muddy children), 15 (Phase 2 debugging: 7 observable propositions, $mis_monitor$, h_1–h_4 and three causal atoms), and 11 (Phase 3 deliberation: 5 positions, 4 assumptions and 2 libraries). In all three, $|Args| \leq 25$.

General worst cases. Without these restrictions, DEL model-checking with event models is PSPACE-complete (Aucher & Schwarzentruber, 2013), skeptical preferred acceptance in Dung-style AFs is Π_2^{P}-complete (Dunne & Wooldridge, 2009), and preferred-extension enumeration has no known polynomial-delay algorithm. The polynomial bounds above hold under (A1)–(A3) of Proposition 2.5, which exclude these cases. □

A.3 PER-OPERATION UPDATES AND PER-TURN COMPOSITION: APPLY

The Interpreter can extract multiple operation instances from one utterance. Each instance is applied to the four components of the dependency map (Definition 2.1) using Algorithm 1: the epistemic plausibility model $\mathcal{M}$, the argumentation framework AF, the commitment record Cm, and the dependency map Dep. For m instances from an utterance by speaker i, the per-turn update is their sequential composition:

$$\mathcal{D}_t^{(0)} = \mathcal{D}_t, \qquad \mathcal{D}_t^{(j)} = \mathsf{Apply}(op_j, args_j, \mathcal{D}_t^{(j-1)}, i) \quad (1 \leq j \leq m), \qquad \mathcal{D}_{t+1} = \mathcal{D}_t^{(m)}.$$

The Interpreter supplies the identifiers of the records each operation acts on, with unused identifiers for new observations and hypotheses. Instances are applied in the order of Table 3, preserving extraction order within each operation type. The guarantees of Lemma 2.2 hold for instances that meet the preconditions of Table 3. In Algorithm 1, ι is the supplied identifier for a record-creation operation, and $\mathsf{recordArg}(\iota, \ldots)$ registers the supplied content at that ID. The notation α_γ denotes an existing record selected by ID with claim γ, and α_e denotes the record supplying evidence e. The algorithm presents the operation effects, and Table 3 states their preconditions.

State composition. Each operation contributes an intermediate state in the per-turn composition. Record identifiers, standing labels, and dependencies in that state determine the inputs to subsequent operations. RESOLVE changes an existing record's standing without replacing its support links. The query guarantees concern the resulting map: grounding follows its current support links and labels, and retraction reports reverse reachability. Queries leave that map unchanged, and Lemma 2.2 keeps its active set conflict-free.

Subsumption at RESOLVE. The optional $subsumes$ argument and its for-loop (Algorithm 1, RESOLVE case) record a cross-hypothesis dependency at resolution time. When one hypothesis subsumes others, as h_4 unifies h_1 and h_3 at Phase 2 T13, the edge is added at RESOLVE, because it does not exist when the subsumed hypotheses are created.

Algorithm 1 Apply(op, $args$, $\mathcal{D}_t$, i) $\to \mathcal{D}_{t+1}$. Per-operation update rule: one extracted operation instance from speaker i updates the four components ($\mathcal{M}$, AF, Cm, Dep) of the dependency map. Operation arguments $args$ are operation-specific (e.g., HYPOTHESIZE takes the candidate γ and its supporting records $deps$). Preconditions are listed in Table 3.

Require: $\mathcal{D}_t = (\mathcal{M}_t, AF_t, Cm_t, Dep_t)$, speaker $i \in Ags$
$(\mathcal{M}', AF', Cm', Dep') \leftarrow (\mathcal{M}_t, AF_t, Cm_t, Dep_t)$
switch op **of**
  case OBSERVE(ψ, ι)**:**
    $\mathcal{M}' \leftarrow \mathcal{M}_t[!\psi]$
    $\alpha \leftarrow$ recordArg(ι, claim=ψ, src=i); $Args' \leftarrow Args_t \cup \{\alpha\}$
    $st'(\alpha) \leftarrow$ active; $Cm'(i) \leftarrow Cm_t(i) \cup \{\alpha\}$; $Dep'(\alpha) \leftarrow \emptyset$
  case HYPOTHESIZE(γ, $deps$, ι)**:** ▷ $deps \subseteq Args_t$ records supports for γ (Definition A.2)
    $\mathcal{M}' \leftarrow \mathcal{M}_t[\Uparrow \gamma]$; $Prop' \leftarrow Prop_t \cup \{\gamma\}$
    $\alpha \leftarrow$ recordArg(ι, claim=γ, src=i); $Args' \leftarrow Args_t \cup \{\alpha\}$
    $st'(\alpha) \leftarrow$ active; $Cm'(i) \leftarrow Cm_t(i) \cup \{\alpha\}$; $Dep'(\alpha) \leftarrow deps$
  case SUPPORT(γ, e)**:**
    $\mathcal{M}' \leftarrow \mathcal{M}_t[\uparrow \gamma]$ if e is generic, else $\mathcal{M}_t[\Uparrow \gamma]$
    $Dep'(\alpha_\gamma) \leftarrow Dep_t(\alpha_\gamma) \cup \{\alpha_e\}$
  case UNDERMINE(γ, e)**:**
    $\mathcal{M}' \leftarrow \mathcal{M}_t[\Uparrow \neg\gamma]$ if e separates $\neg\gamma$ from γ, else partial elimination (Table 3)
    $Att' \leftarrow Att_t \cup \{(\alpha_e, \alpha_\gamma)\}$
    if $st_t(\alpha_\gamma) =$ active **then**
      $st'(\alpha_\gamma) \leftarrow$ weakened
    end if
  case REVISE(γ)**:**
    $\mathcal{M}' \leftarrow \mathcal{M}_t[!\neg\gamma]$ if γ is explicitly negated; Att' updated per Table 3
    $st'(\alpha_\gamma) \leftarrow$ abandoned
  case EXPAND-AWARENESS(p)**:**
    $Prop' \leftarrow Prop_t \cup \{p\}$; $\mathcal{A}_i' \leftarrow \mathcal{A}_i \cup \{p\}$; refine W on p (Definition A.3)
  case RESOLVE(γ, $subsumes$)**:** ▷ α_γ existing; $subsumes \subseteq Args_t$ optional
    $\mathcal{M}' \leftarrow \mathcal{M}_t[!\gamma]$ if consensual, else commit α_{decide} with dissent recorded
    $st'(\alpha_\gamma) \leftarrow$ resolved
    for $\beta \in subsumes$ **do** ▷ record the supporting argument α_γ
      $Dep'(\beta) \leftarrow Dep_t(\beta) \cup \{\alpha_\gamma\}$
    end for
  case QUESTION(χ)**:**
    $\mathcal{M}' \leftarrow \mathcal{M}_t$; enqueue $(\mathbf{B}_i, \chi)$ on the abductive-problem queue (Definition A.2)
end switch
return $\mathcal{D}_{t+1} \leftarrow (\mathcal{M}', AF', Cm', Dep')$

A.4 Full Epistemic-Operation Table

Table 3 gives the preconditions and DEL realisations of the operations in Section 2. The attack additions of UNDERMINE and REVISE are framework updates in the sense of the dynamic logic of abstract argumentation (Doutre et al., 2014). The consensual and authoritative rows describe two readings of RESOLVE, which share one operation label in the eight-class evaluation.

Table 3: Full epistemic-operation table: DEL realisations and preconditions.

Operation	DEL realisation (full)	Precondition
OBSERVE	$[!\psi]$	Sincere assertion; all atomic propositions in ψ are in $Prop_t$. Later-contradicting evidence triggers REVISE when the contradiction is explicit. An implicit restatement of the same attribute with a new value is recorded as a further OBSERVE.
HYPOTHESIZE	$[\Uparrow \gamma]$	Exists surprising χ; γ satisfies Definition A.2 (1)–(3).
SUPPORT	$[\uparrow \gamma]$, or $[\Uparrow \gamma]$ if the new evidence is γ-specific	$\gamma \in Prop_t$; new evidence compatible with γ; not redundant.
UNDERMINE	$[\Uparrow \neg\gamma]$ when evidence distinguishes $\neg\gamma$ from γ; partial world elimination when a γ-prediction is falsified	α_γ labelled active or weakened; new evidence reduces γ's relative plausibility.
REVISE	$[!\neg\gamma]$, or addition of an attack edge in Att	Speaker explicitly retracts or contradicts γ.
EXPAND-AWARENESS	$\mathcal{A}_i \leftarrow \mathcal{A}_i \cup \{p\}$; refine W on p	The utterance contains an atom $p \notin Prop_t$.
RESOLVE (CONSENSUAL)	$[!\gamma]$ (hard public announcement)	$\alpha_\gamma \in S_t \cap \bigcap_i Cm_t(i)$; no incoming attack on α_γ is recorded.
RESOLVE (AUTHORITATIVE)	$\alpha_{decide} \in Cm(i_{authority})$; dissenting commitments recorded; no hard announcement of γ	$\alpha_\gamma \in S_t$ with no recorded incoming attack; the speaker has decision authority and dissent is recorded.
QUESTION	no DEL update; add $(\mathbf{B}_i, \chi)$ to the abductive-problem queue	Utterance is interrogative or flags an unexplained observation.

B Running Examples

B.1 Phase 1: Muddy Children

Figure 5 traces the model through the three rounds of the muddy children puzzle (models $\mathcal{M}_0$–$\mathcal{M}_3$). The per-turn natural-language transcript follows.

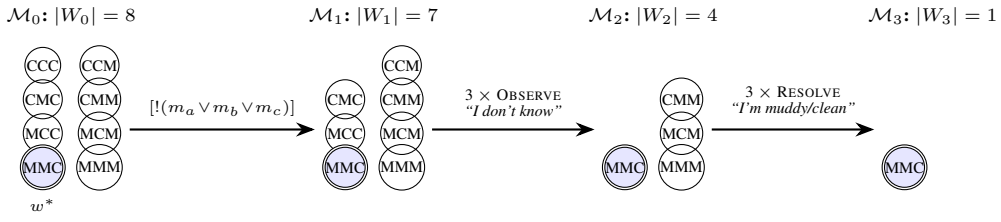

Figure 5: Epistemic-model (DEL) trajectory through the muddy children puzzle: the world set W_t after each round (8, 7, 4, 1 worlds). The actual world $w^*{=}MMC$ is double-circled. Eliminated worlds are dropped and live worlds keep their positions.

We present the muddy children puzzle as a conversation with the full epistemic model traced at each turn. Three children, Alice (a), Bob (b), Carol (c), have been playing outside. Alice and Bob are muddy, and Carol is clean. Each child can see the others' foreheads but not their own.

Propositions. m_a, m_b, m_c ("child x is muddy"). The actual world is $w^* = (m_a, m_b, \neg m_c)$, abbreviated MMC.

Initial model $\mathcal{M}_0$. $W_0 = \{MMM, MMC, MCM, MCC, CMM, CMC, CCM, CCC\}$. Each child i has $w \sim_i w'$ iff w, w' agree on m_j for all $j \neq i$. For Alice (sees Bob=M, Carol=C): $\sim_a$ groups $\{MMC, CMC\}$, $\{MMM, CMM\}$, $\{MCM, CCM\}$, $\{MCC, CCC\}$.

> **Turn 0: Father:** Children, I can see that at least one of you has mud on your forehead.

Operation: OBSERVE (public announcement $\psi_0 = m_a \vee m_b \vee m_c$).
Update: Eliminate CCC. **Model $\mathcal{M}_1$:** $|W_1| = 7$.
Common knowledge: $\mathbf{C}_{abc}(m_a \vee m_b \vee m_c)$.
Individual knowledge: No child knows their own status.

> **Turn 1a: Alice:** I don't know.
>
> **Turn 1b: Bob:** I don't know.
>
> **Turn 1c: Carol:** I don't know.

Operations: Three OBSERVE ("I don't know" is a public announcement).
Reasoning: If any child saw two clean faces, they would know they are muddy (since at least one must be). Nobody knew, so all single-muddy worlds are eliminated.
Model $\mathcal{M}_2$: $W_2 = \{MMM, MMC, MCM, CMM\}$, $|W_2| = 4$.
Common knowledge: At least two children are muddy.

Individual knowledge after $\mathcal{M}_2$:

- Alice sees Bob=M, Carol=C. Consistent worlds in W_2: $\{MMC\}$. *Alice knows she is muddy.*
- Bob sees Alice=M, Carol=C. Consistent worlds in W_2: $\{MMC\}$. *Bob knows he is muddy.*
- Carol sees Alice=M, Bob=M. Consistent worlds in W_2: $\{MMC, MMM\}$. Carol doesn't know yet, but can deduce from Alice's and Bob's round-2 answers.

> **Turn 2a: Alice:** Yes, I'm muddy.
>
> **Turn 2b: Bob:** Yes, I'm muddy.
>
> **Turn 2c: Carol:** Yes, I'm clean.

Operations: Three RESOLVE.
Model $\mathcal{M}_3$: $W_3 = \{MMC\}$. Complete knowledge. $\mathbf{C}_{abc}(m_a \wedge m_b \wedge \neg m_c)$.

Verification. At each step, the system's model must match $\mathcal{M}_0 \to \mathcal{M}_1 \to \mathcal{M}_2 \to \mathcal{M}_3$. The Interpreter must label Turn 0 and Turns 1a–c as OBSERVE and Turns 2a–c as RESOLVE.

B.2 PHASE 2: SYSTEM DEBUGGING

Three engineers debug a cascading failure. They are Alice (backend engineer; can see Auth Service and Payment Service application logs), Bob (infrastructure engineer; can see Payment Service metrics, Redis metrics, and external service dashboards) and Carol (on-call SRE; can see the alerting dashboard and customer complaint tickets but needs Alice or Bob to dig into specifics).

Ground truth (known to us, not to the engineers). Yesterday's deployment introduced a bug in Auth Service: the token refresh path computes expiry from Unix epoch instead of current time, causing refreshed tokens to be immediately expired. The frontend retries expired tokens, generating 3× normal traffic to Auth. Every auth request hits the shared Redis cluster for session lookup, so the retry storm exhausts Redis's connection pool. Payment Service uses the same Redis cluster

for rate-limit counters. With Redis down, rate-limit checks fail and requests go to Stripe unthrottled, triggering Stripe's 429 rate-limit responses. Separately, a stale monitoring rule maps Redis connection errors to "database health degradation" alerts, creating a red herring.

What makes this Phase 2. Unlike Phase 1, the hypothesis space is not given upfront: hypotheses are *constructed* during the conversation through abductive reasoning. The conversation takes a wrong turn (initially attributing auth failures to Redis), evidence requires domain-specific interpretation (error code types), and a causal direction reversal is the key epistemic shift. The monitoring miscategorisation tests awareness expansion.

Formal model specification. We define the symbolic model that the engine maintains alongside the conversation. Unlike Phase 1 where the world set is fixed at 2^k from the start, Phase 2 extends the model dynamically as hypotheses are generated and awareness expands. The abductive mechanism implements Definition A.2: the symbolic engine detects surprising observations ($\mathcal{M}, w \not\models \mathbf{B}_i\chi$), the LLM generates candidate hypotheses γ, and the engine verifies and integrates them via plausibility upgrade.

Agents: $Ags = \{a, b, c\}$ (Alice, Bob, Carol).

Evidence records: the evidence of each SUPPORT or UNDERMINE is registered as a record α_e. At T7 the evidence is o_9, so $Att_7 = \{(o_9, h_2)\}$.

Propositions: the observable propositions are $p_{auth}, p_{pay}, p_{db}, p_{tok}, p_{traffic}, p_{stripe429}, p_{redis}$ (auth failures, payment failures, database alert, token errors, anomalous traffic, Stripe 429s, Redis issues). These monitored signals form $Prop_0$. T1–T3 announce which of them hold.

Awareness: all three engineers hear every utterance, so each proposition in $Prop_t$ is in every $\mathcal{A}_i$, and each EXPAND-AWARENESS is applied to every agent's awareness set. Crucially, $mis_monitor$ (monitoring miscategorises Redis as "database") is *not* in any $\mathcal{A}_i$ until T4.

Hypothesis propositions, together with any new atoms they name, are added to $Prop$ as they are generated:

- T5: $h_1 \coloneqq (p_{redis} \to rate_bypass \to p_{stripe429})$ added to $Prop$.
- T6: $h_2 \coloneqq (p_{redis} \to p_{auth})$ added.
- T11: $h_3 \coloneqq (tok_bug \to retry_loop \to p_{traffic})$ added.
- T12: $h_4 \coloneqq (tok_bug \to retry_loop \to p_{redis})$ added.

Plausibility-model trajectory and awareness expansion at T4 are tracked in the per-turn walkthrough below. The key shifts are that h_1-worlds are upgraded after T5, h_2-worlds are downgraded after T7 (UNDERMINE) and eliminated at T9 (REVISE), $mis_monitor$ enters $\mathcal{A}_i$ at T4, and after T13 only h_4-worlds remain, with h_1 and h_3 linked to h_4 in the dependency map.

> **T1: Carol:** P1 incident. Three alerts firing: auth failure rate up, payment failure rate up, database health degradation. Customer complaints started around 2:15am.

Operations: OBSERVE $\times 4$ (three alerts and the onset of complaints), QUESTION (implicit: "what is causing the failures?").

Why these classifications: Carol is reporting factual observations from her dashboard, not proposing explanations. Each alert is a separate observation. The implicit question is what opens the deliberation.

Model state:

- Observations: o_1: auth failure rate elevated; o_2: payment failure rate elevated; o_3: "database health degradation" alert; o_4: customer complaints from ∼2:15am.
- Hypotheses: none yet.
- Open questions: What is causing the three alert types? Are they related or independent?
- Causal chain: unknown.

Symbolic model $\mathcal{M}_1$: $Prop_1 = Prop_0$. Public announcements of o_1, o_2, o_3, o_4 eliminate worlds inconsistent with these observations. o_4 restates $p_{auth} \vee p_{pay}$ from the customer side. $\mathcal{A}_i = Prop_1$

for all i. Note: $mis_monitor \notin \mathcal{A}_i$ (no agent is yet aware that "database" alert may be mislabelled). No hypotheses in model; no plausibility distinctions beyond observations. All agents share the same epistemic state: $\mathbf{K}_a\mathbf{K}_b\mathbf{K}_c(o_1 \wedge o_2 \wedge o_3 \wedge o_4)$.

> **T2: Alice:** I'm seeing 401 Unauthorized spikes in auth logs starting around 2am. Tokens being rejected as expired. But I also see our auth request volume is way up, about 3× normal. That's strange. More users shouldn't be logging in at 2am.

Operations: OBSERVE (two new facts), QUESTION ("why is traffic 3×?").

Why: Alice reports specific observations from her logs. The 401 error code, the "token expired" detail, and the 3× traffic volume are all factual reports. Her remark that "more users shouldn't be logging in at 2am" signals a surprising observation that doesn't yet have an explanation. This is an implicit abductive problem that will drive later hypothesis generation.

Model state:

- New observations: o_5: 401 "token expired" errors from ∼2am; o_6: auth traffic 3× normal at 2am.
- New open question: Why is auth traffic 3× at 2am? (Anomalous, doesn't match normal usage.)

Symbolic model $\mathcal{M}_2$: $Prop_2 = Prop_1$. Public announcements of o_5, o_6. Worlds where auth errors are not "token expired" are eliminated; worlds where traffic is normal are eliminated. $\mathcal{M}_2, w^* \models \mathbf{K}_a(o_5 \wedge o_6)$; after public announcement, $\mathbf{C}_{abc}(o_5 \wedge o_6)$. *Abductive problem detected (surprise check):* just before o_6 is announced, $w^* \models o_6$ but no agent believes o_6, since nothing in the model predicts elevated traffic at 2am. So o_6 poses an abductive problem in the sense of Definition A.2 (Velázquez-Quesada et al., 2013). The announcement makes o_6 common knowledge but leaves it unexplained. No candidate γ has been proposed yet, so the problem remains open until T11.

> **T3: Bob:** Payment side, I see Stripe returning 429s. We're sending them way more requests than normal. And I can confirm I see the database alert too, Redis connection timeouts from Payment Service.

Operations: OBSERVE ×2 (Stripe 429s with elevated request rate, Redis timeouts).

Why: Pure observation from Bob's infrastructure perspective. The Redis timeouts are particularly important because they will later become central to the causal reasoning. At this point, nobody has proposed a hypothesis yet. The group is still gathering data.

Model state:

- New observations: o_7: Stripe 429 (rate-limit) errors; o_8: Redis connection timeouts from Payment side.
- Note: We now have observations from all three engineers. The picture so far: auth tokens failing, payment requests failing, Redis having connection issues. The question is how these relate.

Symbolic model $\mathcal{M}_3$: $Prop_3 = Prop_2$. Public announcements of o_7, o_8. $\mathbf{C}_{abc}(o_7 \wedge o_8)$. No hypotheses yet. The model contains 8 observations but no causal structure. The world set W_3 still contains all worlds consistent with o_1–o_8, including worlds where the symptoms are independent, worlds where Redis causes everything, worlds where auth causes everything, etc. No plausibility distinctions among these.

> **T4: Carol:** OK, so the database alert is about Redis, not the primary DB. Still concerning. Bob, are you seeing Redis issues from Payment's side?

Operations: EXPAND-AWARENESS (the "database" alert is really a Redis alert), QUESTION.

Why EXPAND-AWARENESS: This is the first moment where a proposition that was previously outside the group's reasoning enters the conversation. Everyone initially took the "database health degradation" alert at face value and reasoned about a database problem. Carol's reclassification introduces the proposition "the monitoring rule conflates Redis with database," which was not previously in anyone's awareness set. In the formal framework, this adds new formulas to $\mathcal{A}_i$ for all agents, expanding the set of worlds they can distinguish.

Model state:

- Observation o_3 reclassified: alert labelled "database" is actually about Redis connection errors.
- Awareness expansion: the proposition $mis_monitor$ has entered the group's awareness. Before this turn, the group was reasoning about a possible primary database problem. That possibility is now distinguishable from a mislabelled Redis alert.

Symbolic model $\mathcal{M}_4$: *Awareness expansion:* $mis_monitor :=$ "monitoring conflates Redis with database." Before T4: $mis_monitor \notin \mathcal{A}_i$ for all i. After T4: $mis_monitor \in \mathcal{A}_i$ for all i. $Prop_4 = Prop_3 \cup \{mis_monitor\}$. *World space restructuring:* Before expansion, agents could not distinguish $w_{db_problem}$ (primary DB failing) from $w_{redis_mislabel}$ (Redis failing, mislabelled as DB). After expansion, these are distinct worlds, and Bob's OBSERVE at T5 (connection pool exhausted) supports the Redis reading. *Effect:* o_3 can now be read either as a primary-database problem or as a mislabelled Redis alert.

> **T5: Bob:** Yes. Connection pool exhausted. We can't get connections to Redis. That's why our rate-limit checks are failing, we store rate-limit counters in Redis. When we can't check the counter, our code falls through and sends the request to Stripe anyway. That would explain the 429s, we're hitting Stripe without rate limiting.

Operations: OBSERVE (pool exhaustion details), HYPOTHESIZE (h_1).

Why HYPOTHESIZE: Bob doesn't just report facts: he constructs an *explanatory chain*: Redis pool exhausted → rate-limit checks fail → requests go to Stripe unthrottled → 429s. This is an abductive step: Bob has a surprising observation (o_7: Stripe 429s) and proposes a hypothesis (h_1) that explains it. In the formal framework, this is an abductive solution integrated at the belief level via plausibility upgrade.

Model state:

- New observation $o_{8'}$: Redis connection pool exhausted (claim p_{redis}).
- h_1: Redis pool exhaustion → rate-limit check failure → unthrottled Stripe requests → 429s. Status: **active**. Plausibility: **high**. Supported by: o_7, o_8.
- This is the first causal hypothesis. It explains the Stripe failures but does not yet explain the auth failures or the Redis exhaustion itself.

Symbolic model $\mathcal{M}_5$: *Abductive cycle for o_7:*

1. *Surprise check (engine):* just before o_7 was announced at T3, $w^* \models o_7$ but Bob did not believe o_7, since no hypothesis predicted Stripe rate-limit errors. So o_7 poses an abductive problem $(\mathbf{B}_b, o_7)$.
2. *Hypothesis generation (LLM):* The LLM Interpreter classifies Bob's utterance as HYPOTHESIZE and extracts the candidate: $\gamma := h_1 := (p_{redis} \rightarrow rate_bypass \rightarrow p_{stripe429})$.
3. *Verification (engine):* (a) Consistency: h_1 is consistent with $\mathcal{M}_4$ (no contradictions). (b) Explanatory: after plausibility upgrade with h_1, agent would believe o_7 (Redis being down plus h_1 entails Stripe 429s). (c) Non-trivial: $h_1 \neq o_7$ and h_1 is not already believed. All checks pass.
4. *Integration (engine):* $Prop_5 = Prop_4 \cup \{h_1, rate_bypass\}$. Plausibility upgrade: h_1-worlds become more plausible for all agents (public communication). For all i, in W_5: $w \models h_1 \implies w \prec_i w'$ for $w' \models \neg h_1$ (given consistency with o_7, o_8).

Epistemic status: $\mathbf{B}_i h_1$ for all i (believed, not known). $\neg \mathbf{K}_i h_1$ because alternative explanations for o_7 have not been ruled out. *Observations explained:* h_1 explains o_7 (Stripe 429s), taking o_8 (Redis timeouts) as its premise. Does **not** explain o_5 (token errors) or o_6 (3× traffic). The abductive problem for o_6 (flagged at T2) remains open.

> **T6: Carol:** So the chain might be: Redis is sick → Payment loses rate limiting → Stripe gets hammered. And separately, Redis being sick → Auth has problems too? Alice, does Auth use Redis?

Operations: SUPPORT (h_1, adopted by Carol as the working picture), HYPOTHESIZE (h_2: Redis → Auth failures), QUESTION.

Why: Carol synthesises the current understanding and extends it. By proposing h_2, she is constructing a unified picture: Redis is the single root cause, and both the payment and auth failures are downstream effects. This is a natural abductive move: explaining all symptoms with one cause is more parsimonious.

Model state:

- h_2: Redis failure → Auth failures (via shared Redis cluster for session caching).
  Status: **active**. Plausibility: **medium**. Supported by: o_8. Basis: if Auth shares Redis, its failures could cascade.
- Leading picture at this point: *Redis is the single root cause*. Everything downstream.
- This picture is **wrong**, and the next turns will overturn it.

Symbolic model $\mathcal{M}_6$: *Second abductive step:* $h_2 := (p_{redis} \rightarrow p_{auth})$ answers the abductive problem posed by o_1, which no agent believed before its announcement at T1. $Prop_6 = Prop_5 \cup \{h_2\}$. *Plausibility upgrade:* h_2-worlds upgraded to plausible (medium). For all i: worlds where $h_1 \wedge h_2$ both hold (single root cause = Redis) are now the most plausible: $w_{h_1 \wedge h_2} \prec_i w_{\neg h_1 \wedge h_2} \prec_i w_{h_1 \wedge \neg h_2} \prec_i w_{\neg h_1 \wedge \neg h_2}$. SUPPORT registers Carol's restatement as $\alpha_e \in Dep(h_1)$, and $[\uparrow h_1]$ leaves this order unchanged, since the most plausible h_1-world $w_{h_1 \wedge h_2}$ is already minimal. *Epistemic status:* $\mathbf{B}_i h_1$ (high), $\mathbf{B}_i h_2$ (medium). The "single root cause" picture is a belief, not knowledge. *Observations explained by* $h_1 \wedge h_2$*:* o_1 (auth failures), o_2 (payment failures), o_7 (Stripe 429s), o_8 (Redis timeouts). Remaining unexplained: o_6 (3× traffic).

> **T7: Alice:** Yes, Auth uses the same Redis cluster for session caching. If Redis is down, token validation would fail because we can't look up the session. That would explain the 401s... but wait, the 401s I'm seeing are specifically 'token expired,' not 'session lookup failed.' Those are different error codes. The tokens are being rejected because their expiry timestamps are in the past, not because Redis is unavailable.

Operations: OBSERVE (o_9: error code is "token expired," not "session lookup failed"), UNDERMINE (h_2).

Why UNDERMINE and not just OBSERVE: Alice initially supports h_2 ("If Redis is down, token validation would fail... that would explain the 401s"), but then notices a *discrepancy*: the specific error code is inconsistent with the Redis-causes-auth hypothesis. If Redis were the cause, the error would be a session lookup failure (503), not a token expiry (401). This observation doesn't just add new data. It directly decreases the plausibility of h_2. In the formal framework, worlds where h_2 is true become less plausible because they predict a different error code than what was observed.

This is the most important diagnostic moment in the conversation. Alice's mid-utterance correction ("but wait...") shows real-time epistemic revision: she starts by supporting the hypothesis and then undermines it within the same turn.

Model state:

- h_2 status: **active** → **weakened**. The evidence (o_9) is inconsistent with the predicted error type.
- New observation o_9: Auth errors are 401 "token expired," not 503 "service unavailable."
- The single-root-cause picture (Redis causes everything) is now under strain.

Symbolic model $\mathcal{M}_7$: *Observation:* o_9 is a further OBSERVE record with claim p_{tok}: the error is "token expired", not a 503. *Undermine* h_2*:* h_2 predicts session-lookup failures rather than token expiry, and o_9 makes this contrast explicit, so UNDERMINE applies $[\Uparrow \neg h_2]$ (Table 3). *Plausibility downgrade:* for all i, h_2-worlds become less plausible: $w_{\neg h_2} \prec_i w_{h_2}$ (reversing the T6 upgrade). *Epistemic status:* $\mathbf{B}_i \neg h_2$ (agents now believe h_2 is false), but $\neg \mathbf{K}_i \neg h_2$ (h_2 is not yet eliminated, Alice hasn't explicitly ruled it out). The ordering is: $w_{h_1 \wedge \neg h_2} \prec_i w_{\neg h_1 \wedge \neg h_2} \prec_i w_{h_1 \wedge h_2} \prec_i w_{\neg h_1 \wedge h_2}$. *Key formal point:* here UNDERMINE downgrades the h_2-worlds without eliminating them. The remaining h_2-worlds are epistemically possible but no longer believed: knowledge corresponds to world elimination, belief to plausibility reordering.

> **T8: Carol:** Hmm. So the auth failures might not be caused by Redis after all?

T9: Alice: I don't think so. The error type is wrong. If Redis were down, Auth would return a 503 Service Unavailable, not a 401 with 'token expired.' I'm seeing 401s. So the token expiry issue is a different problem from the Redis problem.

Operations (T8): QUESTION (Carol seeks confirmation of the implication).
Operations (T9): REVISE (Alice explicitly abandons h_2).

Why REVISE: This is a genuine belief revision, not just weakening. Alice explicitly states that the token expiry issue is "a different problem from the Redis problem." This separates what was previously thought to be a single-cause situation into (at least) two independent problems. In the formal framework, the hard announcement $!\neg h_2$ removes the h_2-worlds.

Model state:

- h_2: **weakened** $\rightarrow$ **abandoned**. Auth failures are not caused by Redis.
- The group's picture has fundamentally changed: there are at least two independent problems, not one.
- Open question (newly urgent): What *is* causing the token expiry errors?

Symbolic model $\mathcal{M}_9$: *Revision:* Alice explicitly negates h_2, so REVISE applies the hard announcement $!\neg h_2$ (Table 3), which eliminates every h_2-world. After T9, $\mathbf{K}_i \neg h_2$ holds for all i. The group now *knows* $\neg h_2$, not merely believes it. *Updated plausibility:* $w_{h_1 \wedge \neg h_2} \prec_i w_{\neg h_1 \wedge \neg h_2}$. No active link between Auth and Redis remains. *Observations now unexplained:* With h_2 abandoned, o_1 (auth failures), o_5 (token errors), and o_6 (3$\times$ traffic) have no causal explanation, and o_6 remains the open abductive problem flagged at T2. *Key formal distinction from T7:* At T7, h_2 was *undermined* (plausibility downgraded but worlds retained). At T9, h_2 is *revised* (worlds eliminated). This is the difference between UNDERMINE and REVISE in the formal framework.

T10: Bob: But then why is auth traffic 3$\times$ normal? If the token expiry issue is independent of Redis, what's generating all that auth traffic?

Operations: QUESTION.

Why this matters: Bob identifies that the unexplained 3$\times$ traffic (o_6) is now a critical clue. If auth failures aren't caused by Redis, then the elevated traffic isn't explained by the current model. This question drives the next abductive step. In the formal framework, this is a recognition of an abductive problem: there exists a surprising observation (o_6) that the current set of hypotheses cannot explain.

Symbolic model $\mathcal{M}_{10}$: No structural change to W or plausibility orderings. The QUESTION keeps o_6 on the abductive-problem queue: $(\mathbf{B}_i, o_6) \in AbdProb_{10}$. The model records that o_6 is surprising and unexplained.

T11: Alice: Could be the frontend retrying. When a user gets a 'token expired' error, the frontend automatically tries to refresh the token. If the new token is also expired, it retries again. That's a retry loop. So the token bug generates its own amplified traffic.

Operations: HYPOTHESIZE (h_3).

Why: Alice proposes a new hypothesis to explain the anomalous traffic observation. This is a classic abductive step: the surprising fact (o_6: 3$\times$ traffic) triggers the generation of an explanatory hypothesis (h_3: a feedback loop where the bug amplifies its own traffic). Note that h_3 involves domain knowledge about the frontend's retry behaviour, something the group becomes aware of through Alice's contribution.

Model state:

- h_3: Token bug $\rightarrow$ frontend retry loop $\rightarrow$ 3$\times$ traffic amplification.
  Status: **active**. Plausibility: **medium**. Explains o_6.
- At this point, the group has two separate explanations: h_1 (Redis $\rightarrow$ Stripe problems) and h_3 (token bug $\rightarrow$ traffic amplification). They haven't yet connected these.

Symbolic model $\mathcal{M}_{11}$: *Abductive cycle for o_6 (open since T2):*

1. *Surprise check:* o_6 (3× traffic) was flagged at T2 as an observation not predicted by any hypothesis. After h_2's abandonment at T9, o_6 still has no explanatory hypothesis in the current model. Abductive problem $(\mathbf{B}_i, o_6)$ remains open.
2. *Hypothesis generation (LLM):* Alice's utterance classified as HYPOTHESIZE. Candidate extracted: $\gamma \coloneqq h_3 \coloneqq (tok_bug \rightarrow retry_loop \rightarrow p_{traffic})$.
3. *Verification (engine):* (a) Consistency: h_3 is consistent with $\mathcal{M}_{10}$. (b) Explanatory: if tok_bug is true, the retry loop produces elevated traffic, so $h_3 \wedge tok_bug$ entails o_6. (c) Non-trivial: $h_3 \neq o_6$. All checks pass.
4. *Integration:* $Prop_{11} = Prop_{10} \cup \{h_3, tok_bug, retry_loop\}$. Plausibility upgrade: h_3-worlds upgraded. For all i: $w_{h_3} \prec_i w_{\neg h_3}$.

New atoms: tok_bug (a deployment bug causes token expiry) and $retry_loop$ (frontend retries create amplification) enter $Prop$ with h_3 (step 4). *Epistemic status:* $\mathbf{B}_i h_3$ (medium plausibility, plausible but unconfirmed). The abductive problem for o_6 is now **closed** (explained by h_3). *Model topology:* Two independent causal chains coexist: h_1 (Redis → Stripe) and h_3 (token bug → traffic). No connection between them yet. o_6 is now explained; o_5 (token errors) is partially explained.

T12: Carol: Wait, so the retry storm from the token bug could be what's exhausting Redis? Not Redis causing the auth problem, but the auth problem causing the Redis overload?

Operations: HYPOTHESIZE + REVISE. **This is the critical turn: causal direction reversal.**

Why this is the key epistemic shift: Carol makes the decisive connection. She realises that h_3 (token bug causes retry storm) can be linked to h_1 (Redis exhaustion causes Stripe failures) by *reversing the causal direction between Auth and Redis*. The group had previously assumed Redis → Auth (hypothesis h_2, now abandoned). Carol proposes Auth → Redis: the retry storm is what overwhelms Redis, not the other way around.

This is both a HYPOTHESIZE (introducing the new causal link: retries exhaust Redis) and a REVISE of h_2: Carol explicitly contradicts Redis → Auth, so the engine adds the attack (h_4, h_2) and h_2 stays abandoned. The result is again a single causal chain, with a structure opposite to the abandoned h_2.

Model state:

- h_4: Token bug → retry storm → Redis connection pool exhaustion.
  Status: **active**. Plausibility: **medium**. Links h_3 to h_1.
- **Key epistemic shift recorded**: Causal direction between Auth and Redis has reversed. Previously: Redis failure was thought to cause auth problems (h_2). Now: auth problems (the token bug and resulting retry storm) are proposed to cause Redis failure (h_4).
- Emerging unified chain: token bug → retries → Redis exhaustion → rate-limit bypass → Stripe 429s.

Symbolic model $\mathcal{M}_{12}$: *Hypothesis:* $h_4 \coloneqq (tok_bug \rightarrow retry_loop \rightarrow p_{redis})$. It answers the problem posed by o_8, which no agent believed before its announcement at T3. Note the causal direction: Auth → Redis, opposite of the abandoned h_2 (Redis → Auth). $Prop_{12} = Prop_{11} \cup \{h_4\}$. *Plausibility upgrade:* h_4-worlds upgraded to plausible. The combined hypothesis $h_1 \wedge h_3 \wedge h_4$ now forms a unified causal chain and is more plausible than the independent-problems picture. For all i: every h_4-world precedes every $\neg h_4$-world, and $w_{h_1 \wedge h_3 \wedge h_4}$ is the most plausible. *Epistemic status:* $\mathbf{B}_i h_4$ (believed, not known, awaiting mechanistic confirmation). *Structural revision:* The dependency map records h_4 with $h_3 \in Dep(h_4)$, so the recorded link between Auth and Redis now runs from Auth to Redis. The model records this as a key epistemic shift with provenance: h_2 (Redis → Auth, T6, abandoned T9 due to o_9) → h_4 (Auth → Redis, T12, supported by h_3). *Unification:* If h_4 is accepted, then $h_1 \wedge h_3 \wedge h_4$ form a single causal chain explaining *all* observations o_1–o_9 and $o_{8'}$.

T13: Alice: That's... actually plausible. If auth traffic is 3×, and every auth request hits Redis for session lookup, the Redis connection pool could be overwhelmed. So the chain would be: token bug → retry storm → Redis exhaustion. And then Redis exhaustion → Payment rate-limit bypass → Stripe 429s. The whole thing cascades from the token bug.

Operations: SUPPORT (confirming h_4 with mechanistic reasoning), RESOLVE (the unified causal chain is accepted).

Why RESOLVE: Alice provides the mechanistic argument that makes h_4 convincing: $3\times$ traffic, every request hitting Redis, pool exhaustion is the expected consequence. She then states the complete unified chain explicitly. At this point, the hypothesis is elevated from tentative belief to high-confidence accepted explanation.

Symbolic model $\mathcal{M}_{13}$ (final): *Support + Resolve:* Alice's mechanistic argument provides sufficient evidence to elevate h_4 from belief to knowledge. It is registered as $\alpha_e \in Dep(h_4)$. RESOLVE announces $!h_4$, which eliminates the $\neg h_4$-worlds, so $\mathbf{K}_i h_4$ holds for all i after T13. It also records h_1 and h_3 as subsumed by h_4 (Appendix A.3), linking the unified chain $h_{unified} = h_1 \wedge h_3 \wedge h_4$ through the dependency map. *Final plausibility:* The model converges to a single maximally plausible world class: $W_{13}^{max} = \{w : w \models h_{unified}\}$. The $\neg h_4$-worlds are eliminated. *Hypothesis lifecycle summary:*

- h_1: $\emptyset \xrightarrow{\text{T5, Hyp}}$ active. At T13 it gains a dependency link to h_4 and joins the unified chain (correct mechanism).
- h_2: $\emptyset \xrightarrow{\text{T6, Hyp}}$ active $\xrightarrow{\text{T7, Und}}$ weakened $\xrightarrow{\text{T9, Rev}}$ abandoned. Disproved by o_9.
- h_3: $\emptyset \xrightarrow{\text{T11, Hyp}}$ active. At T13 it gains a dependency link to h_4 and joins the unified chain.
- h_4: $\emptyset \xrightarrow{\text{T12, Hyp}}$ active $\xrightarrow{\text{T13, Sup+Res}}$ resolved. Key epistemic shift: causal reversal.

Awareness set (final): $\mathcal{A}_i = Prop_{13}$ for all i: the observable propositions, $mis_monitor$, h_1–h_4 and their atoms. *Knowledge (final):* $\mathbf{C}_{abc}(h_4 \wedge \neg h_2)$, and $h_{unified}$ holds in every most plausible world.

Final model state. The unified causal chain (matching the ground truth of Appendix B.2's opening): *deployment token bug → expired tokens → frontend retries → 3× auth traffic → Redis pool exhaustion → Payment rate-limit bypass → unthrottled Stripe requests → 429s*, with the "database health" alert separately flagged at T4 as a possible monitoring miscategorisation. Hypothesis lifecycles are listed above. The per-turn walkthrough records, at each step, the formal grounds (operation, plausibility shift, dependency edge) for the system's tracking, including the two key epistemic shifts at T7–T9 (error-code evidence kills h_2) and T12 (causal direction reversal).

B.3 PHASE 3: ARCHITECTURE DELIBERATION

A team decides how to add real-time collaboration to their document editor. Its members are Alice (product lead; feature scope and timeline), Bob (senior backend engineer; architecture and long-term maintainability) and Carol (frontend engineer; editor integration and user experience).

What makes this Phase 3. Unlike Phases 1–2, there is no objectively correct answer. The conversation produces a *decision* through deliberation, not a *discovery* of pre-existing fact. The core epistemic primitive shifts from knowledge/belief to *commitments* (public, retractable assertions about what the team will do). The conversation involves genuine trade-offs, persistent disagreement, and a decision made under authority that explicitly records dissent and conditions for re-evaluation.

Model notation. We track the model using IBIS-style elements: *Issues* (I_n, questions under deliberation), *Positions* (P_n, proposed answers), *Arguments* (pro/con, attributed to speakers), and *Decisions* (resolved issues with provenance). We also track *Assumptions* (a_n): premises that decisions rest on, retractable when new evidence undermines them.

Classification results (cross-model). Table 4 reports per-model F1 and EM on Phase 3, under the Definitions condition only (matching the cross-scenario comparison in Table 2). Phase 3 reaches comparable or higher accuracy than Phase 2 on all four models, despite the longer 19-turn structure and the epistemic-to-deliberative shift (Section 4).

Formal model specification. The symbolic engine maintains a dependency map $(\mathcal{M}, AF, Cm, Dep)$ as defined in Definition 2.1. Phase-3-specific instantiation:

Table 4: Phase 3 (deliberation, 19 turns) classification, Definitions condition only. EM is exact match in percent, over 5 runs per cell.

	Overall		Key shifts (4 turns)	
Interpreter	F1	EM	F1	EM
Claude Sonnet 4	**0.85**	63	**0.92**	**75**
GPT-4o	0.80	**65**	0.83	50
Qwen2.5-7B	0.75	61	**0.92**	**75**
Llama-3.1-8B	0.66	33	0.59	20

Agents: $Ags = \{a, b, c\}$ (Alice, Bob, Carol), with roles: a = product lead (decision authority), b = senior backend (architecture), c = frontend (editor integration).

Arguments: each position P_n is itself a record in $Args$ with claim "adopt P_n", so attacks on positions are attacks between records. The evidence of each SUPPORT or UNDERMINE is registered as a new record α_e under the identifier the Interpreter supplies (Appendix A.3), committed by the speaker, with the premises the utterance cites as its dependencies. The displayed records carry tuples $(claim, speaker, turn, type)$. Within Dep and $Affected^*$ below, premise labels such as a_i and $timeline$ abbreviate the identifiers of premise records, registered, with their propositions in $Prop$, by the first record that cites them. In truth assertions they denote the corresponding propositions. The records that rest on a withdrawn premise β are reported by $Affected^*(\beta)$ (Definition 2.3). Each HYPOTHESIZE adds its position to $Prop$, and the decision record of an authoritative RESOLVE lists the premises it rests on.

Awareness: $\mathcal{A}_i$ tracks the solution space visible to each agent. All three agents hear every utterance, so each EXPAND-AWARENESS is applied to every agent's awareness set. Initially, only "implement Operational Transformation (OT)" and "implement Conflict-free Replicated Data Type (CRDT)" are in $\mathcal{A}_i$. Library-based approaches enter at T5 via awareness expansion.

T1: Alice: We need real-time collaboration. Users should see each other's edits live, like Google Docs. Ship in six weeks. How do we build it?

Operations: QUESTION (opens root issue), OBSERVE (constraints stated).

Why: Alice defines the problem and constraints. In IBIS terms, this opens the root issue. The six-week timeline and the real-time requirement (opened as issue I_1) are constraints that will shape the evaluation of all positions.

Model:

- Issue I_1: "How to implement real-time collaboration?" Status: **open**.
- Constraints: 6-week deadline; users must see changes in real time.

Symbolic model AF_1: $Args_1 = \{timeline\}$, the premise record of the six-week deadline registered by OBSERVE. $Att_1 = \emptyset$. $Cm(a) = \{timeline\}$, $Cm(b) = Cm(c) = \emptyset$. The QUESTION opens issue I_1, whose requirement is real-time collaboration. $Prop_0 = \{timeline, P_1, P_2\}$: the deadline and the two approaches the team starts from.

T2: Bob: Two established approaches: Operational Transformation, that's what Google Docs uses, and CRDTs, which is what Figma and newer tools use. OT needs a central server for coordination. CRDTs are peer-to-peer capable but more complex to implement.

T3: Alice: What about the six-week timeline? Can we ship either one?

T4: Bob: Implementing either from scratch in six weeks is risky. OT's transformation functions are subtle and buggy. CRDTs have complex data structures.

T5: Carol: What about using a library? Yjs is a mature CRDT library. ShareDB for OT.

Operations (T2): OBSERVE $\times 2$ (two approaches exist), which registers the position records P_1 and P_2.
Operations (T3): QUESTION (feasibility sub-issue under timeline constraint).
Operations (T4): UNDERMINE (P_1 and P_2: both risky within timeline).
Operations (T5): EXPAND-AWARENESS + HYPOTHESIZE (library approach was not previously considered).

Why T5 is EXPAND-AWARENESS: Before Carol's suggestion, the discussion was framed as "which algorithm should we implement?" Carol introduces a new dimension, using existing libraries, that was outside the group's consideration. This is analogous to awareness expansion in the formal framework: a new set of propositions (about specific libraries, their maturity, their integration properties) enters the group's reasoning.

Model: *First key reframing*: the issue shifts from "OT vs CRDT algorithm" to "Yjs vs ShareDB library."

- P_1 (implement OT from scratch), P_2 (implement CRDT from scratch): **set aside** (weakened). Reason: both too risky for the 6-week timeline. The reason is recorded, so these positions are revisited only if the timeline changes.
- P_3 (use Yjs, CRDT library): **open**.
- P_4 (use ShareDB, OT library): **open**.
- Epistemic shift: reframing from algorithm choice to library choice.

Symbolic model AF_5**:** $Args_5 = \{timeline, P_1, P_2, \alpha_1, \alpha_2, P_3, P_4\}$, where $\alpha_1 = (P_1 \text{ risky}, b, T4, con)$ and $\alpha_2 = (P_2 \text{ risky}, b, T4, con)$ are the evidence records of the T4 UNDERMINE, and HYPOTHESIZE registers the position records P_3 ($\equiv \alpha_3$) and P_4 ($\equiv \alpha_4$) at T5. $Att_5 = \{(\alpha_1, P_1), (\alpha_2, P_2)\}$, so $st(P_1) = st(P_2) =$ weakened. $Cm(a) = \{timeline\}$, $Cm(b) = \{P_1, P_2, \alpha_1, \alpha_2\}$, $Cm(c) = \{P_3, P_4\}$. $Dep(\alpha_1) = Dep(\alpha_2) = \{timeline\}$ (if the timeline is relaxed, these arguments lose force). *Awareness expansion:* EXPAND-AWARENESS adds the library atoms *Yjs* and *ShareDB* to *Prop* and to every $\mathcal{A}_i$ at T5, and HYPOTHESIZE adds the positions P_3 and P_4 built on them. Before T5, only $\{P_1, P_2\}$ were in consideration. *Positions:* P_1, P_2 weakened by α_1, α_2. P_3, P_4 active, no attacks yet.

T6: Carol: I've prototyped with Yjs before on a side project. The yjs-prosemirror binding, which is our editor, is well documented. I don't know if ShareDB has the same ProseMirror integration.

T7: Bob: Both work with our Node.js backend. But with Yjs we could go serverless or use a central server. More architectural flexibility.

Operations (T6): SUPPORT (P_3: ProseMirror integration exists and Carol has experience), UNDERMINE (P_4: integration status unknown).
Operations (T7): SUPPORT (P_3: architectural flexibility).

Why these matter: Carol's prior experience with Yjs is both evidence (she's prototyped it) and a practical argument (less ramp-up time). Her uncertainty about ShareDB's integration is an asymmetry: P_3 has a known integration story while P_4 has an unknown one. Bob's point about flexibility adds a different dimension of support.

Model:

- P_3 arguments pro: ProseMirror binding exists (Carol, from experience); architectural flexibility (Bob).
- P_4 arguments con: ProseMirror integration unclear (Carol).
- P_3 is emerging as the stronger candidate, but concerns haven't been raised yet.

Symbolic model AF_7**:** New evidence records: $\alpha_5 = (P_3 \text{ ProseMirror ok}, c, T6, pro)$, $\alpha_6 = (P_4 \text{ ProseMirror unclear}, c, T6, con)$, $\alpha_7 = (P_3 \text{ arch. flexibility}, b, T7, pro)$. $Dep(P_3) = \{\alpha_5, \alpha_7\}$. $Att_7 = Att_5 \cup \{(\alpha_6, P_4)\}$, so P_4 is weakened. $Cm(c) = \{P_3, P_4, \alpha_5, \alpha_6\}$, $Cm(b) = \{P_1, P_2, \alpha_1, \alpha_2, \alpha_7\}$. *Standing:* P_3 is active and grounded through α_5 and α_7, and P_4 is weakened by α_6. P_3 is the stronger candidate.

T8: Bob: CRDTs have a known problem with document size. The CRDT metadata grows over time and can get large for long-lived documents. Yjs has some GC mechanisms but they're not trivial.

T9: Alice: Is that a problem for our initial launch? Our documents are typically 5–10 pages.

T10: Bob: Probably not for launch. It's a long-term concern. But I want to flag it because switching from CRDT to OT later would be a rewrite, not a refactor.

Operations (T8): UNDERMINE (P_3: document size risk).
Operations (T9): QUESTION (is the risk relevant to our context?).
Operations (T10): SUPPORT (low severity for launch) + UNDERMINE (irreversibility makes it a strategic risk).

Why this sequence matters: Bob raises a concern, Alice challenges its relevance to the immediate context, and Bob concedes on the short term but flags the long-term irreversibility. This creates a *conditional risk*, something that's acceptable under current assumptions but becomes problematic if assumptions change. The model must capture not just "Bob has a concern" but the precise conditions under which the concern applies.

Model:

- P_3 argument con: document size risk (Bob). Severity assessment: **low for current use case**, **high if long-lived documents needed**. Risk characteristic: **irreversible**, since switching later is a rewrite.
- Sub-issue I_2: "Is document size a problem for us?" Status: **provisionally settled**, not for launch.

Symbolic model AF_{10}: New evidence records: $\alpha_8 = (P_3 \text{ doc size risk}, b, T8, con)$, $\alpha_9 = (P_3 \text{ ok for short docs}, b, T10, pro)$, $\alpha_{10} = (P_3 \text{ irreversible if wrong}, b, T10, con)$. $Att_{10} = Att_7 \cup \{(\alpha_8, P_3), (\alpha_{10}, P_3)\}$, so P_3 is weakened from T8. $Dep(P_3) = \{\alpha_5, \alpha_7, \alpha_9\}$. $Dep(\alpha_9) = \{a_1\}$, where $a_1 :=$ "docs are short", stated by Alice at T9. The assumption explicitly conditions the argument. $Dep(\alpha_{10}) = \emptyset$ (the irreversibility argument is unconditional). $Cm_{10}(b) = Cm_7(b) \cup \{\alpha_8, \alpha_9, \alpha_{10}\}$. *Key:* α_9 answers α_8 for the launch, conditional on a_1, while α_{10} is an independent attack on P_3. Sub-issue I_2 is provisionally settled: document size is not a launch problem given a_1.

T11: Carol: If we go with Yjs and WebRTC, we could support offline editing natively. User research showed spotty connectivity is a pain point.

T12: Bob: Hmm, but if edits are peer-to-peer, access control is hard. We need role-based permissions.

T13: Carol: Can we use Yjs but with a central server as the sync point? We'd get the CRDT benefits, conflict resolution, offline merge, but the server can enforce access control.

T14: Bob: Yes, that's actually the recommended production setup for Yjs. You run a Yjs WebSocket server as the sync point. And we already run WebSocket servers for notifications.

Operations (T11): SUPPORT (P_3: offline editing, grounded in user research).
Operations (T12): UNDERMINE (P_3 in pure P2P form: access control problem).
Operations (T13): HYPOTHESIZE (new hybrid position P_5: Yjs + central server relay).
Operations (T14): SUPPORT (P_5: standard production setup, fits existing infrastructure).

Why T13 is a key move: Carol resolves the tension between two competing concerns (CRDT benefits vs. access control) by *proposing a hybrid* that keeps the advantages of both. In argumentation terms, she introduces a new position that is not a compromise but a synthesis. In the formal framework, this is hypothesis generation: the hybrid combines options already under discussion into a new position.

Model: *Second key reframing*: from "Yjs peer-to-peer" to "Yjs server-relayed."

- Position P_5: Yjs with central WebSocket server as sync/authority point. Status: **leading**.
- P_5 pro: CRDT conflict resolution (inherent); offline merge (inherent); access control via server (T13); existing WebSocket infrastructure (T14, Bob); ProseMirror binding (T6, Carol).
- P_5 con: the CRDT document-size concern (Bob, T8), recorded against P_3.

- Pure P2P variant P_3: weakened, and now also attacked for access control.
- Epistemic shift: hybrid position resolves the tension between CRDT benefits and permission requirements.

Symbolic model AF_{14}: New records: the evidence records $\alpha_{11} = (P_3$ offline editing, $c, T11, pro)$ and $\alpha_{12} = (P_3$ P2P access control, $b, T12, con)$, the position record P_5 ($\equiv \alpha_{13}$, Carol, T13) registered by HYPOTHESIZE, and the evidence record $\alpha_{14} = (P_5$ std. setup + infra, $b, T14, pro)$. $Att_{14} = Att_{10} \cup \{(\alpha_{12}, P_3)\}$. $Dep(P_3) = \{\alpha_5, \alpha_7, \alpha_9, \alpha_{11}\}$. $Dep(P_5) = \{\alpha_5, \alpha_{11}, \alpha_{14}\}$: the hypothesis rests on the ProseMirror binding and offline merge it shares with P_3, and T14 adds the infrastructure fit. $Cm_{14}(c) = Cm_{10}(c) \cup \{\alpha_{11}, P_5\}$, $Cm_{14}(b) = Cm_{10}(b) \cup \{\alpha_{12}, \alpha_{14}\}$. ***Key:*** the server-relayed design answers the access-control concern by construction, so α_{12} attacks only the pure P2P position P_3. The document-size concerns α_8 and α_{10} are about CRDTs and remain recorded against P_3. ***Standing:*** P_5 is active with no incoming attack, and every record in $Dep^*(P_5)$ is active, so a continuation asserting P_5 is grounded (Definition 2.2).

T15: Bob: I want to come back to the document size issue. If we go CRDT, every edit operation is stored permanently in the CRDT state. For a 10-page document edited for months, the CRDT metadata could be 10–50× larger than the content. Yjs has compaction but it's not trivial. And switching from CRDT to OT later would be a six-month rewrite.

T16: Alice: How confident are you that the problem will actually manifest? Our documents are short and have burst editing, a few days of activity, then they become read-only.

T17: Bob: For the current use case, probably 80% chance it's fine. But the Q2 roadmap includes long-running project documents. Those would be edited continuously for months.

T18: Alice: Q2 isn't confirmed. I don't want to make an architectural decision now based on a feature that might not happen. Here's what I propose: we go with Yjs for launch. Bob, write up the risk with specific thresholds, when should we start worrying. If Q2 confirms long-running documents, we evaluate then.

T19: Bob: I'll write it up. But I want it on the record that I think this is short-sighted. If we'd gone with ShareDB, we wouldn't be carrying this risk at all.

Operations (T15): UNDERMINE (P_3: Bob escalates the CRDT document-size concern with specific numbers and the irreversibility argument).
Operations (T16): QUESTION (Alice challenges the probability of the risk manifesting).
Operations (T17): SUPPORT (P_5: 80% fine for current use) + UNDERMINE (P_3: the Q2 roadmap would raise the CRDT risk).
Operations (T18): RESOLVE (Alice makes the decision by authority, with explicit conditions for revisiting).
Operations (T19): OBSERVE (Bob records dissent as a public commitment).

Why this is the most important Phase 3 moment: This is a decision made *despite* unresolved disagreement. Bob genuinely believes the team is making a mistake, and Alice acknowledges his concern but overrides it based on product priorities. The model must capture not just the decision but the *structure of the disagreement*: who dissented, why, what would change their mind, and what conditions were explicitly agreed as triggers for re-evaluation.

In the formal framework, Alice's T18 is a commitment act: she commits the team to P_5, and the commitment is *conditional*: it depends on the assumptions a_1–a_3 below. Bob's T19 is a public recording of dissent: he accepts the decision but does not retract his argument. This is different from both agreement (Bob doesn't endorse the decision) and from blocking (he doesn't prevent it).

Symbolic model AF_{19} (final): New records: $\alpha_{15} = (P_3$ 10–50× metadata, $b, T15, con)$ (strengthens α_8), $\alpha_{16} = (P_5$ ok for current use, $b, T17, pro)$, $\alpha_{17} = (P_3$ Q2 would raise risk, $b, T17, con)$, $\alpha_{18} =$ (decide P_5 for launch, $a, T18, resolve)$, $\alpha_{19} =$ (dissent: prefers ShareDB, $b, T19, con)$.

$Att_{19} = Att_{14} \cup \{(\alpha_{15}, P_3), (\alpha_{17}, P_3)\}$.

$Dep(P_5) = \{\alpha_5, \alpha_{11}, \alpha_{14}, \alpha_{16}\}$, $Dep(\alpha_{16}) = \{a_1, a_2\}$ and $Dep(\alpha_{18}) = \{P_5, a_1, a_2, a_3\}$, where $a_1 :=$ "docs are short (5–10 pages)", $a_2 :=$ "editing is burst" (Alice, T16) and $a_3 :=$ "Q2 long-running docs not confirmed" (Alice, T18).

Resolve: at T18, P_5 is active with no recorded incoming attack and Alice holds decision authority, so the authoritative RESOLVE of Table 3 applies: $st(P_5) =$ resolved, and Alice commits the decision record α_{18}. Bob's CRDT concerns $\alpha_8, \alpha_{10}, \alpha_{15}, \alpha_{17} \in Cm(b)$ are already on record (they attack P_3, whose CRDT basis P_5 shares), and T19 makes the dissent explicit as α_{19}.

Commitments: $Cm(a) = \{timeline, \alpha_{18}\}$ (Alice commits to the decision). $Cm(b) = \{P_1, P_2, \alpha_1, \alpha_2, \alpha_7, \alpha_8, \alpha_9, \alpha_{10}, \alpha_{12}, \alpha_{14}, \alpha_{15}, \alpha_{16}, \alpha_{17}, \alpha_{19}\}$ (Bob commits to both pro and con arguments but **not** to α_{18}, recording dissent via α_{19}). $Cm(c) = \{P_3, P_4, P_5, \alpha_5, \alpha_6, \alpha_{11}\}$.

Decision status: I_1 resolved with dissent. P_5 accepted. Decision authority: a (product lead). *Conditional commitment:* $Dep(\alpha_{18}) = \{P_5, a_1, a_2, a_3\}$. If any a_i is withdrawn, $Affected^*(a_i)$ reports the decision for re-evaluation. *Dissent record:* $\alpha_{19} \in Cm(b)$ but $\alpha_{18} \notin Cm(b)$. Bob accepts the decision procedurally but does not endorse it epistemically.

Final model.

- Issue I_1: Status $\rightarrow$ **decided with dissent**.
- Decision: P_5 (Yjs, server-relayed via WebSocket). Decided by: Alice (product lead authority).
- Dissent: Bob. Position: prefers ShareDB for long-term architectural safety. Objection recorded: "short-sighted."
- Assumptions underlying the decision:
  - a_1: Documents are short (5–10 pages). *Currently true.*
  - a_2: Editing pattern is burst (few days of activity, then read-only). *Currently true.*
  - a_3: Q2 long-running project documents not confirmed. *Currently true.*
- Conditional commitment: If Q2 roadmap confirms long-running documents, the team will re-evaluate the architecture choice.
- Dependencies: $Dep(\alpha_{18}) \supseteq \{a_1, a_2, a_3\}$. Withdrawing any of them reports the decision for re-evaluation.
- Action items: Bob writes a risk analysis with specific thresholds.
- Positions explored and set aside (with reasons preserved):
  - P_1, P_2 (from-scratch OT/CRDT): set aside because of the 6-week timeline.
  - P_4 (ShareDB): not chosen, since its ProseMirror integration is unclear.
  - P_3 (Yjs pure P2P): set aside, since peer-to-peer sync makes access control hard, and it carries the CRDT document-size concerns.
- Epistemic shifts: (1) T5: reframing from algorithm to library choice; (2) T13: hybrid position resolves CRDT-vs-access-control tension; (3) T18–19: decision made with recorded dissent.

Counterfactual test. One week later, the Q2 roadmap is confirmed with long-running project documents. The verifier should:

1. Identify that assumption a_3 has changed (Q2 now confirmed).
2. Trace the dependency: a_3 is one of the assumptions underlying the P_5 decision.
3. Surface the conditional commitment: the team explicitly agreed to re-evaluate in this scenario.
4. Recall Bob's document size analysis as the relevant prior concern, including his specific numbers (10–50$\times$ metadata growth) and the irreversibility argument (six-month rewrite).
5. *Not* re-explore from-scratch positions (P_1, P_2), since they were set aside because of the timeline, which has not changed.
6. *Not* re-explore pure P2P Yjs (P_3), since it was set aside because of access control and is not reached by $Affected^*(a_3)$.
7. Focus the re-evaluation on P_5 vs. P_4 (ShareDB), since the document size concern is specific to CRDTs and ShareDB was the alternative Bob advocated.

This is the selective re-grounding the dependency map supports: an assumption change reaches exactly the records that rest on it, and questions settled for unrelated reasons stay settled.

Symbolic model under counterfactual (a_3 changes): Assumption a_3 is withdrawn: after the Q2 confirmation, $\mathcal{M}', w \not\models a_3$. $\mathit{Affected}^*(a_3) = \{\alpha_{18}\}$: the decision is the only record that rests on a_3 (Definition 2.3). It is reported for re-evaluation without changing its standing. Bob's Q2 concern α_{17} and the storage concerns α_8, α_{15} are the recorded starting points for the re-evaluation, and his dissent α_{19} names the alternative P_4 (ShareDB). Arguments α_1, α_2 (P_1, P_2 too risky) are *not* affected: $Dep(\alpha_1) = Dep(\alpha_2) = \{timeline\}$, and the timeline has not changed. α_{12} (P_3 P2P access control) is *not* affected, since no recorded dependency path leads from it to a_3.

C Experimental Details

C.1 Benchmarks and Baselines

ReviseQA. We use all 930 verified scenarios of the public release (6,510 edit steps, extending the 210-example subset of the original paper) under the explicit-edit track, with no correction feedback or demonstration reasoning. Accuracy@k has easy, medium and hard levels at $k = 2$, 4 and 7. We report every depth. RAG retrieves top-22 statements from the current post-edit context using TF-IDF. The budget matches the median of 22 recorded premises across the 6,510 edit steps. We add RAG as a retrieval baseline alongside the benchmark's LLM-only protocol. The end-to-end arm uses a single GPT-4o interpretation of each scenario, shared by all QA models. Per-depth results are in Appendix D.1.

MemAB-FC. The FactConsolidation split of MemoryAgentBench (Hu et al., 2026b) streams 455 to 18,332 numbered facts at 6K, 32K, 64K and 262K tokens, with 35–40% superseded under the larger-serial-wins rule. Each length has 100 single-hop and 100 multi-hop questions. LLM-only reads the stream tail-truncated to the QA model's window. RAG retrieves top-50 individual facts from the full stream using TF-IDF. Verifier (native) maintains standing and selects active records by following subject–object links from entities named in the question. A separate control applies the same TF-IDF retriever at top-50 to the active records, isolating the effect of standing under a fixed retrieval configuration. The MemAB-FC protocol and per-length results are in Appendix D.2.

BEAM. The 100K split of BEAM: 20 conversations, 400 probing questions, ten memory abilities, scored against the benchmark's rubric criteria with its judge prompt and Qwen2.5-32B-Instruct-AWQ as judge. RAG and the verifier both rank with BEAM's retriever, bge-small-en-v1.5, and the RAG arm is the benchmark's own design, retrieving BEAM's user–assistant turn pairs up to the verifier's budget. Ingestion, arm budgets and the revised/stands split are in Appendix D.4.

Authored scenarios and RECON. The Phase 2 and Phase 3 scenarios of Section 3 support per-utterance classification (Tables 2 and 4). Phase 2 and RECON's released provenance skeletons (Arya, 2026) supply ground-truth dependency structure for the soundness check (Appendix E.1).

C.2 Models, Prompt Conditions and Serving

Interpreter models and prompt conditions. The classification experiment runs Claude Sonnet 4, GPT-4o, Qwen2.5-7B and Llama-3.1-8B (the last two via vLLM) under three prompt conditions, *Minimal*, *Definitions* and *State-augmented*, with 5 runs per cell, the first at temperature 0 and the other four at 1.0. *Minimal* gives the eight operation names with a one-line gloss each. *Definitions* adds a full definition and a generic example per operation, plus explicit rules for the neighbouring pairs (Undermine vs. Revise, Expand-Awareness vs. Observe, when Hypothesize and Revise co-occur). *State-augmented* further appends the engine state rendered before six checkpoint turns: which hypotheses are active, weakened or abandoned, and when awareness has expanded. Phase 3 uses Definitions only. Scoring is multi-label F1 and exact match per utterance, with separate reporting on the four *key shifts* per scenario, the turns at which standing or awareness changes, drawn from awareness expansion, undermining, hypothesis abandonment, causal reversal and decision with dissent.

Model identifiers and serving. Closed models are called by API alias. Open-weight models are served with vLLM 0.19 on a single A100-40GB.

Name in the paper	Identifier
GPT-4o	`gpt-4o`
GPT-4o-mini	`gpt-4o-mini`
Claude Sonnet 4	`claude-sonnet-4-20250514`
Qwen2.5-7B	`Qwen/Qwen2.5-7B-Instruct`
Qwen2.5-14B	`Qwen/Qwen2.5-14B-Instruct`
Gemma-3-12B	`unsloth/gemma-3-12b-it`
Llama-3.1-8B	`meta-llama/Llama-3.1-8B-Instruct`
BEAM judge	`Qwen/Qwen2.5-32B-Instruct-AWQ`
BEAM retriever	`BAAI/bge-small-en-v1.5`

C.3 Statistics and Supersession Capture

Statistics. Paired comparisons use exact McNemar tests (McNemar, 1947) on discordant pairs: scenarios on ReviseQA and questions on MemAB-FC, where questions within one variant share a single stream and the four stream lengths are independent streams. On BEAM the scores are graded, so paired comparisons are Wilcoxon signed-rank tests (Wilcoxon, 1945) over questions, per QA model and pooled over the five with each (model, question) pair as one observation. The intervals in Figure 3c are 95% percentile bootstraps over questions.

Supersession capture and state agreement. The measurements in Section 4.3 come from the benchmark annotations and the end-to-end arm's update logs. On ReviseQA, 4,361 of the 6,510 edit steps (67.0%) remove a premise inside the conclusion's maintained dependency set, and the GPT-4o Interpreter emits a removal on 4,355 of them (99.9%). On MemAB-FC, 35.4–39.5% of the streamed facts are superseded under the larger-serial-wins rule (161/455 at 6K up to 7,237/18,332 at 262K), and the dependency map the Interpreter builds matches the one built from the structured updates at Jaccard 1.000 at 6K and 0.989 at 32K.

D Full Benchmark Results

D.1 ReviseQA: Full Per-Depth Results

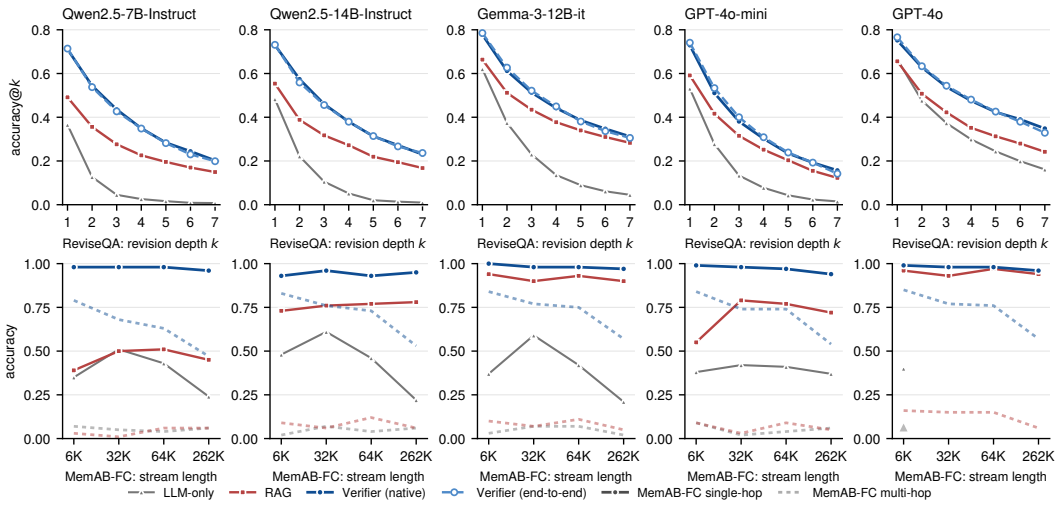

Figure 6: One column per QA model. Top: ReviseQA accuracy@k under all four arms. Bottom: MemAB-FC single-hop (solid) and multi-hop (dotted) accuracy across stream lengths for LLM-only, RAG and Verifier (native).

Figure 6 plots accuracy@k for all five QA models and all four arms, with RAG at top-22. The native arm's lead over RAG is significant at every depth on four QA models and through $k \leq 6$ on Gemma-3-12B (exact McNemar, $p < 0.05$). At the benchmark's medium cut-off, k=4, the native arm scores 0.302–0.475 against RAG's 0.226–0.377 and LLM-only's 0.026–0.299. The end-to-

end arm replaces the benchmark's edit annotations with a GPT-4o Interpreter that translates each natural-language edit into engine updates. Its accuracy tracks the native arm at every depth on all five models, differing by at most 2.26 percentage points at k=2, 0.75 at k=4 and 1.94 at k=7. The end-to-end arm stays above RAG at every depth on all five models.

D.2 MEMAB-FC: PROTOCOL AND PER-LENGTH RESULTS

Protocol. The benchmark supplies serially numbered natural-language facts and instructs agents that newer facts carry larger serial numbers. We parse each fact with 37 hand-written relation templates and a last-"is" fallback. Two facts share a supersession key when they share (template, subject). Under the larger-serial-wins rule, the latest record for each key remains active. A later fact assigning a new value to the same attribute of the same subject is treated as an observation of the new value and a revision of the superseded record. All arms use the same QA prompt and normalised-containment scoring against the gold answer list. LLM-only reads the raw stream tail-truncated to the QA model's context budget. GPT-4o LLM-only is reported at 6K. Each variant contains 100 questions sharing one stream, and the four stream lengths use independent streams.

Standing control. RAG and Verifier + TF-IDF use the same deterministic TF-IDF retriever, top-50 budget and QA configuration. RAG retrieves from the full stream, whereas Verifier + TF-IDF retrieves only from records in good standing. Mean prompt lengths differ by less than 3% in every variant. This comparison isolates the effect of maintaining standing under a fixed retrieval configuration. The control leads RAG at every length on single-hop and at every length on multi-hop except a tie for Qwen2.5-7B at 262K.

Record selection. Verifier (native), the arm in the main results, uses the same active records but selects them by following subject–object links from entities named in the question. Selection starts from at most eight matching subjects and follows links for up to three steps, with at most eight frontier entities per step and 50 selected records. The engine registers the selected record IDs as the dependencies of a question-specific hypothesis before the QA call. The QA model reads the selected facts in serial order. This arm leads RAG at every length on both hop types for all five QA models (Table 5). Pooled over the four lengths, both Verifier and Verifier + TF-IDF lead RAG significantly for every model and hop type (exact McNemar, $p < 0.05$).

End-to-end ingestion. A GPT-4o Interpreter ingested the 6K and 32K streams, deciding which of the five most similar active facts, if any, each incoming fact supersedes, without hand-written supersession templates. The resulting dependency map matches the one built from the structured updates at Jaccard 1.000 at 6K and 0.989 at 32K. We evaluate both selection procedures over this map. For the TF-IDF control, question-level correctness agrees with the structured-update run on 392–398 of 400 questions across the five QA models. For the Verifier (native) selection, the three evaluated models differ from their native counterparts by at most 1.5 percentage points on either hop type when pooling the two lengths, and remain above RAG (Table 6).

Published systems. MemoryAgentBench reports FactConsolidation results with a GPT-4o-mini backbone. With this backbone, our LLM-only arm scores 0.395/0.052 single-/multi-hop, compared with the published long-context agent's 0.45/0.05. Our fact-level RAG scores 0.708/0.065, compared with 0.48/0.03 for the benchmark's BM25 agent over 512-token chunks. Verifier + TF-IDF reaches 0.980/0.228 under the same retrieval configuration as our RAG, and Verifier (native) reaches 0.970/0.715.

D.3 MEMAB-FC: MULTI-HOP ANALYSIS

Chain analysis. We run a chain search for each of the 400 multi-hop questions. Starting from the gold answer, we search backwards through records whose object matches the current target, then continue from their subjects, for at most six links. A chain is found when a subject occurs in the question. The search finds a chain over active records for 336 questions and none for the remaining 64. These labels describe subject–object connectivity to the gold answer. We then apply the same search to the records selected by each arm.

Evidence selection. We record the evidence each arm passes to the QA model. Fact and active-record counts per length are 455/294, 2,310/1,473, 4,580/2,889 and 18,332/11,095. Chain labels agree across the five QA models within each arm.

Table 5: MemAB-FC accuracy per stream length (n=100 questions per cell). Verifier uses the record selection of the main results. Verifier + TF-IDF is the standing control, retrieving top-50 active records with the same retriever as RAG. Both maintained-state arms use structured updates. Bold marks the best accuracy within each model and column.

		Single-hop				Multi-hop			
QA model	Arm	6K	32K	64K	262K	6K	32K	64K	262K
Qwen2.5-7B	LLM-only	0.35	0.51	0.43	0.24	0.07	0.05	0.04	0.06
	RAG	0.39	0.50	0.51	0.45	0.03	0.01	0.06	0.06
	Verifier + TF-IDF	0.95	0.94	0.92	0.91	0.18	0.10	0.14	0.06
	Verifier	**0.98**	**0.98**	**0.98**	**0.96**	**0.79**	**0.68**	**0.63**	**0.47**
Qwen2.5-14B	LLM-only	0.48	0.61	0.46	0.22	0.02	0.07	0.04	0.06
	RAG	0.73	0.76	0.77	0.78	0.09	0.06	0.12	0.06
	Verifier + TF-IDF	**0.98**	**0.97**	**0.98**	0.94	0.36	0.14	0.20	0.10
	Verifier	0.93	0.96	0.93	**0.95**	**0.83**	**0.76**	**0.73**	**0.53**
Gemma-3-12B	LLM-only	0.37	0.59	0.42	0.21	0.03	0.07	0.07	0.02
	RAG	0.94	0.90	0.93	0.90	0.10	0.07	0.11	0.05
	Verifier + TF-IDF	**1.00**	**0.98**	**0.98**	**0.97**	0.39	0.21	0.26	0.15
	Verifier	**1.00**	**0.98**	**0.98**	**0.97**	**0.84**	**0.77**	**0.75**	**0.57**
GPT-4o-mini	LLM-only	0.38	0.42	0.41	0.37	0.09	0.02	0.04	0.06
	RAG	0.55	0.79	0.77	0.72	0.09	0.03	0.09	0.05
	Verifier + TF-IDF	**0.99**	**0.98**	**0.98**	**0.97**	0.38	0.20	0.22	0.11
	Verifier	**0.99**	**0.98**	0.97	0.94	**0.84**	**0.74**	**0.74**	**0.54**
GPT-4o	LLM-only	0.40	—	—	—	0.06	—	—	—
	RAG	0.96	0.93	0.97	0.94	0.16	0.15	0.15	0.06
	Verifier + TF-IDF	**0.99**	**0.98**	**0.98**	0.95	0.48	0.27	0.32	0.15
	Verifier	**0.99**	**0.98**	**0.98**	**0.96**	**0.85**	**0.77**	**0.76**	**0.57**

Table 6: MemAB-FC with the Verifier (native) selection: native and end-to-end accuracy pooled over 6K and 32K (n=200 questions per hop type and model).

	Single-hop		Multi-hop	
QA model	Native	End-to-end	Native	End-to-end
Qwen2.5-7B	0.980	0.975	0.735	0.740
Qwen2.5-14B	0.945	0.940	0.795	0.800
Gemma-3-12B	0.990	0.985	0.805	0.820

Table 7: MemAB-FC multi-hop chain coverage and QA accuracy under structured updates. Each question contributes five question–model pairs to the pooled accuracy. The two arms use the same active state and chain diagnostic.

	Verifier + TF-IDF		Verifier	
Chain diagnostic	Questions	Accuracy	Questions	Accuracy
Chain contained in selected records	107	0.636	291	0.928
Active chain found, not fully selected	229	0.071	45	0.102
No active chain found	64	0.066	64	0.134

Among the 1,558 incorrect question–model pairs under the TF-IDF control, 1,064 have an active chain that is not fully contained in the retrieved facts. The Verifier (native) selection raises the number of questions with a chain in the QA input from 107 to 291 (Table 7). On these groups, pooled accuracy is 0.636 for the TF-IDF control and 0.928 for Verifier. These groups contain different questions. For Verifier, 1,350 of the 1,455 pairs with a selected chain are answered correctly.

Figure 1 example. The example is the 18th question in the MemAB-FC 6K multi-hop split, with Qwen2.5-7B for QA. RAG retrieves facts 29, 123 and 163 but not 33, and answers Paris. Verifier (native) selects records 33, 113 and 123, including the two records linking Madame du Barry to London, and answers London. With GPT-4o interpretation, the dependency map contains 294 active records from 455 input facts, and Verifier (end-to-end) also answers London.

Superseded gold answers. For 61 of the 64 questions with no active chain found, the search finds a chain when superseded records are included. For example, one question asks for the capital of the country that gave rise to the sport played by Rogério Ceni. A chain using superseded records passes through *rugby union was created in the country of India* (#186) and *The capital of India is New Delhi* (#158). Fact #341 replaces the latter with Grosseto, while the gold answer remains New Delhi.

D.4 BEAM: Protocol, Split and Results

Data. We use the whole 100K split of BEAM (Tavakoli et al., 2026): 400 probing questions with 1,051 rubric criteria. Every answer is scored with the benchmark's unified judge prompt, with Qwen2.5-32B-Instruct-AWQ as judge, one call per criterion with scores in $\{0, 0.5, 1\}$, averaged per question. BEAM scores event ordering with a Kendall τ_b coefficient over the order and presence of events. We score it with the same rubric judge as the other nine abilities so that all ten share one scale. All arms answer with the benchmark's own RAG answer prompt and differ only in the context.

Ingestion. A GPT-4o Interpreter extracts at most 20 atomic statements per message from both speakers, 53,222 in all. Each statement is decided against the live active set: the candidates are the most similar active statements under three retrieval views, capped at 20, and one call returns which candidate, if any, states an earlier value of the same attribute and whether the value changed. The ownership rule keeps a user commitment active when the revising statement is the assistant's. In all, 18,696 statements are retired. The dependency map is built once and shared by every QA model.

Arms and budgets. *LLM-only* reads the transcript tail-truncated to the QA model's window, 12.0K tokens for the 16K-window open models and 109.6K tokens for GPT-4o and GPT-4o-mini. Eighteen of the 20 conversations exceed the latter. *RAG* follows BEAM's own design, bge-small-en-v1.5 over user–assistant turn pairs, with pairs added in similarity order up to the length of the verifier's top-100 active statements for that question, 3.3K tokens on average. *Verifier (end-to-end)* ranks the active statements with the same retriever, adds the source turn of each in rank order up to the same length, and appends a retraction list of every statement in those turns that was later replaced by a different value, together with its replacement, 3.7K tokens in all on average. A turn whose statements are all retired never enters.

Reading the split. For each question BEAM lists the evidence turns its answer rests on. A question is labelled *revised* when one of those turns holds a statement that the dependency map marks as replaced by a later statement with a different value, and *stands* otherwise. The label is read from the map before any answer is scored. Abstention questions have no evidence turns, nor do four summarization and one preference question, so 355 of the 400 carry a label, 156 stand and 199 revised, across nine abilities. RAG and the end-to-end verifier were run three times with identical settings for every QA model but GPT-4o, and Table 8 and Figure 3c report the mean of the three runs. A repeated run changes a model's score by at most 0.010. Pooled over the five QA models, the end-to-end verifier scores 0.347 against RAG's 0.353 on the 780 question–model pairs where every premise still stands and 0.249 against 0.202 on the 995 where one was revised. Paired Wilcoxon tests put the verifier ahead where a premise was revised on three QA models ($p<0.05$) and find no difference where every premise still stands on any of the five.

Per-model results. Averaged over all ten abilities the end-to-end verifier is level with or ahead of RAG for every QA model (Table 8).

Table 8: BEAM, all 400 probing questions per QA model: score over all ten abilities / on knowledge update. RAG and Verifier: mean of three runs (GPT-4o: one run).

QA model	LLM-only	RAG	Verifier
Qwen2.5-7B	0.166 / 0.087	0.256 / 0.258	**0.301 / 0.350**
Qwen2.5-14B	0.198 / 0.062	0.340 / 0.400	**0.341 / 0.433**
Gemma-3-12B	0.136 / 0.075	0.258 / **0.263**	**0.259** / 0.258
GPT-4o-mini	0.279 / 0.300	0.350 / 0.425	**0.351 / 0.446**
GPT-4o	0.286 / 0.325	0.328 / 0.400	**0.362 / 0.487**
Pooled (n=2000 / 200)	0.213 / 0.170	0.306 / 0.349	**0.323 / 0.395**

E Analysis Experiments

E.1 Soundness of *Affected**: Phase 2 and RECON

Table 9 collects every measurement of the structural check given ground-truth dependency structure: the authored Phase 2 scenario and the third-party RECON benchmark, under the one-step and iterated queries, with GPT-4o answering the same question as a reference. RECON (Arya, 2026) is a benchmark of 24 investigation case files (50–100K tokens each) with six memory tasks. Its *counterfactual* task queries *Affected*$^*(\beta)$ for a trigger record β: each case ships a provenance skeleton whose evidence chains carry explicit `requires`/`establishes` attributes, and each counterfactual scenario names a trigger step together with the steps that depend on it and those that do not. We verified that the gold `dependent_steps` equal within-chain reachability over those attributes for all 64 scenarios of the 16 medical and finance case files (32 per domain). The medical and finance skeletons serialise these attributes, and the dependents in the counterfactual task span several chain links, so the task exercises the walk.

Oracle protocol. Each chain is loaded into the engine as hypotheses with *Dep* set from `requires`/`establishes`. We then query the trigger. Two LLM arms answer the same question with GPT-4o at temperature 0: *isolated chain* receives only the trigger's chain ($\sim$550 tokens), and *full skeleton* receives the entire case skeleton ($\sim$51K tokens), RECON's own oracle setting. Scoring counts every predicted step outside the gold dependent set as a false positive. Phase 2 contributes 16 decisions (9 dependent) from 4 retraction queries over 4 hypotheses. The one-step query decides 12 of the 16 correctly and *Affected** all 16. RECON contributes 64 scenarios with 198 dependent steps. Each of the full-skeleton reader's three inexact answers includes one extra step.

Table 9: Soundness of the retraction query on Phase 2 and RECON given ground-truth dependency structure. Exact: whole affected set correct; TP/FP/FN over dependent steps, any step outside the gold set counted as FP.

Query	Input	Exact	TP/FP/FN	Cost per query
Phase 2 (authored dialogue, Appendix B.2)				
one-step *Affected* (Definition 2.3)	GT *Dep*	1/4	5/0/4	$< 1\,\mu$s
Affected* (Corollary 2.4)	GT *Dep*	**4/4**	9/0/0	$< 1\,\mu$s
RECON (third-party, 16 case files)				
one-step *Affected*	skeleton	13/64	66/0/132	$< 1\,\mu$s
Affected*	skeleton	**64/64**	198/0/0	$0.4\,\mu$s
GPT-4o, isolated chain	$\sim$550 tokens	64/64	198/0/0	1 call
GPT-4o, full skeleton	$\sim$51K tokens	61/64	198/3/0	1 call, $\sim$\$0.13

On RECON's own multiple-choice and count questions for these scenarios, 103 of 186 are answerable from the affected set, with options matched to chain steps by content-word overlap. On these, *Affected** scores 0.948 under RECON's penalised scoring, against 0.615 for one-step *Affected*.

Reading. The one-step query misses exactly the multi-hop cases, which is why the verifier runs the iterated query *Affected** (Corollary 2.4). *Affected** matches the gold on every Phase 2 decision and every RECON scenario, at a per-query cost that remains nearly constant over the measured state sizes (Figure 4c).

E.2 Retraction Latency and Token Cost

Latency protocol. We measure the query on synthetic dependency maps $\mathcal{D}_t$ at $K \in \{13, \ldots, 2000\}$ turns under two regimes: *naive*, with $|Args_t| \sim 0.3K$, and *bounded*, with $|Args_t| \leq 50$, modelling Phase-2-like working sets where resolved hypotheses retire. In both regimes the verifier stays at 0.25–$0.46\,\mu$s per query while history replay grows linearly from $1.1\,\mu$s to $190\,\mu$s. Figure 4c reports medians over 5 seeds $\times$ 200 queries per seed in the naive regime, with shading spanning the per-seed medians. The bounded regime coincides with it. Per-query cost varies little with K and $|Args_t|$ and is governed by the size of the affected set, whose mean is below 1 in these states. Proposition 2.5's $\mathcal{O}(|Args| + |Dep|)$ is the dense-graph worst case.

Token accounting. For the per-query prompt costs in Figure 4a,b, we count the tokens of the exact prompt each arm of Section 4 sends for each MemAB-FC variant and question, using the GPT-4o tokenizer. Mean prompt lengths across the eight variants are 753–811 tokens for RAG and 97–174 for the native verifier, while the full-context prompt for the GPT models grows from 6.0K to 114.5K, tail-truncated to their input budget at 262K. For the open models it is truncated from 32K onward, at 11.4–11.7K. Untruncated, the 262K stream costs 268K tokens per query. The active-record TF-IDF control and RAG both retrieve top-50 facts, with mean prompt lengths within 3% of each other at every variant. On the write side, the end-to-end arm's Interpreter ingestion costs 96.6K tokens for the 6K stream and 494.7K for the 32K stream. Using the native verifier's prompt lengths to estimate savings against untruncated full-context QA gives token-denominated break-even at 16.3 queries (6K) and 15.3 queries (32K), of the 100 queries per variant. Verifier (native) ingests the benchmark's numbered facts by template and pays no LLM tokens at write time.

F Comparison with Related Work

Table 10 compares five capabilities. *Incremental updates* maintain a persistent state as interactions arrive. *Fact revision* updates or invalidates stored facts superseded by new information. *Support links* explicitly connect conclusions to supporting premises or evidence. *Retraction effects* compute the transitive dependents of a withdrawn premise. *Runtime checks* test whether a candidate continuation's recorded support chain remains in good standing at the current turn.

Table 10: Capabilities of conversational memory and reasoning systems. A checkmark denotes a mechanism specified in the cited work.

System	Incremental updates	Fact revision	Support links	Retraction effects	Runtime checks
Mem0	✓	✓			
A-MEM	✓				
Generative Agents	✓		✓		
Zep/Graphiti	✓	✓			
D-SMART	✓	✓			
VISTA			✓		
Ours	✓	✓	✓	✓	✓

Memory updates and support links. Mem0 updates or deletes stored facts (Chhikara et al., 2025). Zep invalidates conflicting fact edges while retaining history (Rasmussen et al., 2025). D-SMART prunes conflicting triples and searches reasoning trajectories over its knowledge graph (Lei et al., 2025). A-MEM updates note context, keywords, and tags and creates associative links (Xu et al., 2025). It does not specify a mechanism for retiring superseded facts or recording conclusion-to-premise support. Generative Agents store reflection-to-evidence pointers (Park et al., 2023), and VISTA reconstructs reasoning dependency trees from conversation histories (Zhang et al., 2026). Our verifier uses standing and support links together to check candidate continuations and identify exactly the recorded transitive dependents of a withdrawn premise, preserving unaffected claims (Corollary 2.4).

Related verification objectives. Dynamic epistemic friction uses DEL to model resistance to belief updates in dialogue (Obiso et al., 2025). AgentArmor constructs program dependency graphs from tool-execution traces and enforces security policies through a type system (Wang et al., 2025).

Neither procedure checks the current standing of a conversational conclusion's support chain. Our verifier performs this check over the maintained claim dependencies, enabling detection of reliance on withdrawn premises even when the proposed continuation violates no tool-use policy.