MEMTACT: COST-OPTIMAL CONTROL OF FINDABILITY AND RESIDENCY IN LONG-TERM LLM MEMORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-term memory allows large language model agents to reuse past information without repeatedly reading an ever-growing interaction history. Yet memory systems must solve two fundamentally different problems: they must make relevant evidence easy to find, and they must make frequently used information cheap to consume. These objectives are often entangled in a single compressed representation, even though retrieval and direct reading impose different requirements and costs. Moreover, the right representation cannot be determined once at storage time: future queries reveal both how information is searched for and which evidence is worth keeping readily accessible. We introduce MemTact, a use-oriented framework that treats navigation and information residency as distinct cost decisions. Navigation uses source-linked semantic hierarchies to connect concrete evidence with queries at broader levels of abstraction. Residency adapts to observed query usage, weighing retrieval, recovery, reading, and update costs. The framework distinguishes recoverable, findable, and resident service states; the current implementation combines a fixed hierarchical index with query-driven residency updates. Because queries can require complementary evidence, we formulate joint evidence placement and derive an exact maximum-weight-closure solution under conjunctive demands, additive state costs, and no hard capacity constraint. Across 460 questions from RULER, EventQA, and LongMemEval, MemTact achieves 61.52% accuracy, exceeding RAPTOR and A-MEM by 19.35 and 11.09 percentage points, respectively. It uses 62.6% fewer online API tokens than A-MEM and 66.4% fewer than QueryLink, demonstrating a favorable balance between answer accuracy and serving cost.

1 INTRODUCTION

Long-term memory allows language models to reuse past information without repeatedly processing the entire history. Yet how information should be found and how much of it should be read are different questions. Source records describe particular people, activities, and events, whereas future queries may refer to categories, relationships, or exact details. A representation that matches a query may omit the evidence needed to answer it; a representation that contains sufficient evidence may repeatedly expose the reader to irrelevant detail. When original records remain available for recovery, memory design therefore concerns more than what survives compression. It must determine how information is located, consumed, and recovered over successive queries.

Prior work improves different parts of this process. RAPTOR constructs multiple abstraction levels through recursive clustering and summarization (Sarthi et al., 2024). SeCom studies memory granularity and improves retrieval using topical segments and compression-based denoising (Pan et al., 2025). LLMLingua-2 learns to retain information under a reduced text budget (Pan et al., 2024). These methods provide useful components for retrieval and compression. However, selecting a better static representation alone does not determine how much recovered evidence should remain available for repeated reading. We ask: **when original evidence remains recoverable, how should memory support access at different semantic granularities while adapting the evidence repeatedly presented to the model to reduce cost across a query sequence?**

This question involves two related granularity mismatches. The first concerns retrieval. A representation of a specific event may provide a weak match for a category-level query, while a generalized

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.

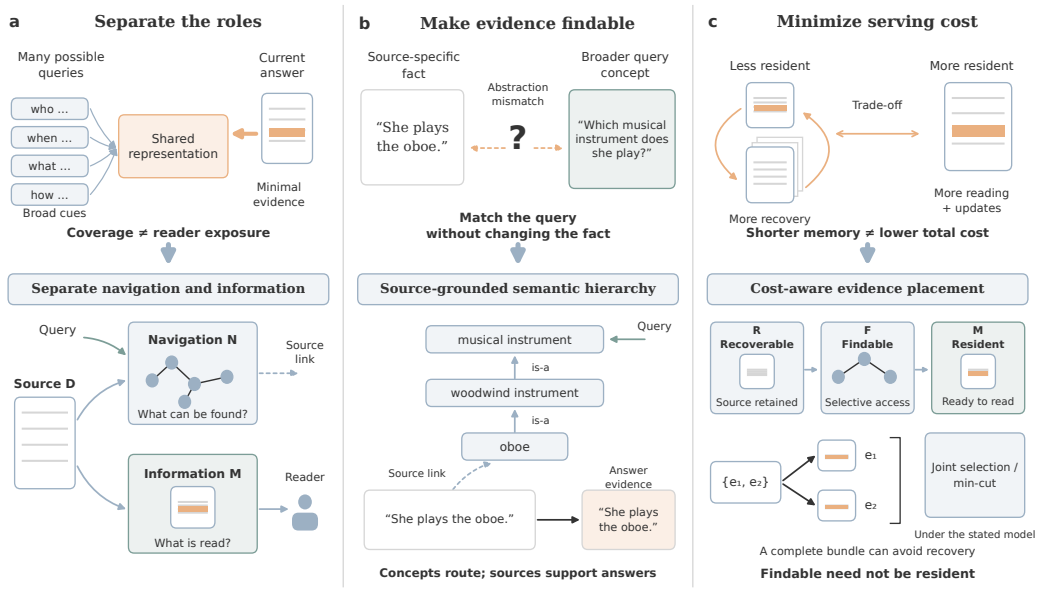


Figure 1: **Three design principles of MemTact.** (a) Navigation supports broad query coverage, while information memory controls the evidence read for each answer. (b) Source-grounded concept chains bridge abstraction mismatches: concepts route queries and original text supports answers. (c) Evidence placement trades recurring reading and update costs against recovery. Complementary evidence is modeled jointly, with minimum-cut selection for the objective in Section 3.4.

representation may obscure the details that distinguish similar events. We construct several retrieval views of each source block, linking literal source phrases to broader concepts. Queries can match these views at different abstraction levels, while every view identifies the same underlying evidence unit. Additional semantic access paths therefore need not increase the content presented to the answer model: concepts guide retrieval, and source text supplies evidence.

The second mismatch concerns repeated consumption. Evidence needed for the current answer is not necessarily evidence worth retaining for every subsequent access. Recovery reveals that the current compact representation did not meet a particular demand. Retaining the entire recovered block would also make its other details recur whenever that block is retrieved. The value of retaining a sentence thus depends on both how often it is needed and how often its block is read, as well as the recovery cost it can avoid. The appropriate amount of resident evidence depends on the access stream induced by retrieval and cannot be determined by a single offline compression pass. MemTact uses source sentences cited in the final answer as admission candidates and adjusts their available capacity using recovery cost and additional reading burden.

We express this organization through the distinction between *findability* and *residency*. Figure 1 summarizes these design principles. Findability is supported by retrieval views that lead to original records. Residency determines which evidence is included in the representation read first when a block is accessed. MemTact builds source-linked hierarchical views and initial compact representations offline. Online, it retrieves blocks, reads their current compact representations, recovers requested originals when needed, and updates resident evidence from citations and cost feedback. Retrieval views remain fixed while sentence residency changes across queries. This allows the system to adapt what the reader sees while preserving its existing semantic access paths.

Realizing this design requires both representation learning and online control. We adapt a shared generation and embedding model to make source phrase selection, concept generation, and retrieval discrimination compatible. Sparse combinations of concept levels provide alternative views while retaining the complete source text. The online controller connects cited recovery evidence to the recurring reading cost of its block, admits complete source sentences within a bounded budget, and

selectively removes unused fragments under admission pressure. Stable source mappings connect the two components, separating the roles of generated concepts, retrieved blocks, and cited evidence.

Our contributions are threefold. First, we formulate recoverable memory as a cost-control problem across queries, distinguishing semantic access granularity from evidence residency and identifying their dependence on the retrieval-induced access distribution. Second, we develop source-linked hierarchical retrieval through joint generation and embedding adaptation and sparse multi-view construction. Third, we introduce citation-driven residency control that adjusts block budgets from observed recovery and reading costs, allowing evidence recovered for one query to become reusable memory for later queries. Together, these components implement a complete retrieval, reading, recovery, and update process.

2 Related Work

Retrieval representations. RAPTOR (Sarthi et al., 2024) recursively clusters and summarizes source chunks, retrieving both summaries and original leaves across abstraction levels. Its hierarchy aggregates information across chunks; MemTact instead constructs source-phrase–concept views within each block and returns that block's current readable evidence. QueryLink (Hu et al., 2026) aligns queries and memories through multiple semantic views and returns their corresponding dialogue chunks. It already separates retrieval cues from answering evidence. MemTact builds on this distinction through explicit concept chains, while additionally adapting how much recovered source evidence remains available for later reading. Related approaches organize memory through hierarchical consolidation (Rezazadeh et al., 2025; Li et al., 2026) or hierarchical and multi-granularity graphs (Huang et al., 2025; Zhan et al., 2026). Our retrieval comparison therefore examines alternative access structures, while residency control addresses their subsequent reading cost.

Adaptive memory. A-MEM (Xu et al., 2025) constructs structured notes, links related memories, and evolves existing representations when new memories arrive. MemTact focuses on a different update signal: evidence cited after answering queries. It admits recovered source sentences and adjusts block budgets from observed recovery and recurring reading costs, keeping retrieval views fixed. Query-driven adaptation also appears in REALM's retrieval-feedback reconsolidation (Song et al., 2026), while Mem-T learns construction and retrieval policies with access to raw turns (Yue et al., 2026). These approaches establish the value of feedback and recovery; MemTact studies the narrower decision of which recovered evidence should remain resident across subsequent accesses.

Compression and cumulative cost. Query-aware compression (Jiang et al., 2024; Nagle et al., 2024) trades prompt length against answer quality, while measured end-to-end savings also depend on compression overhead (Kummer et al., 2026). Pichay (Mason, 2026) already considers cumulative retention and recovery costs in context-window paging. MemTact applies this cost perspective to persistent source-linked memory: a retained sentence adds reading cost whenever its block is accessed, but avoids recovery only when its evidence is needed. This distinction motivates block-level budgets and citation-driven sentence admission.

3 Methods

3.1 Overview

As illustrated in Figure 1, MemTact separates how evidence is found from what the reader repeatedly consumes. Given source blocks $\mathcal{D} = \{D_i\}_{i=1}^{N}$, each block has fixed retrieval views V_i and a dynamic readable representation M_i^t. Source-linked semantic hierarchies connect queries and sources at different abstraction levels. After each query, observed evidence use and costs guide residency updates: evidence worth repeated reading stays resident, while other evidence remains findable through the fixed index and recoverable from the original source. Because several evidence items may be needed together, we also formulate residency as joint selection over evidence bundles.

For queries $q_1, \ldots, q_T$, let ρ denote the retrieval rule and π a residency policy using only current and past observations. Our lifecycle objective is

$$\begin{aligned} \min_{\rho,\pi} \quad & C_{\text{build}} + \mathbb{E}\left[\sum_{t=1}^{T} (C_{\text{ret},t} + C_{\text{read},t} + C_{\text{rec},t} + C_{\text{upd},t})\right] \\ \text{subject to} \quad & \text{QA}(\rho, \pi) \geq \tau. \end{aligned} \tag{1}$$

The terms account for construction, retrieval, reading, recovery, and updates in common cost units. The quality target τ is assessed by independent question-answering evaluation. This objective makes the trade-off explicit: shorter resident context saves reading cost but may increase recovery, while retaining evidence incurs repeated reading and update costs. MemTact learns retrieval offline and adapts residency online; the objective specifies the goal rather than a guarantee of global optimality.

3.2 Source-linked hierarchical retrieval

A query may describe a general concept while its supporting source mentions a specific instance. To bridge this mismatch, we select informative phrases that occur verbatim in each source block and generate two successive broader concepts conditioned on their local meanings:

$$p_{ij} \longrightarrow a_{ij}^{(1)} \longrightarrow a_{ij}^{(2)}.$$

Each chain retains its source phrase and excerpt, linking abstract retrieval cues to concrete evidence. The hierarchies are local to individual blocks.

We encode alternative abstraction paths as separate views rather than merging all concepts into one vector. Besides the original block, V_i contains views that prefix the complete source text with individual phrase–concept annotations or pairs of annotations. For normalized encoder e_θ, retrieval combines the original-text match with the strongest view match:

$$\begin{aligned} s(q_t, D_i) &= (1-\alpha) e_\theta(q_t)^\top e_\theta(D_i) + \alpha \max_{v \in V_i} e_\theta(q_t)^\top e_\theta(v), \\ R_t &= \text{TopK}_i \; s(q_t, D_i). \end{aligned} \tag{2}$$

Results are deduplicated by source block. The reader receives the selected blocks' current representations M_i^t; concept annotations serve retrieval only.

We adapt GritLM-7B with shared LoRA parameters for grounded phrase selection, contextual concept generation, and retrieval (Muennighoff et al., 2025; Hu et al., 2022). Training combines definition-conditioned WordNet supervision, multi-positive retrieval with same-conversation negatives, and embedding preservation (Miller, 1995). Training details are provided in Appendix A. The resulting index remains fixed during online use.

3.3 Query-driven residency updates

We initialize M_i^0 by compressing each source block and mapping retained fragments to stable source-sentence identifiers. For each query, the reader first answers from the retrieved compact representations. If these are insufficient, it can request the corresponding originals for one additional reading. Final citations identify the source sentences used in the answer and supply candidates for subsequent residency updates.

Residency trades recurring reading cost against avoidable recovery cost. A retained sentence is read whenever its block is retrieved, whereas its recovery benefit arises only when that sentence is needed. Citation frequency alone therefore does not determine whether retention is economical. Updates should favor evidence whose expected recovery savings exceed its added reading and update costs, using only current and past observations.

The online controller implements this trade-off through block-level admission budgets. Observed recovery usage attributable to cited blocks supports growth; repeated reading of added text counters that growth. Recovered, cited sentences can replace their compressed fragments with complete source text. Unused resident fragments may be replaced when needed for a feasible admission. These updates change M_i^t without modifying V_i or D_i, preserving access to evidence that is not resident. Budget accounting and replacement rules are implementation details, separate from the joint selection model below.

3.4 JOINT SELECTION OF EVIDENCE BUNDLES

Evidence items can be complementary: retaining one item may not avoid a recovery call if another required item is still missing. We therefore model the saving of a complete evidence bundle, rather than assigning the same recovery saving independently to every item.

With navigation fixed, we model the residency-dependent terms of Equation equation 1 through a restricted placement objective. Let S be the selected set of fully resident evidence items. Over a common decision horizon, c_f denotes item f's additive reading and state-transition cost. A demand requires a conjunctive bundle A_j, and $w_j \geq 0$ is its expected avoidable recovery cost. Omitting costs independent of S, joint selection minimizes

$$J(S) = \sum_{f \in S} c_f + \sum_j w_j \, \mathbf{1}[A_j \not\subseteq S]. \tag{3}$$

Each recovery saving is counted once, when all required items are resident. Shared evidence incurs its residency cost once even when it supports several bundles. Thus, jointly useful evidence can be selected even when no item is worthwhile alone.

For fixed cost estimates, conjunctive demands, additive item costs, and no hard shared capacity constraint, this objective reduces to maximum-weight closure (**?**). An evidence node has weight $-c_f$, a bundle node has weight w_j, and each bundle depends on all its evidence nodes. A minimum cut gives the optimal joint selection for this estimated objective. This formulation captures evidence complementarity; the online block-feedback controller above is a separate practical policy, not an exact solver for this objective.

4 EXPERIMENTS AND RESULTS

Our evaluation follows the two decisions in MemTact: finding relevant evidence and adapting the evidence retained for subsequent queries. We first measure retrieval quality, then compare complete systems in answer accuracy and online token usage. Controlled dynamic-versus-fixed and component experiments examine how query experience translates into better answers and more efficient evidence reuse. Training-stage results are presented in Appendix A.

4.1 EXPERIMENTAL SETUP

Data and protocols. Retrieval evaluation uses 631 evidence-annotated questions over 424 LoCoMo memory blocks. Candidates are restricted to each question's source conversation, and each configuration returns at most five complete blocks within a 512-token text budget. System evaluation comprises 460 selected questions over 2,885 blocks: 100 from RULER, 300 from EventQA, and 60 from LongMemEval. The EventQA set combines questions from three context-length variants against the same full source context. We further evaluate dynamic residency on an independently initialized 300-question EventQA stream and run a three-repeat component ablation on 15 frozen fact-reuse questions.

Systems and metrics. All answering experiments use `gpt-6-luna` and the same source blocks. MemTact uses the final adapted GritLM-7B for hierarchical retrieval and initializes readable memory at 25% target retention. RAPTOR, A-MEM, and QueryLink use base GritLM-7B with their respective memory and reading procedures. Retrieval metrics are mean per-question evidence recall and all-evidence recall, the proportion of questions for which every annotated evidence block is retrieved. Answer accuracy is computed by normalized reference-answer containment. Online API tokens sum reported input and output tokens across query processing, including query generation where applicable. Reader tokens sum compact-reading and recovery calls. Appendices B–D provide configurations, scoring, and repeat protocols.

4.2 EVIDENCE RETRIEVAL

MemTact retrieves more supporting evidence within the same text budget. The final retrieval configuration achieves 69.00% evidence recall and 62.28% all-evidence recall on LoCoMo, improving over base GritLM original-text retrieval by 6.97 and 6.02 percentage points, respectively (Table 1).

Both configurations return original source blocks under the same five-block/512-token budget. The improvement therefore gives the reader more relevant evidence without expanding the returned-text allowance.

Table 1: **Evidence retrieval on 631 LoCoMo questions.** The comparison is between base original-text retrieval and MemTact's final adapted encoder with hierarchical views. Metrics are percentages; higher is better.

Retrieval configuration	Evidence recall ↑	All-evidence recall ↑
Base GritLM, original text	62.03	56.26
MemTact, final adaptation + hierarchical views	**69.00**	**62.28**

The hierarchy provides multiple semantic access paths to the same evidence: its 8,166 views cover 424 source blocks, with at most 23 views per block. Views participate in ranking, while the selected block supplies the answer context. This separates broader semantic matching from the amount of evidence read. Appendix A reports the training-stage comparisons that produce the shared generation and retrieval model.

4.3 System-level accuracy and online usage

MemTact combines higher accuracy than RAPTOR and A-MEM with substantially lower online usage than A-MEM and QueryLink. Across 460 questions, MemTact answers 283 correctly, reaching 61.52% accuracy (Table S4, Appendix B). This exceeds RAPTOR by 19.35 percentage points and A-MEM by 11.09 points. Its 1.743 million online API tokens are 62.6% lower than A-MEM and 66.4% lower than QueryLink. Figure 2(a) places these results in a common accuracy–usage plane: MemTact improves both measures relative to A-MEM and occupies an intermediate point between RAPTOR's lower usage and QueryLink's higher accuracy.

The largest gains appear on EventQA. MemTact answers 197 of 300 questions correctly, compared with 87 for RAPTOR and 130 for A-MEM. QueryLink reaches 230 correct answers. On RULER, RAPTOR and A-MEM achieve 87 and 86 correct answers, respectively, compared with MemTact's 68; on LongMemEval, the four systems range from 16 to 20 correct answers (Figure 2(b)). These task-level results identify EventQA as the main source of MemTact's aggregate advantage. The following experiment traces how its dynamic memory develops within an EventQA query stream.

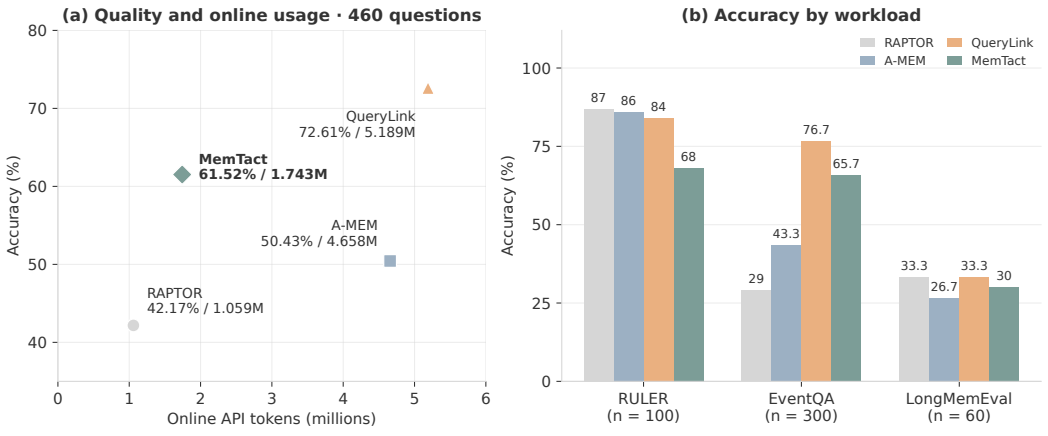


Figure 2: **System-level accuracy and online usage.** (a) Accuracy versus online input-plus-output API tokens across 460 questions; the upper left combines higher accuracy with lower usage. (b) Accuracy by workload. MemTact improves both aggregate accuracy and online usage relative to A-MEM, while EventQA contributes the largest accuracy gains over RAPTOR and A-MEM.

4.4 DYNAMIC RESIDENCY ACROSS SUCCESSIVE QUERIES

Dynamic residency improves answer accuracy at comparable token usage. We hold hierarchical top-five retrieval, the reader, and the initial compact representations of 1,044 EventQA blocks fixed. Both arms start from the same initial memory with independent empty response caches. Fixed memory can recover source evidence for the current question; dynamic memory additionally retains cited evidence and updates admission budgets after each query. Over 300 questions, dynamic residency increases correct answers from 184 to 191, a 2.33-percentage-point gain, with a 0.89% increase in Reader tokens (Table 2).

Table 2: **Dynamic versus fixed readable memory on the 300-question EventQA stream.** Both arms use identical retrieval and initial compression, with source recovery available throughout.

Measure	Fixed readable memory	Dynamic residency
Correct answers ↑	184/300	**191/300**
Accuracy (%) ↑	61.33	**63.67**
Compact-reading tokens	1,157,617	1,159,359
Recovery-reading tokens	266,286	277,218
Total Reader tokens	1,423,903	1,436,577
Queries requiring recovery	87	95
Original-block openings	203	205
Sentence completions / evictions	0 / 0	37 / 0

The advantage becomes clearer as query experience accumulates. After 100 queries, both arms have answered 43 questions correctly. At query 200, dynamic memory leads 122 to 121; by query 300, the lead grows to 191 versus 184 (Figure 3(b)). In the last 100 queries, it answers six more questions correctly. This pattern is consistent with progressive adaptation to observed evidence demand: information recovered and retained during earlier interactions becomes available to later queries. The growing late-stream advantage provides empirical support for the role of accumulated query feedback in MemTact.

The dynamic arm completes 37 source sentences, expanding resident content from 198,649 to 199,526 locally measured tokens. Across the stream, compact-reading API usage increases by only 1,742 tokens, while recovery accounts for the remaining 10,932-token increase. At comparable reading usage, dynamic residency therefore supports more accurate answers by updating the evidence available to subsequent queries (Figure 3).

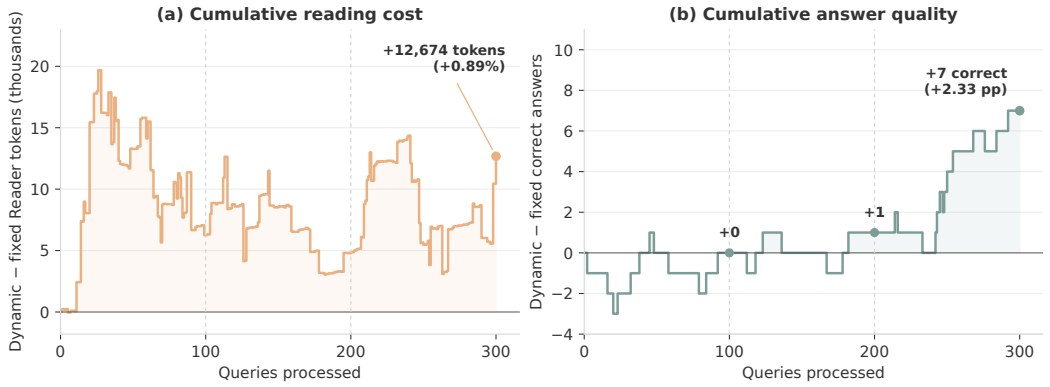


Figure 3: **Better answers at comparable token usage.** Cumulative dynamic-minus-fixed differences in Reader tokens (a) and correct answers (b) over 300 ordered EventQA queries. The accuracy advantage is 0, 1, and 7 correct answers at queries 100, 200, and 300. The final gain is 2.33 percentage points with 0.89% additional Reader usage. Dashed lines mark the three 100-query groups; curves retain the observed query order without smoothing.

4.5 COMPONENT ABLATION AND EVIDENCE REUSE

The ablation targets fact reuse to examine how MemTact benefits from accumulated query experience. Dynamic residency gains value when later queries need evidence accessed earlier. When queries are dispersed across unrelated facts, retained evidence has fewer opportunities for reuse, leaving less room for dynamic optimization. We therefore use the 15 fact-reuse questions identified in a discovery run to form a small, controlled reuse scenario.

We enable or disable hierarchical retrieval and dynamic residency in a 2×2 design. The target list is frozen before three independent repeats, which characterize variation within the same reuse scenario. In each repeat, dynamic configurations process the original query prefixes to reconstruct the preceding memory state; all configurations are then compared on the same target evaluations. Table 3 reports target-query usage, with the prefix protocol detailed in Appendix D.

Table 3: **Component ablation on 15 frozen fact-reuse questions, repeated three times.** Counts and tokens are summed over the 45 target evaluations per configuration.

Config.	Hierarchy	Dynamic	Correct /45 ↑	Target Reader tokens ↓	Recovery queries	Block openings
A	No	No	27	130,999	17	30
B	Yes	No	**28**	127,984	12	30
C	No	Yes	25	118,981	10	16
D: MemTact	Yes	Yes	**28**	**111,537**	7	8

With hierarchy enabled, dynamic residency reduces target Reader tokens by 12.85%, from 127,984 to 111,537, with both configurations achieving 28 correct target evaluations (Figure 4(a)). Recovery queries fall from 12 to 7 and source-block openings from 30 to 8 (Table 3). The reduction appears in all three repeats: 13.98%, 15.57%, and 8.36%, with dynamic-versus-fixed correct counts of 8 versus 9, 9 versus 9, and 11 versus 10 (Figure 4(b)). MemTact cites previously retained facts in 22 target evaluations, directly showing that earlier admissions become evidence for later answers.

The complete configuration combines accurate retrieval with efficient reuse. Removing hierarchical retrieval from dynamic memory lowers the correct count from 28 to 25 and increases target tokens from 111,537 to 118,981. Among the four configurations, MemTact achieves the joint-highest correct count and the lowest target usage. Together with the full-stream results, the ablation illustrates the two stages of dynamic memory use: queries first supply feedback and evidence for adaptation, and later accesses benefit from the accumulated resident content.

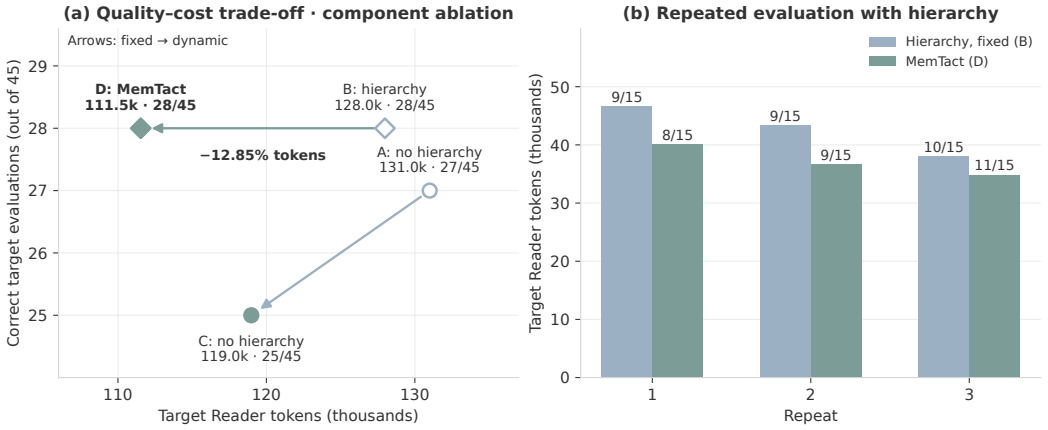


Figure 4: **Component ablation on fact-reuse queries.** (a) Quality–cost scatter for the four configurations. Arrows from A to C and B to D show the effect of enabling dynamic residency without and with hierarchy, respectively. Open and filled markers denote fixed and dynamic memory. (b) Per-repeat usage of hierarchy-enabled fixed memory (B) and MemTact (D), with correct counts above bars. The same 15 frozen targets are evaluated three times. MemTact reduces aggregate target usage by 12.85%, with 28 correct evaluations in both B and D. Prefix queries form dynamic states and are reported separately in Appendix D.

5 Discussion

The idealized analysis in Appendix E illustrates why maintaining access to evidence and repeatedly exposing that evidence to the reader can warrant different decisions. Its state-space comparison motivates this separation, but does not quantify the empirical performance of hierarchical retrieval or the feedback-budget controller. The implemented method specifies these decisions through source-linked retrieval views, block-level feedback, and citation-driven sentence admission. Our code is available at `https://anonymous.4open.science/r/MemTact`.

References

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*, 2022. URL `https://www.microsoft.com/en-us/research/publication/lora-low-rank-adaptation-of-large-language-models/`.

Xuxian Hu, Zhu Teng, Wei Zhang, Ming He, and Jianping Fan. QueryLink: Leveraging Query-Memory Alignment for Long-Term Reasoning in LLM Agents. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 15608–15621. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.765. URL `https://aclanthology.org/2026.findings-acl.765/`.

Haoyu Huang, Yongfeng Huang, Junjie Yang, Zhenyu Pan, Yongqiang Chen, Kaili Ma, Hongzhi Chen, and James Cheng. Retrieval-Augmented Generation with Hierarchical Knowledge. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 6044–6060. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-emnlp.321. URL `https://aclanthology.org/2025.findings-emnlp.321/`.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. LongLLMLingua: Accelerating and Enhancing LLMs in Long Context Scenarios via Prompt Compression. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1658–1677. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.91. URL `https://aclanthology.org/2024.acl-long.91/`.

Cornelius Kummer, Lena Jurkschat, Michael Färber, and Sahar Vahdati. Prompt Compression in the Wild: Measuring Latency, Rate Adherence, and Quality for Faster LLM Inference. In *Advances in Information Retrieval*, volume 16483 of *Lecture Notes in Computer Science*, pp. 257–271. Springer, 2026. doi: 10.1007/978-3-032-21289-4_17. URL https://link.springer.com/chapter/10.1007/978-3-032-21289-4_17.

Kai Li, Xuanqing Yu, Ziyi Ni, Yi Zeng, Yao Xu, Zheqing Zhang, Xin Li, Jitao Sang, Xiaogang Duan, Xuelei Wang, Chengbao Liu, and Jie Tan. TiMem: Temporal-Hierarchical Memory Consolidation for Long-Horizon Conversational Agents. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 21700–21720. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.1091. URL https://aclanthology.org/2026.findings-acl.1091/.

Tony Mason. The Missing Memory Hierarchy: Demand Paging for LLM Context Windows, 2026. URL https://arxiv.org/abs/2603.09023. arXiv preprint arXiv:2603.09023.

George A. Miller. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11): 39–41, 1995. URL https://wordnet.princeton.edu/citing-wordnet.

Niklas Muennighoff, Hongjin Su, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. Generative Representational Instruction Tuning. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/70cf215430492f7d34830a24e744b3f1-Abstract-Conference.html.

Alliot Nagle, Adway Girish, Marco Bondaschi, Michael Gastpar, Ashok Vardhan Makkuva, and Hyeji Kim. Fundamental Limits of Prompt Compression: A Rate-Distortion Framework for Black-Box Language Models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3009. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/ac8fbba029dadca99d6b8c3f913d3ed6-Abstract-Conference.html.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. LLMLingua-2: Data Distillation for Efficient and Faithful Task-Agnostic Prompt Compression. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 963–981, 2024. doi: 10.18653/v1/2024.findings-acl.57. URL https://aclanthology.org/2024.findings-acl.57/.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Xufang Luo, Hao Cheng, Dongsheng Li, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Jianfeng Gao. SeCom: On Memory Construction and Retrieval for Personalized Conversational Agents. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/e56f394bbd4f0ec81393d767caa5a31b-Abstract-Conference.html.

Alireza Rezazadeh, Zichao Li, Wei Wei, and Yujia Bao. From Isolated Conversations to Hierarchical Schemas: Dynamic Tree Memory Representation for LLMs. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/0382cb76309820f71c6eacd47b36ce71-Paper-Conference.pdf.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/8a2acd174940dbca361a6398a4f9df91-Abstract-Conference.html.

Yuanyi Song, Yukai Wang, Xinbei Ma, Zhihui Fu, Jianghao Lin, Weiwen Liu, Jun Wang, Huarong Deng, Yong Yu, and Weinan Zhang. Retrieval-Driven Memory Reconsolidation for Long-Term LLM Agents, 2026. URL https://arxiv.org/abs/2609.16053. arXiv preprint arXiv:2609.16053.

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.

Yanwei Yue, Boci Peng, Xuanbo Fan, Jiaxin Guo, Qiankun Li, and Yan Zhang. Mem-T: Densifying Rewards for Long-Horizon Memory Agents, 2026. URL https://arxiv.org/abs/2601.23014. arXiv preprint arXiv:2601.23014.

Bihao Zhan, Zongsheng Cao, Jie Zhou, Bo Zhang, and Liang He. FlowRAG: Synergizing Explicit Reasoning via Frequency-Aware Multi-Granularity Graph Flow, 2026. URL https://arxiv.org/abs/2606.17856. arXiv preprint arXiv:2606.17856.

A TRAINING THE SHARED GENERATION AND RETRIEVAL MODEL

A.1 TRAINING STAGES AND CHECKPOINT SELECTION

We adapt GritLM-7B in four stages: parent generation, joint generation and retrieval, retrieval refinement, and source-phrase supervision. The stages progressively equip one model with concept generation, source discrimination, and grounded access-point selection.

Table S1: **Training configuration.** Step counts and epoch numbers refer to the stage in the corresponding row.

Stage	Data and objective	Learning rate	Training extent	Selected checkpoint
Parent generation	WordNet: 34,961 train / 4,383 dev / 4,417 test; direct-parent generation, q/v LoRA	10^{-4}	1 epoch; effective batch 16; 2,186 steps	Epoch 1
Joint adaptation	Generation, retrieval contrast, and embedding preservation	10^{-5}	2 epochs; 188 steps	Epoch 2
Retrieval refinement	747 original training queries; 1,045 positive relations; 4 hard and 2 random negatives per sampled query	10^{-5}	374 steps	Step 94
Source-phrase supervision	58 training / 14 phrase-validation examples; phrase CE + 10×preservation + 0.2×parent CE	2×10^{-5}	4 epochs; 60 steps	Epoch 4

WordNet supervision raises valid structured output from 0% to 100% on the fixed 200-example development probe, with 61.00% exact reference-set match. Exact match is 58.80% on the 4,417-example test split. WordNet is partitioned by child lemma; concepts can also occur as parents in other examples. The metric measures agreement with the supplied parent-name sets.

The final stage starts from refinement step 94. Selection among the initialization and four epoch checkpoints maximizes exact annotated phrase F1, breaking ties by target-token cross-entropy, subject to an original-text retrieval recall drop of at most 0.01 on local validation queries. The selected epoch-4 checkpoint reaches phrase F1 of 0.1571 on 14 examples and original-text recall of 0.57947 on 123 validation queries. Phrase references are task-assistant annotations. The validation source is excluded from final-stage phrase training and was used in earlier retrieval training; these measurements track the final stage's incremental adaptation.

A.2 RECONCILING GENERATION WITH RETRIEVAL

Joint training restores retrieval while adapting the model for generation. With original-text indexing fixed, generation-only SFT lowers evidence recall from 62.03% to 58.45%. Joint generation, retrieval, and preservation training recovers it to 61.98%, within 0.05 percentage points of the base encoder (Table S2; Figure S1). The same pattern appears in all-evidence recall, which changes from 56.26% to 52.77% and then to 56.10%. This motivates joint optimization when the adapted model must serve both generation and embedding.

Table S2: **Retrieval through the initial adaptation stages.** All rows use the same original-text index, 631 LoCoMo questions, and five-block/512-token return budget. Metrics are percentages.

Encoder	Evidence recall ↑	All-evidence recall ↑
Base GritLM	62.03	56.26
Generation-only SFT	58.45	52.77
Joint adaptation	61.98	56.10

A.3 PRESERVATION DURING SUBSEQUENT ADAPTATION

Refinement retains closely similar STS and SciFact scores while improving the sampled WordNet exact match. Both SciFact configurations use the same 512-token limit, truncating 1,142 corpus

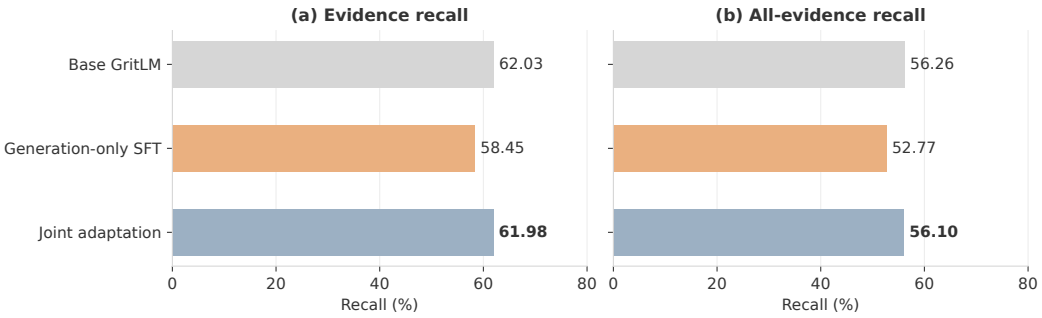


Figure S1: **Retrieval preservation during generative adaptation.** The original-text index and return budget are fixed across the base encoder, generation-only SFT, and joint adaptation. Joint training restores evidence recall from 58.45% to 61.98% and all-evidence recall from 52.77% to 56.10% on the same 631 LoCoMo questions.

Table S3: **Preservation diagnostics.** The first four rows compare the joint checkpoint with retrieval refinement step 94; the final three describe source-phrase supervision from that reference.

Stage and measure	Before	After
Refinement: STS Spearman	0.844155	0.843166
Refinement: SciFact nDCG@10	0.782688	0.782879
Refinement: SciFact Recall@100	0.970000	0.970000
Refinement: WordNet exact-set match, 512 examples	0.556641	0.560547
Phrase supervision: original-text recall, 123 local queries	0.580149	0.579472
Phrase supervision: mean document cosine to step 94	—	0.999392
Phrase supervision: mean query cosine to step 94	—	0.999671

inputs. During source-phrase supervision, document and query cosine similarities to the refinement checkpoint remain above 0.999, and local original-text recall changes by 0.000677. Table 1 in the main text evaluates the final adapter together with its hierarchical retrieval views.

B SYSTEM CONFIGURATIONS AND SCORING

The system comparison uses 1,210 RULER blocks, 1,044 EventQA blocks, and 631 LongMemEval blocks with source-isolated retrieval. All methods use `gpt-6-luna` for answering, with a maximum output allowance of 8,192 tokens. The external baselines use base GritLM-7B and configured input/evidence caps of 24,000/12,000 `cl100k` tokens. The retrieval-only experiment uses its separate 512-token return budget.

Table S4: **Answer accuracy and online API usage across 460 questions.** Workload columns give correct-answer counts. Online usage is the sum of input and output API tokens, in millions.

Method	RULER /100 ↑	EventQA /300 ↑	LongMemEval /60 ↑	Total /460 ↑	Accuracy (%) ↑	Online API tokens (M) ↓
RAPTOR	**87**	87	**20**	194	42.17	**1.059**
A-MEM	86	130	16	232	50.43	4.658
QueryLink	84	**230**	**20**	**334**	**72.61**	5.189
MemTact	68	197	18	283	61.52	1.743

RAPTOR. The implementation follows recursive UMAP/GMM clustering with the supplied source blocks as leaves, at most five layers, requested 100-token summaries with a 1,024-token API cap, and collapsed all-layer top-five retrieval. Its clustering target is 3,500 tokens.

A-MEM. The robust upstream path constructs notes in source order, with conditional memory evolution and linking. Consolidation occurs every 100 evolution events. Base GritLM-7B replaces the retriever. Memory construction uses source records; evaluation questions and reference answers are not added to memory.

QueryLink. The fixed-block adaptation retains memory summaries, query views, top-three retrieval per view, union of retrieved items, two neighboring items on either side, and a two-call answering procedure. Evidence blocks are retained in chronological order within the evidence cap. Each query has two expansion calls and two reading calls. Retrieval follows the upstream path that excludes the mixed index.

The initial 260-question evaluation comprises 100 RULER, 100 EventQA, and 60 LongMemEval questions. The extension adds 100 questions each from EventQA rows 7 and 12, answered against row 2's full context, in the order row 2, row 7, row 12. Baselines reuse saved indices and prefix outputs. MemTact uses its third recorded 260-question pass and replays the saved prefix to reconstruct its state before continuing. The archived 260-question source assignments contain 161 entries labeled "train" and 99 labeled "test"; the evaluation here uses the combined selected workload.

Scoring lowercases text, removes punctuation and articles, and normalizes whitespace before testing reference-answer containment. EventQA uses the parser's extracted answer; the other tasks use the saved answer or its parsed form. LongMemEval is evaluated with this same lexical rule. The 1,840 final answers in Table S4 were rescored against the supplied references and reconciled with the question IDs.

C ONLINE USAGE AND QUERY-STREAM PROGRESSION

Table S5: **Exact online API usage.** The 200-question extension is included in the total. Tokens include API-reported input and output.

Method	Online API tokens	Extension-200 API tokens	Online calls
RAPTOR	1,058,765	529,359	460
A-MEM	4,657,553	2,386,751	460
QueryLink	5,189,301	2,491,867	1,840
MemTact	1,742,588	860,665	595

QueryLink's online total comprises 538,071 query-expansion tokens and 4,651,230 reading tokens. Usage sums historical and extension call ledgers once. Replayed saved responses do not add new API usage, and cached input tokens remain part of reported input tokens.

The independent 300-question residency experiment starts from fresh compact memories and independent caches with frozen top-five rankings. Its dynamic arm answers 191 questions correctly; the system-comparison continuation answers 197 EventQA questions correctly. Each result belongs to its stated protocol. The paired stream contains 171 questions answered correctly by both arms, 96 answered incorrectly by both, 20 correct only with dynamic residency, and 13 correct only with fixed memory.

Table S6: **Accuracy progression within the paired query stream.** Each group follows the original order, and cumulative counts retain all preceding queries.

Queries	Fixed in group	Dynamic in group	Cumulative fixed	Cumulative dynamic	Cumulative advantage
1–100	43	43	43	43	0
101–200	78	79	121	122	+1
201–300	63	69	184	191	+7

This temporal breakdown locates the growing advantage within a single EventQA stream. Figure 3 plots its full, unsmoothed trajectory. Both arms follow the same query order; dynamic memory carries the updated state from each query into the next.

D FACT-REUSE ABLATION PROTOCOL

The 15-query target set defines a focused scenario in which later questions can reuse previously accessed facts. Three independent repeats evaluate the components under the same reuse opportunities. Within each dynamic run, the preceding queries supply the evidence and feedback carried into the targets.

The target set contains eight RULER and seven EventQA questions, selected as the union of questions citing previously admitted facts in the discovery runs of the dynamic configurations. The list is frozen before three new repeats. Each dynamic repeat processes the first 64 RULER and 243 EventQA questions in their original order: 15 targets and 292 additional prefix questions. Every repeat, configuration, and source starts with a fresh state and isolated response cache. Static configurations evaluate the same 15 targets directly.

Table S7: **Target performance and prefix Reader usage by repeat.** A/B/C/D correspond to Table 3. Prefix queries form the dynamic state; main-text target usage covers the frozen target evaluations. Tokens include input and output.

Repeat	Config.	Correct /15	Target tokens	Prefix tokens	Workflow tokens
1	A	9	42,676	0	42,676
1	B	9	46,588	0	46,588
1	C	8	37,515	1,230,233	1,267,748
1	D	8	40,074	1,216,508	1,256,582
2	A	8	48,953	0	48,953
2	B	9	43,383	0	43,383
2	C	9	43,840	1,239,459	1,283,299
2	D	9	36,628	1,185,671	1,222,299
3	A	10	39,370	0	39,370
3	B	10	38,013	0	38,013
3	C	8	37,626	1,231,655	1,269,281
3	D	11	34,835	1,195,761	1,230,596

Across the three repeats, C processes 3,701,347 prefix tokens and D processes 3,597,940, in addition to their target totals of 118,981 and 111,537 tokens. Each configuration evaluates the same 15 targets three times. All 180 target records and 1,752 additional prefix records have been reconciled with the summaries. Prior-admission citations occur in 23 target evaluations for C and 22 for D, using only admissions completed before the current target query.

Relative to B, D reduces target tokens by 13.98%, 15.57%, and 8.36% across the three repeats. The corresponding correct counts are 8 versus 9, 9 versus 9, and 11 versus 10. These repeated measurements characterize evidence reuse on the frozen target set.

E IDEALIZED MOTIVATION FOR SEPARATING FINDABILITY AND RESIDENCY

This appendix uses an idealized state model and a controlled simulation to motivate separating retrieval access from readable residency. It does not evaluate the implemented hierarchical retrieval and feedback-budget controller in Section 3.

E.1 RECOVERABLE MEMORY STATE

For each source memory unit D_i, the idealized model represents two derived states:

$$X_i^t = (N_i^t, M_i^t), \tag{4}$$

where N_i^t is a source-linked *navigation memory* used to recover D_i, and M_i^t is a compact *information memory* presented to the reader. The source D_i is immutable.

At the evidence level, we represent state with two binary variables,

$$x_f^t = (z_f^t, r_f^t), \tag{5}$$

where $z_f = 1$ denotes that evidence f is findable through the maintained retrieval surface and $r_f = 1$ denotes that f is resident in the reader-facing memory. We assume residency implies access, so $r_f \leq z_f$. This yields three relevant states:

$$(0,0) : \text{not maintained as findable and non-resident}, \tag{6}$$

$$(1,0) : \text{findable but non-resident}, \tag{7}$$

$$(1,1) : \text{findable and resident}. \tag{8}$$

A shared representation that ties access to residency restricts the feasible set to

$$\mathcal{X}_{\text{shared}} = \{(0,0), (1,1)\}, \tag{9}$$

whereas the decoupled model permits

$$\mathcal{X}_{\text{decoupled}} = \{(0,0), (1,0), (1,1)\}. \tag{10}$$

Therefore,

$$\mathcal{X}_{\text{shared}} \subset \mathcal{X}_{\text{decoupled}}, \tag{11}$$

and for any common cost function C,

$$\min_{X \in \mathcal{X}_{\text{decoupled}}} C(X) \leq \min_{X \in \mathcal{X}_{\text{shared}}} C(X). \tag{12}$$

The inequality is strict whenever $(1,0)$ is cheaper than both shared alternatives. If c_f^N is the cost of maintaining access, g_f the expected recovery cost while non-resident, m_f the cost of not maintaining access, and c_f^R the recurring residency cost, a sufficient strict-gain condition is

$$c_f^N + g_f < \min\big(m_f,\ c_f^N + c_f^R\big). \tag{13}$$

E.2 LIFECYCLE-COST OBJECTIVE

MemTact optimizes cumulative serving cost rather than compression ratio or retrieval accuracy in isolation. For query q_t, we decompose cost as

$$C_t = C_{\text{retrieval}}^t + C_{\text{read}}^t + C_{\text{recovery}}^t + C_{\text{update}}^t. \tag{14}$$

The control objective over a horizon T is

$$\min_{\pi}\ \mathbb{E}_\pi \left[\sum_{t=1}^{T} C_t\right] \qquad \text{s.t.} \qquad \text{QA}(\pi) \geq \tau, \tag{15}$$

where π is the memory-control policy and τ is an answer-quality target.

This formulation makes adaptation an amortization problem. A state transition may increase the cost of the current query through rewriting or index maintenance, but lower later cost by avoiding

Table S8: Optimal evidence states in the controlled state-space experiment. The findable-but-nonresident state $(1, 0)$ is strictly better than both shared alternatives in 20.65% of instances.

State	Meaning	Fraction
$(0, 0)$	unavailable / non-resident	8.33%
$(1, 0)$	findable / non-resident	20.83%
$(1, 1)$	findable / resident	70.83%
Strict $(1, 0)$ gain over shared states		20.65%

repeated recovery. For a single evidence item f over an estimated future window of H relevant memory accesses, let p_f be the probability that f is needed, r_f the avoidable recovery cost, ℓ_f the incremental resident read cost per access, and u_f the one-time update cost. Under this simplified model, promotion is beneficial when

$$Hp_f r_f > u_f + H\ell_f. \tag{16}$$

Equivalently,

$$p_f > \frac{\ell_f}{r_f} + \frac{u_f}{Hr_f}. \tag{17}$$

A fixed frequency threshold is therefore justified only when the relevant cost terms are sufficiently homogeneous. Under heterogeneous costs, equal-frequency evidence can require different decisions.

E.3 Controlled state-space analysis

We first ask whether decoupling exposes a genuinely useful state rather than merely enlarging the feasible set formally. We evaluate 10,800 controlled instances spanning reuse count, retrieval difficulty, source-recovery cost, resident size, and navigation cost. Table S8 reports the optimal state distribution.

No instance violates Equation 12. More importantly, strict gain is concentrated in a coherent region: evidence is costly enough to keep accessible, but its reuse is insufficient to justify recurring residency cost. This illustrates the value of separating findability and residency under the specified idealized costs. The comparison concerns the explicitly coupled state constraint, rather than an external memory method, and does not measure the performance of the implemented controller.