LEDGER: MEMORY AS COMPUTATION FOR CONVERSATIONAL AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-term conversational memory is often treated as a retrieval problem, yet many questions require computation over evidence distributed across interactions. Retrieving relevant memories does not specify which facts to count, combine, or compare. We introduce Long-term Evidence Decomposition for Grounded Executable Reasoning (LEDGER), which makes memory aggregation an explicit computation rather than an implicit step in answer generation. LEDGER extracts reusable, source-linked facts independently of future queries. A semantic router uses candidate facts to specify what to compute, while a deterministic executor evaluates the resulting restricted program over the full fact store. Valid computed records supplement rather than replace retrieved raw evidence. On LongMemEval, LEDGER improves accuracy from 64.4% to 68.9%, with gains from 70.1% to 76.6% on multi-session questions and from 65.1% to 82.5% on knowledge updates. On LoCoMo, accuracy improves from 57.3% to 61.8% against the same baseline. These results support treating conversational memory not merely as context to retrieve, but as evidence to compute over.

1 INTRODUCTION

Retrieving relevant conversational memories is not the same as answering a question that requires those memories to be combined. Long-term memory systems index a growing history, retrieve evidence for a new query, and ask a language model to produce an answer. Benchmarks such as LongMemEval and LoCoMo evaluate the resulting ability to recall and reason over extended conversations (Wu et al., 2025; Maharana et al., 2024). Yet successful retrieval leaves a distinct computational problem whenever no single session states the answer. Counting distinct events, summing expenses, comparing periods, or resolving an updated preference requires selecting and normalizing facts across sessions before applying an operation. Evidence availability is necessary for these tasks, but it is not sufficient for evidence integration.

We investigate this distinction through an audited LongMemEval development split. Session-level Vanilla RAG retrieves at least one annotated evidence session in its top five for $87/93 = 93.5\%$ of answerable questions, while answer accuracy from the top-10 payload is $58/93 = 62.4\%$. These metrics use different retrieval depths and are not a matched decomposition of pipeline errors. A more targeted coverage audit, however, finds that 21 of 29 retrieval-hit failures already contain every annotated evidence-session ID. The corresponding count is $8/9$ for multi-session failures. Session-ID coverage does not guarantee that every supporting span is available, but these cases motivate studying integration rather than treating additional retrieval coverage as the only remedy.

A selected 18-question probe further shows that query-agnostic extraction recovers 32 of 33 required semantic labels from retrieved chunks. With manually specified downstream aggregation, it answers 15 questions correctly, versus 12 for Chunked RAG. This supports the feasibility of reusable local facts, not general superiority over query-aware extraction.

We introduce LEDGER to turn this observation into an explicit computational interface for conversational memory. Offline, an extractor consolidates each session into typed, source-linked facts without observing future queries. At query time, a semantic router selects one of six aggregation operators and binds its filters and operands, or emits `NONE` for an unsupported query. A deterministic executor evaluates the program over the fact store. The computed record is then appended to Parent RAG's raw sessions rather than replacing them. Invalid or unsupported programs leave the parent

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.

memory payload unchanged. This design separates reusable extraction, query-dependent semantic interpretation, execution, and final answer synthesis.

Deterministic execution remains conditional on correct facts and programs and raw-text preservation does not prevent a wrong computed record from misleading the answer model. We therefore evaluate both final accuracy and the intermediate interfaces, distinguishing development diagnostics from the final system.

On the 402-example LongMemEval test split, LEDGER reaches 68.9% accuracy versus 64.4% for Parent RAG. Gains are concentrated in the targeted question types: multi-session accuracy rises from 70.1% to 76.6%, and knowledge-update accuracy from 65.1% to 82.5%. The reported totals imply unchanged pooled accuracy on the remaining types, not an absence of individual regressions. On LoCoMo, accuracy rises from 57.3% to 61.8%. Component ablations complement these comparisons, while earlier router failures explain why operator selection alone is an inadequate measure of successful program induction.

Our contributions are threefold:

- **Diagnosis:** an evidence-coverage audit and extraction probes that expose post-retrieval integration failures in conversational memory.
- **Architecture:** query-agnostic consolidation, semantic routing, and deterministic aggregation with provenance and a raw-text overlay.
- **Evaluation:** held-out and cross-dataset comparisons, component ablations, and stage-wise diagnostics that separate final task accuracy from extraction, program induction, and execution.

Together, the results motivate evaluating not only what conversational memory can retrieve, but also which computations it can support reliably.

2 Related Work

Conversational memory and evidence utilization. LongMemEval evaluates information extraction, multi-session and temporal reasoning, knowledge updates, and abstention (Wu et al., 2025). LoCoMo studies extended conversational histories (Maharana et al., 2024), while Multi-Session Chat supplies long-term open-domain conversations (Xu et al., 2022). Recent UtilMem diagnostics examine cross-session synthesis under semantically similar distractors and identify utilization failures even in high-recall settings (Qing et al., 2026). We therefore do not claim to originate the distinction between access and utilization. Our focus is a narrower class of explicit aggregation computations and their extraction, binding, and execution interfaces.

Persistent memory representations. MemoryBank (Zhong et al., 2024) and MemGPT (Packer et al., 2023) establish persistent memory and virtual-context management. Later systems include temporal knowledge graphs (Rasmussen et al., 2025), dynamically linked notes (Xu et al., 2025), scalable fact and graph memories (Chhikara et al., 2025), and retain–recall–reflect pipelines (Latimer et al., 2026). Offline extraction and persistence are not themselves our novelty. LEDGER specifies an executable computation over stored facts and retains raw evidence for answer synthesis.

Memory organization and runtime routing. xMemory separates reusable evidence components before grouping them in a revisable hierarchy and retrieving them adaptively (Hu et al., 2026). Its aggregation organizes memory for retrieval, while ours evaluates arithmetic, set, and latest-state operations. BudgetMem instead performs query-conditioned runtime memory processing with a learned router that selects module-level computation budgets (Zhang et al., 2026). LEDGER's router chooses a semantic program and binds its support facts and operands, rather than allocating budgets. These are related but distinct uses of consolidation and routing, and our experiments do not establish superiority over these systems.

Semantic parsing and external execution. Chain-of-thought prompting elicits intermediate natural-language reasoning (Wei et al., 2022), whereas PAL separates interpretation from program execution (Gao et al., 2023). Text-to-SQL maps questions to database queries (Yu et al., 2018), and

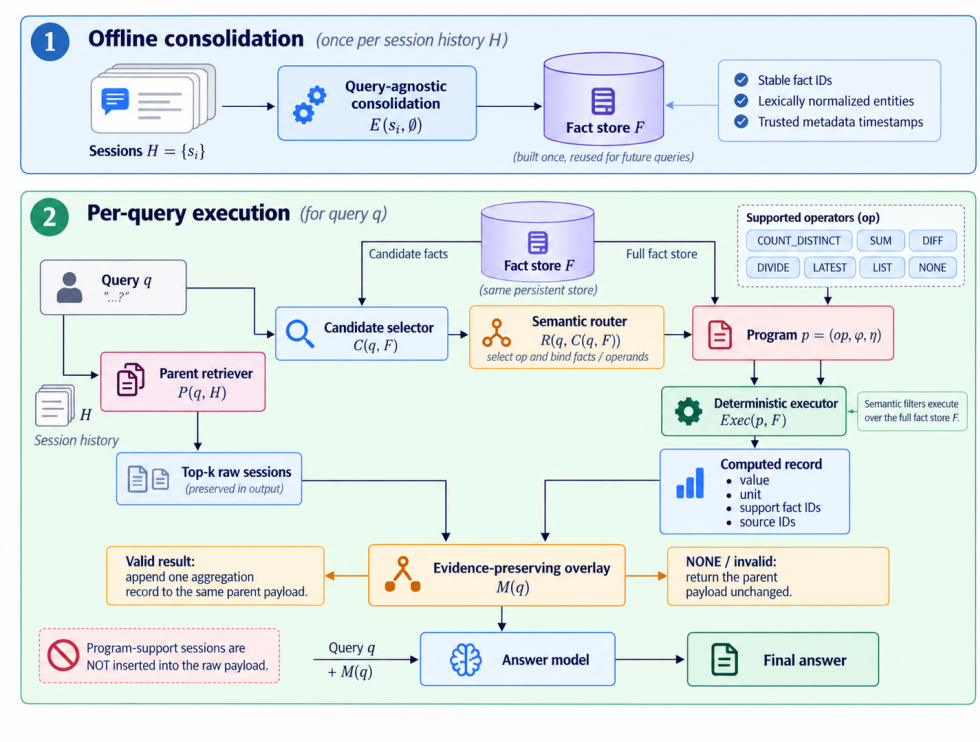

Figure 1: LEDGER separates offline, query-agnostic consolidation from query-time semantic routing and deterministic execution. A valid computed result is appended to the parent retriever's raw top-k sessions. When the router outputs NONE or produces an invalid route, the parent payload is returned unchanged. Program-support sessions are not inserted into the raw payload.

Chain-of-Table evolves a structured intermediate representation through operations (Wang et al., 2024). The LEDGER router is a restricted semantic parser, not a new parsing formalism. Unlike a curated database, its record envelope is fixed but its predicates, entities, and temporal information come from noisy dialogue. Our PAL-style baseline induces facts and a program from raw history at query time. In contrast, LEDGER reuses previously extracted facts and induces only the program.

Inference-time structuring and context use. StructRAG selects and constructs a question-conditioned information structure before reasoning (Li et al., 2025). LEDGER instead consolidates sessions before the query, then executes a restricted program and overlays its result on retrieved text. Long-context studies also show that relevant information in a prompt is not necessarily used consistently (Liu et al., 2024). Our diagnostics measure this availability–integration gap without assuming a particular positional failure mechanism.

3 Method

3.1 Problem Formulation

Let $H = \{s_i\}_{i=1}^{n}$ contain sessions with role-marked turns, stable identifiers, and trusted metadata times t_i. For query q, the parent retriever $P_k(q, H)$ ranks chunks and returns their complete sessions. We target functions of facts distributed across sessions:

$$y = g\left(q, \bigcup_i \mathcal{I}(s_i)\right),$$

where $\mathcal{I}(s_i)$ denotes the information asserted by the user in session i. Extraction constructs reusable evidence. Program induction selects facts and specifies g. Execution evaluates the program. Parent retrieval supplies raw evidence alongside the computed result.

Table 1: The aggregation language. $F_\phi = \{f \in F : \phi(f) = 1\}$ is the filtered support set and π projects the requested field. Every operator is evaluated by code, not by a language model.

Operator	Denotation	Representative query
`COUNT_DISTINCT`	$\lvert\{\pi(f) : f \in F_\phi\}\rvert$	How many weddings have I attended?
`SUM`	$\sum_{f \in F_\phi} \textit{value}(f)$	How much did I spend on books in total?
`DIFF`	difference of two separately filtered operands	How much heavier am I than at my last check-up?
`DIVIDE`	ratio of two separately filtered operands	What did the mugs cost on average?
`LATEST`	$\pi(\arg\max_{f \in F_\phi}(\textit{time}, \textit{source}))$	Where do I live now?
`LIST`	order-preserving deduplication of $\pi(F_\phi)$	Which cities have I said I want to visit?
`NONE`	$\perp$, with the parent payload returned unchanged	Open-ended or single-session questions

3.2 Query-Agnostic Session Extraction

Each session is consolidated independently:

$$F_i = E(s_i; \varnothing), \qquad F = \bigcup_{i=1}^{n} F_i,$$

where $\varnothing$ denotes the absence of a future query. Each fact has the core representation

$$f = (\textit{predicate}, \textit{entity}, \textit{value}, \textit{unit}, \textit{time}, \textit{source}).$$

The prompt requests atomic user-supported claims, preserved negation, within-session deduplication, and final updated values, excluding assistant speculation. Additional audit fields record polarity, evidence-turn IDs, qualifiers, temporal scope, and update state.

The parser overwrites *source* with the session ID and *time* with trusted metadata. Closed-schema validation checks structure and source references but cannot establish entailment, temporal interpretation, or semantic deduplication. Extraction performs neither cross-session entity merging nor arithmetic. Session-local processing enables reuse across future queries and separate audits of extraction recall.

Store preparation. Accepted records receive stable fact IDs and lexically normalized entity strings for grouping or distinct counting. Normalization does not resolve aliases, event identity, or cross-session coreference. Its rules and the extraction settings appear in Appendix C.

3.3 Semantic Router

At query time, an optional candidate selector $C(q, F)$ presents a bounded set of facts while preserving their stable IDs. The router receives the query and fact content, but never the benchmark question-type label:

$$p = R(q, C(q, F)) = (op, \phi, \eta).$$

Here *op* selects the operation, ϕ filters support facts, and η specifies operand filters, aggregation axis, return field, and unit. Table 1 lists six aggregation operators and the abstention action `NONE`.

The router decides whether to admit a query, then binds its operator to facts and operand roles. Operator prediction alone is insufficient: filters and left–right assignments determine the result. The selector C bounds only the induction context, while semantic filters execute against the complete store F, independently of transient candidate indices.

The filter ϕ can constrain predicates and rendered fact text by include or exclude terms, numerical type and bounds, units, status and event role, and metadata time boundaries. Non-empty filter fields are conjoined, while alternatives inside a text-any field are disjoined.

Snapshot queries such as *latest* use trusted session timestamps. Unresolved relative windows route to `NONE` unless representable from metadata. Session timestamps identify snapshots, not necessarily the occurrence times of events described within them.

3.4 Deterministic Execution

The executor first selects $F_\phi = \{f \in F : \phi(f) = 1\}$ and then applies op as specified in Table 1:

$$z = \mathrm{Exec}(p, F).$$

Arithmetic operates on parsed numerical values without an LLM call. Distinct counts use normalized entities only when that axis is selected; `LATEST` breaks timestamp ties by source ID.

Execution is reproducible for fixed inputs. Answer correctness additionally requires correct facts, filters, operand roles, and numerical interpretation; the executor cannot repair their semantics.

3.5 Evidence-Preserving Raw-Text Overlay

The executor returns the value, unit, supporting fact IDs, and source IDs as a structured side channel $a(z)$. The memory payload sent to the answer model is

$$M(q) = \begin{cases} P_k(q, H) \oplus a(z), & op \neq \texttt{NONE} \ \wedge \ z \text{ is valid,} \\ P_k(q, H), & \text{otherwise,} \end{cases}$$

where $\oplus$ appends one aggregation record to the same k complete parent sessions. Explicit abstention, empty support, invalid numerical output, or `NONE` returns the same ordered parent-memory objects. A configured confidence threshold can also trigger fallback. The raw component remains the parent top-k, without privileged insertion of program-support sessions.

Preserving raw evidence and fallback payloads does not guarantee unchanged answers: incorrect side channels can mislead the answer model, and stochastic generation can differ on identical inputs. We measure answer-level regression empirically, including outside the aggregation slice.

4 Experimental Setup

4.1 Data and Splits

LongMemEval. We use the 500-example LongMemEval-S benchmark (Wu et al., 2025). A deterministic seed-42 split is stratified by (*question type*, *is abstention*), producing 98 development and 402 test examples. Diagnosis, prompt development, program correction, and model selection use the development partition; test questions, reference answers, and outcomes are excluded from these activities. Final results are reported on test402. Adaptive reuse makes development comparisons and p-values exploratory.

The aggregation slice comprises multi-session and knowledge-update questions: 26 and 15 in development, and 107 and 63 in test. An extraction probe uses 18 development questions whose top-15 retrieved chunks cover every annotated evidence-session ID. The probe groups those same chunks by session without expanding them; session-ID coverage does not establish supporting-span coverage.

Router training data. The V11.3 training-data contract uses source-disjoint programmatic supervision derived from Multi-Session Chat (Xu et al., 2022), excluding LongMemEval-shaped examples and development-oracle programs. The earlier SFT attribution configuration is a separate diagnostic, not the final router.

LoCoMo. LoCoMo (Maharana et al., 2024) serves as a secondary cross-dataset check rather than a primary benchmark. We evaluate the 1,540 non-adversarial questions (categories 1–4) from the ten released conversations, excluding the adversarial category, under the same judge and prompt as LongMemEval and without re-inducing the operator set. We report end-to-end accuracy and component ablations alongside LongMemEval.

4.2 Evidence Protocols

Final V11.3 evaluations are separated from earlier development diagnostics. The extraction probes and attribution configurations retain their saved fact stores. Semantic Router v2 and the corrected-program reference use program-support sessions, whereas final LEDGER uses Parent top-k sessions.

These diagnostic scores are not controlled gains over Parent RAG; Table 4 records the protocol differences.

The component analysis evaluates query-agnostic facts, deterministic execution, raw-text overlay, entity normalization, candidate selection, and restricted operator subsets. These ablations test which design choices contribute to end-to-end accuracy. The earlier development matrix instead contrasts raw and structured evidence with LLM aggregation or induced execution to diagnose where errors arise.

4.3 Baselines

Baselines vary evidence representation and aggregation mechanism.

- **No Memory** answers without conversation history.
- **Vanilla RAG** embeds and retrieves complete sessions with the MiniLM-based Sentence Transformers encoder `all-MiniLM-L6-v2`[1] (Wang et al., 2020; Reimers & Gurevych, 2019).
- **Chunked RAG** retrieves 200-token windows with 50-token overlap.
- **Parent RAG** retrieves chunks but returns their complete parent sessions; it is the principal retrieval baseline and fallback policy.
- **Long Context** supplies all sessions chronologically. **Long Context + CoT** also asks the model to enumerate evidence before answering (Wei et al., 2022).
- **PAL-style** induces facts and a program from all raw sessions at query time, executes with the same runtime, and overlays the result on raw text (Gao et al., 2023).
- **Fact Store + LLM** delegates aggregation over query-agnostic facts to the answer LLM.
- **LEDGER** uses the fact store, semantic router, deterministic executor, and parent-text overlay.

4.4 Metrics and Statistical Comparisons

Overall task accuracy is the mean of per-example LLM-judge correctness; the stored field `task_accuracy_macro` denotes this micro mean. We separately measure any-hit@k, evidence coverage, required-label recall, and execution-result equivalence. The LongMemEval judge prompt has a stored SHA-256 guard. Paired comparisons align sample IDs and queries; reported development McNemar tests are exploratory. Aggregate held-out scores alone do not establish statistical significance.

4.5 Fairness and Leakage Controls

Within each comparison, the answer model, judge, temperature, and answer-length limit are fixed. Final LEDGER and Parent RAG share the parent policy. Diagnostic fact-store cells share saved facts and candidate selection. Question type supports stratification, evaluation slices, and the judge rubric, but is never a router feature.

Input tokens and model calls are not matched across representations; side-channel prompts also differ. The automatic development PAL run counts three pipeline errors as incorrect and scores $22/41$. Two post-hoc source repairs yield $24/41$, excluded from the primary attribution cell. Thus the development matrix is descriptive rather than a strict causal 2×2 ablation. The provider does not honor the requested random seed.

4.6 Implementation Details

Development evaluations use Claude Haiku 4.5 for answers and judging, temperature zero, requested seed 42, and output limits of 256 and 10 tokens. Retrieval uses `all-MiniLM-L6-v2`. Vanilla

[1] `https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2`

Table 2: Task accuracy (%) on held-out LongMemEval and LoCoMo. The two LongMemEval question-type columns are subsets of test402. Bold denotes the best result in each column.

Method	LongMemEval test402 (n=402)	Multi-session (n=107)	Knowledge update (n=63)	LoCoMo (n=1,540)
No Memory	13.4	6.5	9.5	11.2
Vanilla RAG	61.7	57.9	60.3	52.4
Chunked RAG	58.5	60.7	54.0	50.8
Parent RAG	64.4	70.1	65.1	57.3
Long Context + CoT	62.2	50.5	63.5	54.6
PAL-style	62.9	53.3	63.5	55.1
Fact Store + LLM	65.7	62.6	68.3	56.9
LEDGER V11.3	**68.9**	**76.6**	**82.5**	**61.8**

RAG answers from ten sessions; the diagnostic any-hit and coverage metrics use five. Development fact-store diagnostics select at most 32 facts, capped at eight per source session.

The V11.3 source-first extraction contract uses `claude-haiku-4-5-20251001`, temperature zero, 4,096 output tokens, and zero application, SDK, or transport retries. Earlier diagnostic fact stores retain their original extraction settings. Appendix C records the full final configuration and distinguishes these implementation boundaries.

5 Results

5.1 End-to-End Performance

Table 2 reports the final system comparison. On LongMemEval test402, LEDGER reaches 68.9% accuracy, exceeding Parent RAG by 4.5 percentage points and the strongest overall baseline, Fact Store + LLM, by 3.2 points. Relative to Parent RAG, multi-session accuracy improves by 6.5 points and knowledge-update accuracy by 17.4 points. The reported totals imply the same pooled accuracy on the remaining question types. This aggregate equality does not establish that every type or individual example is unchanged, nor does it follow mechanically from the fallback design.

On LoCoMo, LEDGER reaches 61.8%, compared with 57.3% for Parent RAG, a 4.5-point improvement. This cross-dataset evaluation is consistent with the usefulness of the interface beyond the development benchmark. We do not interpret it as evidence that the supported operator language covers all conversational-memory tasks. The final comparisons are descriptive accuracy differences; we do not attach significance claims without corresponding paired uncertainty estimates.

Component ablations. Table 8 tests the final system's components. Removing the query-agnostic fact store reduces LEDGER to program induction over raw context, which is exactly the PAL-style configuration of Table 2; the two coincide by construction and score identically at 62.9%. Replacing deterministic execution with LLM aggregation reduces test402 accuracy to 66.2%. Removing the raw-text overlay reduces it to 64.2%, no better than the 64.4% of Parent RAG itself: substituting the computed record for the retrieved sessions discards the gain entirely, which is why the overlay appends rather than replaces. Removing entity normalization or restricting the operator set yields 65.7% and 66.4%, respectively. The corresponding LoCoMo variants also underperform the full system. These ablations support complementary roles for structured evidence, execution, and raw context; they do not imply additive or independent component effects.

5.2 Why Retrieval Coverage Is Insufficient

The development coverage audit identifies a concrete motivation for the architecture. Among 29 answerable retrieval-hit failures, 21 already contain every annotated evidence-session ID; for multi-session questions the count is $8/9$. At a coarser level, any-hit@5 is $87/93 = 93.5\%$, while answer accuracy from top-10 sessions is $58/93 = 62.4\%$, or $62/98 = 63.3\%$ over the complete development split. The different retrieval depths make this a descriptive contrast, not a stage-wise

Table 3: Exploratory attribution on dev41. Deterministic configurations include automatic program induction. PAL is fully automatic; its three pipeline errors count as incorrect. Protocol differences preclude a strict causal interpretation (Appendix D).

Context	Aggregation	Correct	Accuracy (%)
All raw sessions	LLM/CoT	21/41	51.2
All raw sessions	PAL-style deterministic	22/41	53.7
Query-agnostic facts	LLM	**26/41**	**63.4**
Query-agnostic facts	diagnostic SFT + deterministic	20/41	48.8

error decomposition. A task-informed audit classifies seven of the eight partial-coverage hit-but-wrong cases as temporal and one as plausibly coverage-fixable, without reassigning the six no-hit cases. That one case corresponds to approximately 1.1 percentage points on the 93 answerable questions; it is neither a retrieval intervention estimate nor an upper bound on possible retrieval gains.

On the 18-example session-ID-covered probe, Chunked RAG answers 12 questions correctly, compared with 14 for query-aware extraction and 15 for query-agnostic extraction followed by manually specified aggregation. The query-agnostic extractor recovers $32/33 = 97.0\%$ required labels, with the three remaining task errors assigned to aggregation in the audit. This selected probe supports the feasibility of reusable extraction, not a statistically established advantage over query-aware extraction. Figure 3 summarizes these development diagnostics.

5.3 Representation and Program-Induction Diagnostics

The exploratory dev41 configurations in Table 3 separate raw versus structured evidence and LLM versus executable aggregation. Fact Store + LLM scores $26/41$, compared with $21/41$ for raw-context LLM/CoT; the exact paired McNemar test is not significant ($p = 0.227$). Fully automatic PAL-style induction scores $22/41$ ($p = 1.000$ versus raw-context LLM/CoT). Repairing two failed traces post hoc gives $24/41$, which is excluded from the primary matrix. The early diagnostic SFT router scores $20/41$, below the $26/41$ obtained by delegating aggregation over the same fact store to the answer LLM. Read against the final ablation in Table 8, where replacing deterministic execution instead costs five questions, this sign reversal is the most informative observation in the matrix. Deterministic execution does not add accuracy; it removes variance around whichever program is induced. When induction is weak, as in this diagnostic configuration whose programs are execution-equivalent on only $4/13$ corrected targets (Section 6), committing to the program is worse than letting the answer model hedge over the retrieved facts. When induction is reliable, the same commitment pays. The executable interface is therefore an amplifier of program quality rather than an independent source of gain, which is why we report routing fidelity separately from end-to-end accuracy. These configurations differ in prompts, contexts, and call structure, so the matrix is not a fully controlled factorial experiment. The development aggregation slice contains 41 questions, so a single example moves its accuracy by 2.4 points. Development and held-out effect sizes are therefore not distinguishable at this resolution, and we do not interpret differences between them.

Privileged expressivity references. On the same dev41 slice, Parent RAG scores $29/41$, the hand-written Semantic Router v2 scores $33/41$, and a corrected-program reference scores $34/41$ (Table 5). The latter two use program-support raw sessions rather than the Parent top-k payload; the corrected reference additionally uses gold source IDs and answer-conditioned program correction. They provide diagnostic evidence that selected questions can be expressed and executed, not controlled gains over Parent RAG or a mathematical upper bound.

In the earlier equivalence audit, Semantic Router v2 matches all 13 corrected executable targets, but only $21/41 = 51.2\%$ across the full slice. Some matches reflect identical unavailable-value outputs, so output agreement is weaker than semantic program correctness.

6 Failure Analysis and Limitations

A stage-wise error taxonomy. We distinguish evidence availability, local extraction, program induction, deterministic execution, and answer synthesis or judging. A failure is assigned to a stage

only when the corresponding artifact supports that attribution; otherwise it remains ambiguous. Two interface measures follow: the fraction of queries assigned a non-NONE program, and execution-result equivalence conditional on such a route. Low values of the first arise from conservative routing, incomplete facts, or an unsupported computation; low values of the second point to errors within accepted routes.

Operator selection is not semantic binding. For the early diagnostic SFT router in the $20/41$ attribution cell, the operator is correct on $11/13$ corrected targets, but only $4/13$ programs are execution-equivalent. Typical errors select incomplete fact sets, bind the wrong event roles to operands, or confuse historical and current values. This diagnostic motivates semantic filters and full-store execution; it is not an evaluation of the final learned router.

An open schema remains difficult to normalize. A three-example full-store audit yields 683 facts and 365 distinct predicate strings, with 75.6% of predicate types occurring once. This small audit illustrates schema fragmentation, not its prevalence across the benchmark. Lexical normalization does not resolve arbitrary predicate paraphrases or event coreference.

The computational scope is limited. The DSL supports distinct counts, sums, differences, ratios, latest-state selection, and lists. Audited examples also require multiplication, utterance-grounded time expressions, or event coreference outside the current metadata-time policy. More operators would not by themselves repair missing facts or incorrect binding, so routing coverage is not a measure of expressivity alone. For NONE or invalid programs the system returns the same ordered Parent RAG payload, but a valid yet semantically wrong program can still append misleading information: the unchanged pooled accuracy on non-target types is an empirical aggregate, not a no-regression theorem.

Evaluation limits. Development probes are small and selected, and privileged references use different context contracts from the final system. Execution equivalence can count coincident outputs without establishing semantic correctness, and some gold answers or judge decisions remain ambiguous. The provider does not honor the requested seed, and our aggregate comparisons carry no paired confidence intervals or repeated-run uncertainty; we therefore claim no statistical significance. Cost and latency are not quantified.

7 Conclusion

Retrieving the right sessions does not ensure that an agent can answer from them. On an audited development split, most retrieval-hit failures already contain every annotated evidence session, which places a computational problem downstream of retrieval rather than inside it. LEDGER addresses that problem by consolidating each session into reusable, source-linked facts without observing future queries, routing a query into a restricted aggregation program, executing that program deterministically, and appending the result to the retrieved sessions rather than replacing them. On held-out LongMemEval and on LoCoMo, this improves the aggregation-heavy question types while leaving pooled accuracy on the remaining types unchanged.

The mechanism is narrower than the improvement suggests. Deterministic execution removes variance around an induced program, but it does not supply the program. Our diagnostics show that operator selection is largely solved while semantic fact and operand binding is not, and that an executable interface built on a weak router performs worse than no interface at all. The open question for conversational memory is therefore not only which facts can be recalled, but which operations can be reliably induced and executed over them.

Reproducibility Statement

The LongMemEval development and test partitions are fixed by a seeded stratified split, and every reported comparison holds the answer model, judge prompt, decoding temperature, and output limits constant. Appendix C specifies the fact schema, retrieval depth, router backbone and supervision, training hyperparameters, checkpoint-selection rule, and the scoring of abstentions and failures; Appendix D lists the controlled and uncontrolled factors in each comparison; and Appendix E reports the component ablations. Router supervision is synthesised programmatically from MSC and is source-disjoint from all evaluation data. The provider does not honour the requested seed, so reruns may differ slightly. Code, the synthetic supervision generator, and the extracted fact stores will be released upon publication.

AI Use Statement

Generative AI tools, including OpenAI Codex, were used to inspect and summarize implementation and experiment artifacts; assist qualitative error analysis and interpretation of results; prepare manuscript figures and tables; support literature discovery and summarization; and draft or edit manuscript text and visualization code. GPT Image was used for an earlier diagram draft that was subsequently replaced by code-native vector artwork. Language models are also components of the evaluated memory systems, as specified in the experimental setup. The authors are responsible for verifying the reported measurements, citations, interpretations, and all AI-assisted material, and take responsibility for the final content of the work.

References

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready AI agents with scalable long-term memory. arXiv preprint arXiv:2504.19413, 2025. URL https://arxiv.org/abs/2504.19413.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: Program-aided language models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 10764–10799. PMLR, 2023. URL https://proceedings.mlr.press/v202/gao23f.html.

Zhanghao Hu, Qinglin Zhu, Runcong Zhao, Di Liang, Hanqi Yan, Yulan He, and Lin Gui. Beyond RAG for agent memory: Retrieval by decoupling and aggregation. arXiv preprint arXiv:2602.02007, 2026. URL https://arxiv.org/abs/2602.02007.

Christopher Latimer, Nicolò Boschi, Andrew Neeser, Chris Bartholomew, Gaurav Srivastava, Xuan Wang, and Naren Ramakrishnan. Hindsight: Structured agent memory that retains, recalls, and reflects. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 275–285. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-demo.27.

Zhuoqun Li, Xuanang Chen, Haiyang Yu, Hongyu Lin, Yaojie Lu, Qiaoyu Tang, Fei Huang, Xianpei Han, Le Sun, and Yongbin Li. StructRAG: Boosting knowledge intensive reasoning of LLMs via inference-time hybrid information structurization. In *International Conference on Learning Representations*, pp. 36107–36124, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/5975754c7650dfee0682e06e1fec0522-Abstract-Conference.html.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of LLM agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13851–13870, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.747.

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

Peijun Qing, Fobo Shi, and Soroush Vosoughi. UTILMEM: Benchmarking evidence utilization in long-term conversational memory. arXiv preprint arXiv:2608.30508, 2026. URL https://arxiv.org/abs/2608.30508.

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. URL https://arxiv.org/abs/2501.13956.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3982–3992. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1410.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems*, volume 33, pp. 5776–5788, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Zilong Wang, Hao Zhang, Chun-Liang Li, Julian Martin Eisenschlos, Vincent Perot, Zifeng Wang, Lesly Miculicich, Yasuhisa Fujii, Jingbo Shang, Chen-Yu Lee, and Tomas Pfister. Chain-of-table: Evolving tables in the reasoning chain for table understanding. In *International Conference on Learning Representations*, pp. 55587–55610, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/f53fd88a4340063ecd258c0ae9948b40-Abstract-Conference.html.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837, 2022. doi: 10.52202/068431-1800.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. LongMemEval: Benchmarking chat assistants on long-term interactive memory. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/d813d324dbf0598bbdc9c8e79740ed01-Abstract-Conference.html.

Jing Xu, Arthur Szlam, and Jason Weston. Beyond goldfish memory: Long-term open-domain conversation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5180–5197, Dublin, Ireland, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.356.

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, pp. 17577–17604. Curran Associates, Inc., 2025. doi: 10.52202/085713-0593.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3911–3921, Brussels, Belgium, 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1425.

Haozhen Zhang, Haodong Yue, Tao Feng, Quanyu Long, Jianzhu Bao, Bowen Jin, Weizhi Zhang, Xiao Li, Jiaxuan You, Chengwei Qin, and Wenya Wang. Learning query-aware budget-tier routing for runtime agent memory. arXiv preprint arXiv:2602.06025, 2026. URL https://arxiv.org/abs/2602.06025. Accepted at the International Conference on Machine Learning, 2026.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. MemoryBank: Enhancing large language models with long-term memory. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19724–19731, 2024. doi: 10.1609/aaai.v38i17.29946.

A Operator Semantics by Example

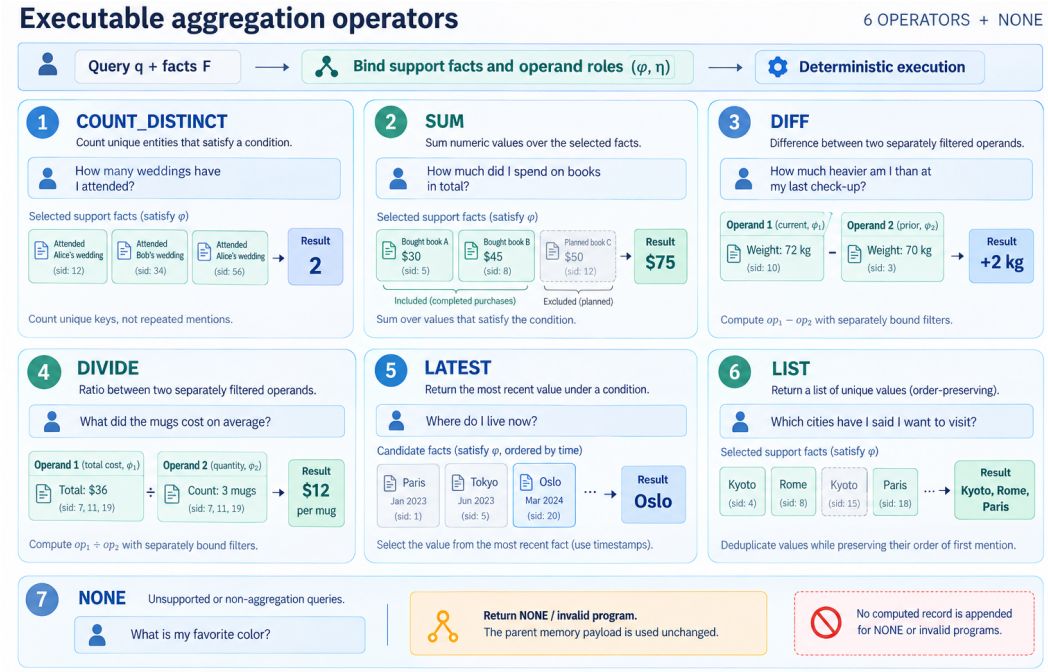

Figure 2: Worked instances of the aggregation language of Table 1. Each panel shows a query, the support facts selected by the filter, their source session identifiers, and the executed result. The `DIFF` and `DIVIDE` panels show the separately bound operand roles whose misassignment is the dominant program-induction failure in Section 6.

B Diagnostic Protocols and Supporting Results

Table 4 distinguishes the development diagnostics from the final evaluation. Figure 3 summarizes the retrieval, extraction, and attribution evidence. Table 5 reports the development-specialized and privileged references; their overlays differ from that of the final system.

Table 4: Evidence protocols. Development diagnostics isolate individual interfaces and use different contracts from the final system. "Legacy facts" are the saved diagnostic stores, rather than outputs of the strict extractor in Section 3.2.

Protocol and slice	Evidence	Program and answer context	Interpretation
Retrieval diagnosis; dev98, 93 answerable	Vanilla sessions; separate gold-session audit	Answering uses top 10 sessions; availability uses top 5	Descriptive availability–accuracy contrast
Extraction probes; 18 multi-session	Same top-15 chunks, grouped by source session	Query-aware or blind extraction; manually specified aggregation	Extraction feasibility under session-ID coverage
Attribution; dev41	All raw sessions or legacy query-agnostic facts	LLM aggregation, automatic PAL-style induction, or a diagnostic SFT router	Exploratory paired contrasts, not a causal factorial ablation
Semantic Router v2; dev41	Legacy facts	Hand-written, development-specialized router; program-support raw sessions	DSL and filter expressivity under a diagnostic overlay
Corrected-program reference; dev41	Legacy facts plus privileged source and answer information	Answer-conditioned program correction; program-support raw sessions	Privileged feasibility reference, not an upper bound
LEDGER V11.3; test402 and LoCoMo	Query-agnostic facts and retrieved parent sessions	Automatic router; Parent top-k plus computed side channel	Held-out task accuracy and cross-dataset evaluation

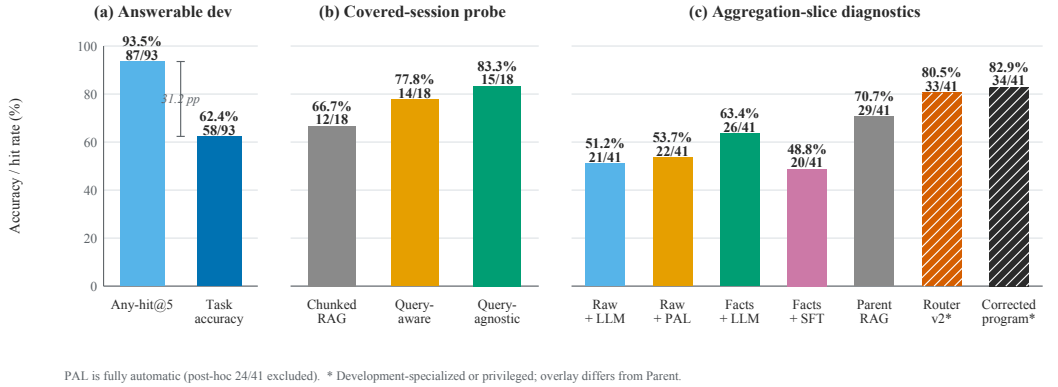

Figure 3: Development diagnostics. (a) On the same 93 answerable examples, retrieving at least one gold session at $k = 5$ is more frequent than a correct answer produced from the top-10 session payload; this is a descriptive contrast, not a matched stage decomposition. (b) On a slice where retrieval covers every annotated session ID, query-aware and query-agnostic extraction operate on the same chunks grouped by session and use manually specified downstream aggregation. (c) The first four bars are exploratory attribution configurations; PAL is the fully automatic $22/41$ run and excludes two post-hoc repairs. The final three bars are context-setting references. Hatched bars are development-specialized or privileged diagnostics, not held-out learned-router results.

Table 5: Aggregation-slice diagnostics on the same 41 LongMemEval development examples. The Parent row is context only: both program-based rows use the program-support-session overlay rather than the Parent top-k payload, so cross-row differences are not controlled gains over Parent RAG. Semantic Router v2 is development-specialized; the corrected-program reference additionally uses privileged program information. Neither is a held-out result for the final V11.3 pipeline.

System	Overall	Multi-session	Knowledge update
Parent RAG (context reference)	29/41 (70.7)	19/26 (73.1)	10/15 (66.7)
Semantic Router v2[†]	33/41 (80.5)	20/26 (76.9)	13/15 (86.7)
Corrected-program reference[‡]	34/41 (82.9)	20/26 (76.9)	14/15 (93.3)

Values in parentheses are percentages. †: development-specialized heuristic; ‡: privileged program reference.

C Implementation Details

Table 6: Implementation settings. Earlier diagnostic fact stores retain their artifact-specific settings; they are distinguished from the V11.3 extraction contract.

Component	Setting
Fact schema	Core six-tuple plus polarity, evidence-turn, qualifier, temporal-scope, and update-state audit fields
Provenance	Stable session ID and first-turn metadata time overwrite generated source/time fields
Router inputs	Query and fact content; no benchmark question-type label
Operator set	`COUNT_DISTINCT`, `SUM`, `DIFF`, `DIVIDE`, `LATEST`, `LIST`, `NONE`
Temporal boundary	Metadata snapshots allowed; unresolved relative windows fall back
Entity normalization	Lowercase, article/punctuation/space normalization, and simple plural heuristic
Answer/judge	Claude Haiku 4.5, temperature 0, requested seed 42; 256/10 maximum output tokens
Diagnostic fact stores	Artifact-specific earlier extraction settings and saved outputs; not the strict current schema
V11.3 extraction contract	`claude-haiku-4-5-20251001`, temperature 0, 4,096 output tokens, and zero application, SDK, or transport retries
Retrieval encoder	`all-MiniLM-L6-v2`
Chunked/parent windows	200 tokens with 50-token overlap
Confirmed retrieval diagnosis	Top 10 sessions for answering; any-hit and coverage at $k = 5$
Extraction feasibility probe	Same top-15 chunks grouped by session
Dev41 diagnostic selector	Top 32 facts, capped at 8 per source session
Final V11.3 configuration	
Parent retrieval	Ten parent sessions in the answer payload; any-hit and coverage are measured at $k = 5$
Router backbone	`phi-3-mini-4k-instruct`
Router supervision	980 programmatically synthesised query–program rows from MSC dialogues, split into 700 training, 140 calibration, and 140 gate rows; source-disjoint from all evaluation data
Router training	QLoRA rank 16 over a 4-bit NF4 base with double quantisation; learning rate 5×10^{-5}; effective batch size 64 (per-device 1, gradient accumulation 16, four devices)
Checkpoint selection	Selected on the 140-row synthetic calibration split (135 rows evaluated once per epoch) by the harmonic mean of execution-equivalent program accuracy and support F1. Neither LongMemEval nor the held-out synthetic gate split informs selection
Confidence threshold	not enabled in the reported configuration
Scoring of abstentions and failures	A `NONE` route, an empty support set, or a null value returns the unmodified parent payload, which is scored normally. An exception in the policy, answer, or judge call is scored as incorrect and retained in the denominator, as is a judge response that does not begin with *yes* or *no*
Restricted operator subsets	`DIFF` disabled

The diagnostic numerical parser interprets an explicit ratio as the second value divided by the first and maps a numerical interval to its midpoint. These deterministic conventions do not establish that a parsed quantity has the semantics intended by the query.

D FAIRNESS AUDIT

Table 7: Fairness controls and unresolved comparability gaps for the development attribution matrix.

Property	Status	Detail
Sample IDs and queries	Controlled	All cells use the same 41 examples
Answer model and judge	Controlled	Same model family, temperature, and judge prompt
Question-type access	Controlled	Withheld from router; evaluation only
Parent policy	Outside this matrix	The final LEDGER–Parent comparison shares a parent policy; the four attribution configurations do not test that comparison
Fact store	Controlled	Fact-based cells share precomputed facts
Input-token budget	Not controlled	Full history, facts, and parent-plus-side channel differ
Answer prompt	Partially controlled	Side-channel policies use specialized instructions
API calls	Not controlled	PAL-style uses an additional induction call
Repair policy	Not controlled	The primary automatic PAL cell counts one malformed-program and two unknown-source rows as incorrect; the two source repairs are reported only as a post-hoc diagnostic
Stochasticity	Residual	Provider ignores requested seed

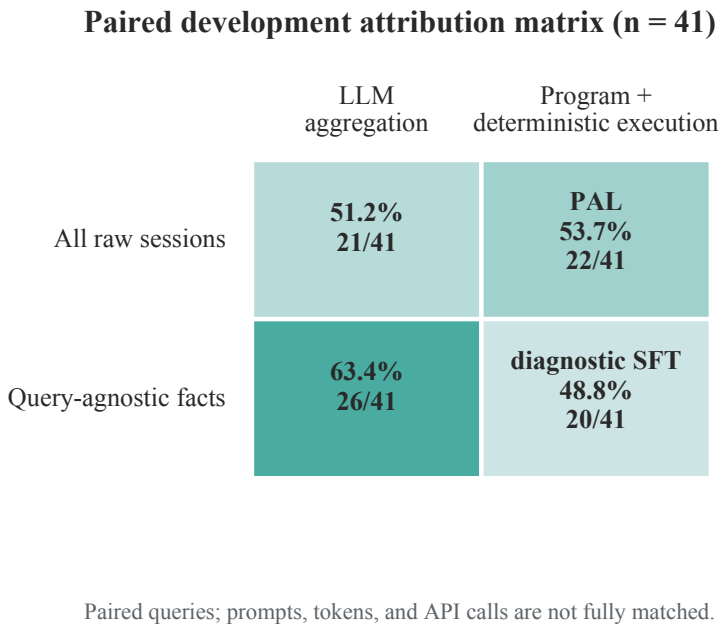

Figure 4: The completed development attribution matrix as a two-axis view. The PAL cell is the fully automatic $22/41$ run. Every cell uses the same 41 queries, but the protocol differences in Table 7 prevent a strict causal interpretation.

E ABLATION RESULTS

Table 8: Ablation results. Dev aggregation reports correct answers out of 41, with accuracy (%) in parentheses; the remaining columns report accuracy (%). Lexical and dense candidate selection score 4.9 and 2.4 percentage points below the hybrid selector used in the full system, on the development slice. The executor ablation retains the full pipeline's router, prompt, and overlay, and therefore differs from the Fact Store + LLM baseline in Table 2, which uses its own answer prompt.

Variant	Dev aggregation	Test402	LoCoMo
Full V11.3 LEDGER	**32/41 (78.0)**	**68.9**	**61.8**
Without query-agnostic facts	22/41 (53.7)	62.9	55.1
LLM instead of deterministic executor	27/41 (65.9)	66.2	58.4
Without raw-text overlay	29/41 (70.7)	64.2	57.5
Without entity normalization	28/41 (68.3)	65.7	58.0
Restricted operator subsets	30/41 (73.2)	66.4	59.2