Cost and accuracy of long-term graph memory in distributed LLM-based multi-agent systems

BENEDICT WOLFF bjpwolff@kth.se

JACOPO BENNATI jbennati@kth.se

January 12, 2026

Abstract

Distributed multi-agent systems use large language models to enable collaborative intelligence while preserving privacy, yet systematic evaluations of long-term memory under network constraints remain limited. This study presents a flexible testbed comparing mem0, a vector-based memory framework, and Graphiti, a graph-based knowledge graph, using the LOCOMO long-context benchmark. Experiments were conducted under unconstrained and constrained network conditions, measuring computational, financial, and accuracy metrics. Results indicate that mem0 significantly outperforms Graphiti in efficiency, with faster loading times, lower resource consumption, and minimal network overhead, while accuracy differences are not statistically significant. Applying a statistical pareto efficiency framework, mem0 is identified as the optimal choice that balances cost and accuracy in DMAS.

Keywords: distributed multi-agent systems, long-term memory, large language models, benchmarking, vector database, knowledge graph

Open-source research: the source code of the testbed, experimental results as well as notebooks for analyzing and evaluating the results are available at: https://github.com/wolffbe/dmas-long-context-memory.

Contents

List of acronyms

AES accuracy-efficiency score

AI artificial intelligence

API application programming interface

AWS Amazon Web Services

CI Confidence Interval

CpSQ cost per successful query

CPU central processing unit

DMAS distributed multi-agent system

IDK i don't know

JSON JavaScript object notation

LLM large language model

LTS long-term support

MAS multi-agent system

Q&A questions and answers

RAG retrieval-augmented generation

RAM random access memory

SLM small language model

TCO total cost of ownership

TCP transmission control protocol

USD United States Dollar

vCPU virtual central processing unit

WSL Windows Subsystem for Linux

1 Introduction

Artificial intelligence (AI) agents based on large language model (LLM) are capable of methodically translating goals into actions, accessing tools interfacing third-party services, and learning from past experiences autonomously retrieved from memory [1].

By connecting multiple LLM-based agents, multi-agent system (MAS) achieve a higher form of collective intelligence that has been applied in various fields and industries [2–8], e.g. poetry [9], construction [10] and finance [11].

distributed multi-agent system (DMAS) emerged from a need to mitigate privacy risks as centralized parties in MAS process sensitive user information [12]. The vision is a global network of DMAS that collaborate using local private memory to complement knowledge-intensive human labor, research and innovation [13].

Long-term memory benchmarks such as LOCMO exist to evaluate the contextual learning ability of LLM [14] and MAS [15]. Various memory providers have already proven to enhance contextual learning using such benchmarks [15–18]. Recent research has also been focused on identifying best practices and design patterns to increase the cost-effectiveness of MAS [19].

However, systematically evaluating the contextual learning capabilities of novel memory frameworks combined with newly researched best practices and design patterns for MAS remains challenging. Memory frameworks are often assessed solely on accuracy and token usage without integration into actual MAS, overlooking additional computational costs and latency. New design patterns are typically evaluated in domain-specific implementations that lack repeatable and comparable performance tests across different MAS and benchmarks. Finally, no research exists so far that systematically examines and enhances long-term memory in DMAS, particularly across various network scenarios.

1.1 Theoretical framework and literature study

A review of 67 papers addressing LLM-based agents, MAS and DMAS revealed five common fields of research across all three agent types, namely technology, optimization, safety and security, networking and memory. No existing literature on safety and security for LLM-based agents and memory for DMAS was identified during this review. This chapter highlights the theoretical framework and literature that provide the foundation to this, presumably, first research of memory for DMAS.

LLM-based agents

The four components of LLM-based agents are *perception*, the ability to interact with an environment and receive feedback, *planning*, the ability to divide tasks into smaller sub-tasks, *memory*, the ability to persist an understanding of sources and operations over time, and *action*, the ability to take necessary steps to reach a given goal [20]. A second approach similarly divides an LLM-based agent's structure into profile, perception, mutual interaction, and evolution [21].

Optimizing agent effectiveness ranges from parameter-driven methods such as fine-tuning to parameter-free approaches such as prompt engineering and external knowledge retrieval [22]. A third parameter-free approach is chain-of-thought reasoning enabling an LLM to structure complex thought processes [23]. Additionally, the cost of an agent is optimized by decomposing multi-LLM selection challenges into a relaxed form [24], extracting and caching plan templates from successful task completions [25] or adjusting the complexity of a given task [26].

Use cases for agents in networking are network slicing [27], viewport prediction, adaptive bitrate streaming, cluster job scheduling [28], optimizing networks in Open Radio Access Network environments [29], mitigating data exfiltration scenarios [30], and managing software-defined networks [31]. Another use case is the distributed inference of LLM at the edge of a network by dividing the model into partitions, which are then assigned to different devices [32], for example, mobile phones [33]. The placement and migration of such distributed LLM have also been managed using ant colony algorithms [34].

Literature on agent memory consists of frameworks and benchmarks. Memory frameworks are categorized into vector storages such as VIMBank [35] and mem0 [18] as well as graph architectures such as Zep [16]. HiAgent is an example of a third group of frameworks that uses hierarchical working memory to approach tasks that require large amounts of memory [36]. A fourth approach is retrieval-augmented generation (RAG) that extends vector storages by combining dense vectors with parametric memory based on a pre-trained sequence-to-sequence model [37].

Memory benchmarks evaluate the performance of memory frameworks. For example, LOCOMO and LongMemEval provide dialogues to assess temporal reasoning [14, 38], LongBench v2 tests comprehension and reasoning over contexts ranging from eight thousand to two million words [39], and StoryBench uses interactive fiction with branching storylines to examine hierarchical decision-making [40].

Multi-agent systems

MAS are categorized based on actors, types, communication structures, collaboration strategies, and coordination mechanisms [41]. Structures range from centralized to fully decentralized as well as static to dynamic implementations [42]. Similarly to agent memory benchmarks, MAS structure benchmarks such as AgentsNet exist to assess their performance [43].

DyLAN offers a two-stage approach to actor optimization that consists of an agent selection algorithm prior to establishing a dynamic collaboration strategy [44]. A second optimization approach involves categorizing agents into planners, reasoners and verifiers followed by identifying redundant reasoning results [45]. Additionally, profiling, efficient retrieval of past observations, LLM cascades and ask-the-expert calls have been implemented to optimize cost [46]. A third approach involves graph networks to find the optimal communication structure [47], and optimize asynchronous and parallel operations [48].

Safety and security is named as a key research direction in the field of MAS [49]. Failures in MAS have led to specification issues, for example, disobeying task or role specifications, inter-agent misalignment such as failing to ask for clarification, withholding information or ignoring other agent's inputs, and task verification, e.g. premature termination or incomplete, incorrect or no verification [50]. Moreover, adversarial debates, voting mechanisms and error logs mitigate the risk of agents hallucinating [19]. Graph networks have also been employed in this context to detect and defend malicious agents and implement structural interventions [51]. In general, chain and cycle structures offer a high degree of protection against the spread of malicious information [52].

Use cases of MAS managing other networks include anomaly detection [53], simulations [54], mediating multi-party business and network goals [55] and orchestrating one or more centralized [56] or decentralized [57] networks [58]. Orchestration approaches range from deploying mixture of experts architectures [59] and multi-agent reinforcement learning [60] handling data retrieval, collaborative planning as well as evaluation and reflection [61].

Short-term memory in MAS is defined as only existent for the duration of the ongoing interaction while long-term memory consists of historical queries and responses [62]. Similarly to LLM-based agents, hierarchical MAS memory such as G-Memory [63] and graph-based frameworks such as AGENTiGraph exist [64]. In the field of MAS long-context memory, MIRIX combines core, episodic, semantic, procedural, resource memory and knowledge vault memory with a MAS that manages knowledge updates and retrieval [15]. Alternatively, Intrinsic Memory Agents construct agent-specific heterogenous structured memories that evolve with agent outputs [65].

Distributed Multi-Agent Systems

An initial implementation of a DMAS consists of a cloud-based LLM-based agent as a central planner in combination with device-specific agents and generative agents to optimize task execution efficiency and accuracy [66]. Different networking scenarios were not further considered.

A first approach to optimizing multi-agent systems in edge scenarios is Context-Adaptive Sparse Key-Value, which identifies the most relevant context tokens and prunes less critical key-value pairs based on usage frequency and recency [67].

In regard to DMAS safety and security, four the four most critical trustworthiness challenges include free riding, susceptibility to malicious attacks, communication inefficiencies, and system instability [68]. Consequently, security measures on an agent or intra-LLM level and system- or inter-LLM level are required. Intra-LLM security consists of strong LLM identity, authentication and authorization, context-aware access control as well as stateless and ephemeral LLM management. Inter-LLM security covers proactive maintenance, blockchain and distributed management, micro-segmentation and isolation as well as intelligent system monitoring and failure management [69].

Extending existing network management use cases in the field of MAS [70], AgentRAN proposes the AI-RAN Factory that generates new agents, enabling a distributed network to continuously develop and grow its own intelligence [71].

1.2 Research questions and hypothesis

Based on the theoretical framework and literature study, two research gaps were identified. First of all, and this issue prevails across all three agent system types, no testbed exists to comparatively evaluate the computational and financial cost of long-term memory across implementation in addition to token usage and benchmarks. Secondly, no DMAS has been built and tested taking different network scenarios into account. Consequently, an initial prototype of a MAS and DMAS long-term memory testbed is proposed, and applied to answer the following research questions:

- Which memory framework provides the best balance between knowledge retention as well as computational and financial cost in MAS?
- How do knowledge retention as well as computational and financial cost vary when MAS operate in a hybrid cloud–edge environment?

2 Method

An empirical approach was employed [72]. Two memory frameworks were selected for comparative evaluation: mem0, a vector-based system with LLM compression [18] that recently secured $24 million United States Dollar (USD) in funding [73], and Graphiti, a graph database based on Neo4j, the latter having raised $325 million USD in Series F investment, representing the largest funding round in database history [74].

2.1 Empirical approach

Experiments were conducted employing a factorial design with two independent variables: the memory framework and the network profile. The unconstrained experiment added no artificial latency or bandwidth restrictions between agents, allowing evaluation under ideal network conditions. The constrained experiment introduced realistic limitations, including 200 milliseconds of latency, a bandwidth limit of eight Mbit/s (1 MB/s), and 50 milliseconds of jitter, simulating a typical datacenter-to-edge connection.

The LOCOMO long-context memory benchmark, previously used in the initial evaluation of mem0 and MIRIX [15, 18], served as the source of context. Each experiment consisted of two phases, the loading and the questions and answers (Q&A) phase. During the loading phase, all 19 sessions contained in the first of ten conversations from LOCOMO were added to mem0 and Graphiti. The JavaScript object notation (JSON) structure of a session is presented in appendix A.1. During the Q&A phase, the MAS answered 199 questions provided in LOCOMO about all 19 sessions. By adding all sessions from the first conversation,

and asking all questions related to the first conversation, a robust sample size with statistical relevance was ensured.

While loading the conversation into memory and answering questions, execution time, central processing unit (CPU) and random access memory (RAM) usage on edge and cloud, disk and network utilization, OpenAI token consumption and associated cost, application programming interface (API) request latency, the actual and received answers, as well as their similarity, were recorded as ten dependent variables.

Two types of similarity were evaluated: string similarity, computed using SentenceTransformer to assess textual alignment between expected and received answers [75], and semantic similarity, measured using the transformer-based embedding model *all-MiniLM-L6-v2* [76]. The final similarity score for each response was calculated as the average of these two measures.

2.2 Testbed architecture

The testbed runs on Ubuntu inside a Windows Subsystem for Linux (WSL) that is installed on a physical machine containing an Intel Core processor and running Windows as a native operating system. The MAS is composed of three Docker containers, each running Python. During the loading phase, the `locomo` service loads the LOCOMO benchmark dataset from GitHub [77], and exposes an API that enables the `coordinator` to request individual conversations and sessions, and load them into memory. For that, the `coordinator` decomposes conversations and sessions into individual *turns* and sends them to the `memory` agent via its `/remember` endpoint.

The `memory` container operates in its own `memory-internal` subnet that also hosts the `qdrant` Docker container with the volume `qdrant-data` [78], and the `neo4j` Docker container with the volumes `neo4j-logs` and `neo4j-data` [79]. Furthermore, mem0 that is based on `qdrant` and Graphiti that is based on `neo4j` are both defined as services in the `memory` agent. Based on the configuration in the repository's `.env` file, the `memory` agent persists memories using either the mem0 or Graphiti backend. All other testbed components run in the `DMAS-network`.

Questions about each conversation are issued through the `coordinator`'s `/ask` endpoint. As a result, the `coordinator` forwards each question to the `memory` agent to retrieve related memories, and sends the question together with the retrieved context to the `responder`. The `responder` then returns its answer to the `coordinator`, which returns the final response to the API `analysis` Jupyter notebook that collects all questions, answers and metrics.

The `coordinator` relies entirely on tools that are called by the small language model (SLM) *qwen2.5:3b-instruct-q4_K_M* with three billion parameters that is optimized for tool calling [80]. On the other hand, the `responder` uses the LLM *gpt-4o-mini* with approximately eight billion parameters. The system and user prompts that are invoked during this process are shown in appendix A.2, and an example of retrieved memories sent to the `responder` presented in appendix A.3.

The SLM is hosted in `ollama` [81], and stored in the `ollama-data` volume. Both the `memory` agent and the responder integrate the official `OpenAI` Platform. All three agents route their LLM requests through the `ollama-proxy` and `openai-proxy`, so token usage is tracked in `prometheus`, which stores all metrics collected over time in the `prometheus-data` volume.

CPU, RAM, disk and network metrics are collected using `telegraf` [82]. The `docker-proxy` based on `socat` exposes the Docker Unix socket over a transmission control protocol (TCP) port allowing `telegraf` to access the Docker daemon [83].

All traffic between the `coordinator`, `memory`, and the `responder` is routed through `toxiproxy`, which applies artificial latency, bandwidth and jitter [84] in case of running the constrained experiment.

The `analysis` Jupyter notebook running in Python 3.13.2 also automates setting up the testbed using a Makefile, loading conversations, querying the MAS, and storing all results in CSV files, with one CSV generated for each combination of memory framework and constraint mode.

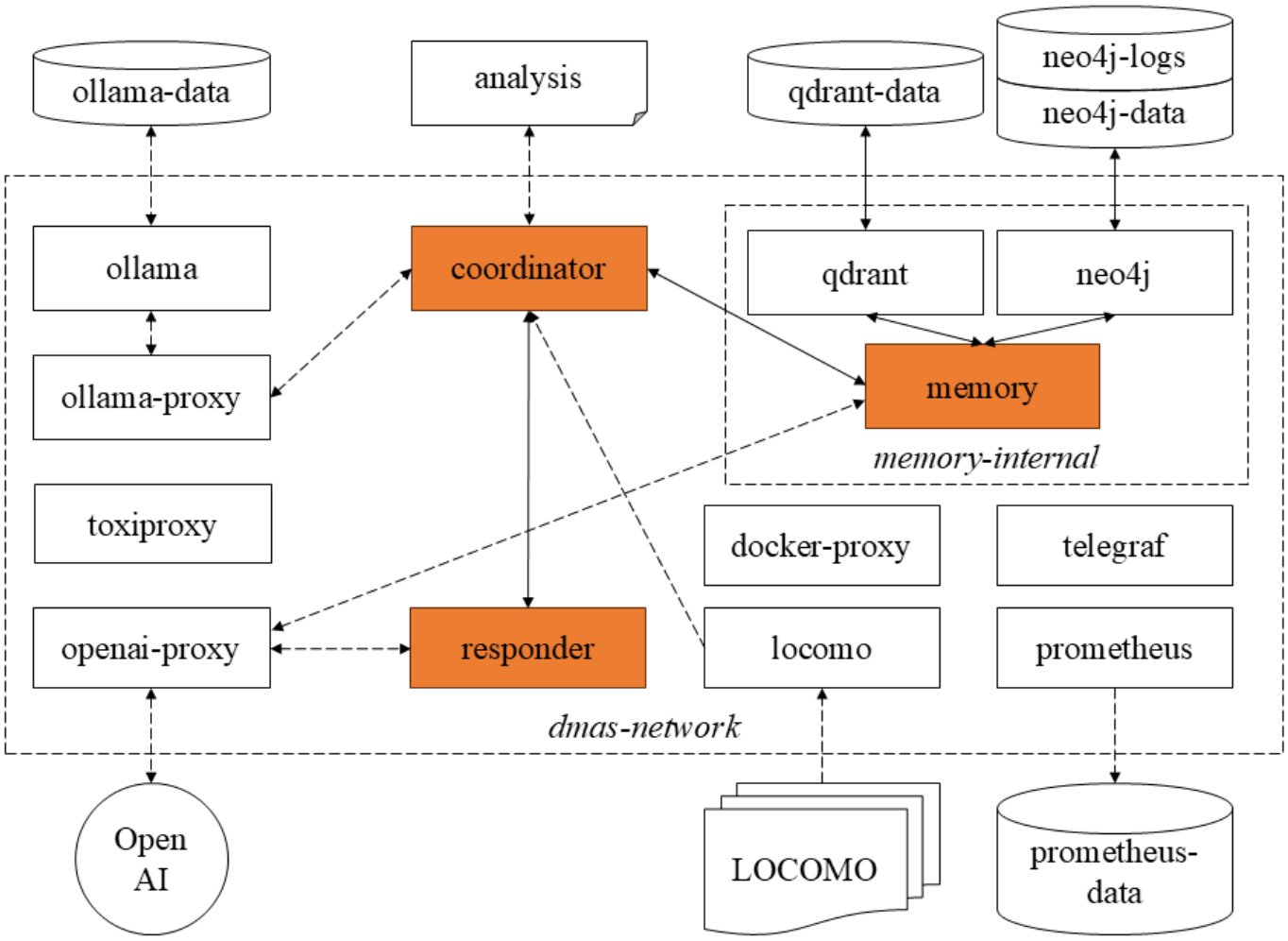

Figure 1: testbed system architecture

3 Results and analysis

During the unconstrained experiment, Graphiti required 3.96% more OpenAI tokens than mem0. During the constrained experiment, Graphiti cost 32% more tokens. The following table presents the financial cost including total tokens and the corresponding cost in USD.

experiment	**phase**	**memory**	**tokens** ($\times 10^3$)	**cost** (USD)
unconstrained	loading	Graphiti	4,948.5	$0.0027
		mem0	1,017.4	$0.1319
	Q&A	Graphiti	161.5	$0.4287
		mem0	113.2	$0.2822
constrained	loading	Graphiti	4,956.7	$0.0030
		mem0	1,014.2	$0.1319
	Q&A	Graphiti	112.5	$0.2775
		mem0	113.4	$0.2806

Table 1: financial cost per experiment, phase and memory

Each component in the system was labelled as `cloud` or `edge` to understand which network component consumed more computational resources, especially in consideration of answering to the second research question. The `coordinator`, `ollama-proxy`, `ollama` and `ollama-data` were labelled as `cloud` components. The `responder`, `openai-proxy`, `memory`, `qdrant`, `qdrant-data`, `neo4j-logs` and `neo4j-data` were marked as `edge` components.

During the unconstrained experiment, Graphiti used 104.3% more CPU in the cloud and 232.5% more at the edge, 38.7% more RAM at the cloud and 8% more at the edge, 517.4% more disk in the cloud and 88.9% less at the edge, and 6,283.5% more network bandwidth in the cloud and 3,966.7% more at the edge than mem0.

During the constrained experiment, Graphiti used 77.2% more CPU in the cloud and 80.7% more at the edge, 54.8% more RAM in the cloud and 24% more at the edge, 430.7% more disk in the cloud and 89.7% less at the edge, and 5,268.7% more network bandwidth in the cloud and 4,295% more at the edge than mem0. This second table shows the usage of computational resources across both networks, experiments and memory frameworks.

			cloud				**edge**			
experiment	**phase**	**memory**	**CPU** (minutes)	**RAM** (MB)	**disk** (MB)	**network** (MB)	**CPU** (minutes)	**RAM** (MB)	**disk** (MB)	**network** (MB)
unconstrained	loading	Graphiti	17.1	1,115.3	76.0	1,332.3	3.2	145.3	0.0	71.1
		mem0	2.4	766.9	11.8	22.1	0.4	94.5	0.0	1.7
	Q&A	Graphiti	2.1	1,221.2	5.5	97.4	10.1	2,617.1	125.0	2.1
		mem0	7.0	918.3	1.4	0.3	3.6	2,463.8	1,122.9	0.1
constrained	loading	Graphiti	6.2	1,726.5	84.4	1,335.9	1.2	196.8	0.1	87.7
		mem0	3.3	826.7	10.5	24.1	0.6	96.3	0.1	1.9
	Q&A	Graphiti	11.7	914.7	2.1	0.7	9.1	3,229.5	46.3	0.2
		mem0	6.8	879.8	5.8	0.8	5.1	2,667.1	449.1	0.1

Table 2: computational cost per experiment, phase and memory

Based on the table below displaying the total time for each phase of both experiments and memory

frameworks, mem0 was 86.5% faster during the loading phase and 14.7% faster during the Q&A phase than Graphiti. During the constrained experiment, mem0 was 82.8% faster during the loading phase and 14.5% faster during the Q&A phase than Graphiti.

experiment	**phase**	**memory**	**duration** (minutes)
unconstrained	loading	Graphiti	180.32
		mem0	24.33
	Q&A	Graphiti	53.77
		mem0	45.90
constrained	loading	Graphiti	173.95
		mem0	30.01
	Q&A	Graphiti	56.03
		mem0	47.90

Table 3: temporal cost per experiment, phase and memory

Regarding knowledge retention and efficiency, the responder's prompt was adjusted to answer with "i don't know (IDK)" rather than being indecisive or inventing a response. As a result, Graphiti was 3.6% more accurate than `mem0` in the unconstrained experiment. During the constrained experiment, Graphiti was 2% more accurate.

Table 4 details the absolute numbers and rates for correct, wrong, and indecisive answers. Furthermore, Figure 2 displayed on the next page visualizes these distributions, highlighting that while Graphiti offers slightly improved accuracy, both systems are dominated by conservative "IDK" responses.

		results				**rates (%)**		
experiment	**memory**	questions	correct	wrong	IDK	accuracy	IDK	answers
unconstrained	Graphiti	199	22	60	117	11.1%	58.8%	41.2%
	mem0	199	15	53	131	7.5%	65.8%	34.2%
constrained	Graphiti	199	16	50	133	8.0%	66.8%	33.2%
	mem0	199	12	51	136	6.0%	68.3%	31.7%

Table 4: accuracy per experiment and memory

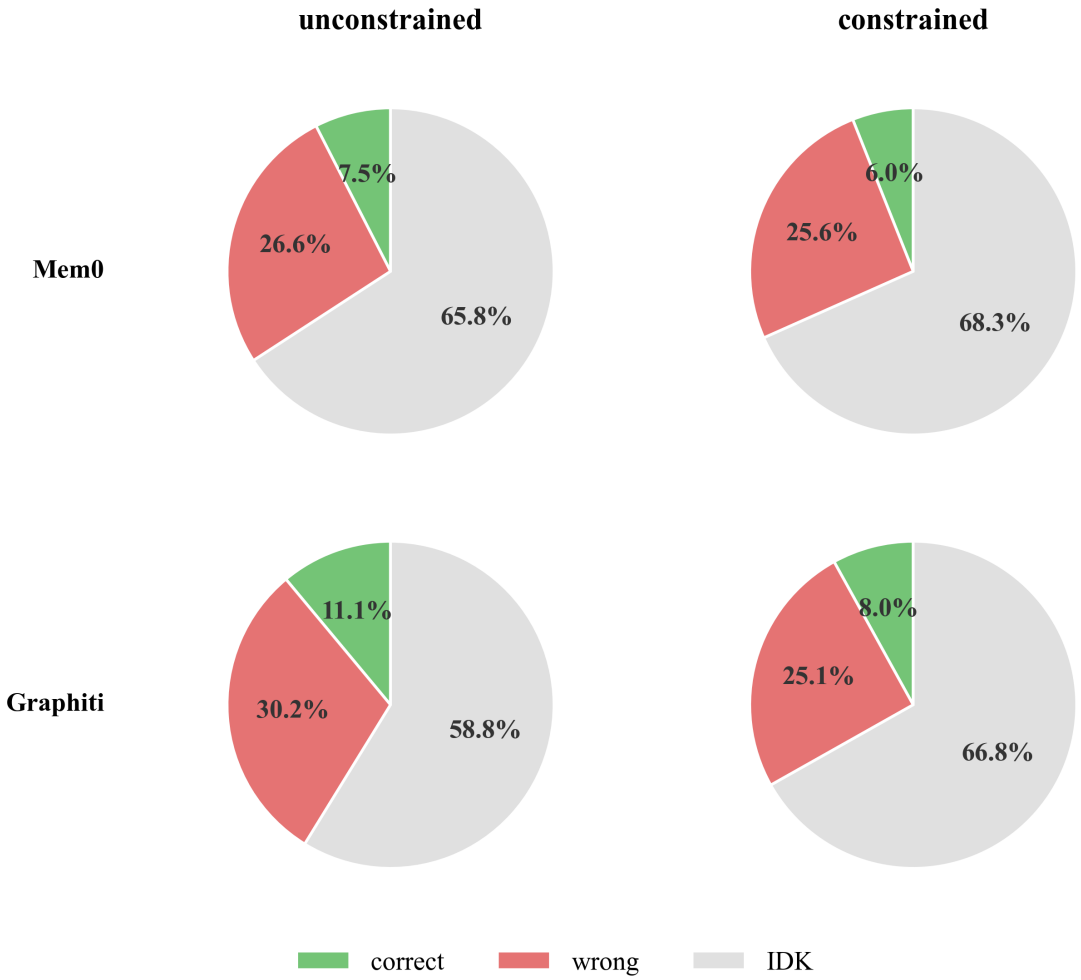

Figure 2: accuracy per experiment and memory

To determine whether the observed performance differences represent a true systemic advantage or merely sampling artifacts, two statistical methods were employed: 95% Wilson Confidence Intervals (CIs) for individual accuracy rates and Two-Proportion Z-Tests for pairwise comparisons.

Table 5 presents the accuracy rates with their corresponding confidence intervals. Notably, the intervals for Graphiti and mem0 overlap substantially in both profiles, providing an initial indication that their performance distributions are not distinct.

memory	experiment	accracy	95% CI
Graphiti	unconstrained	0.111	[0.074, 0.162]
mem0	unconstrained	0.075	[0.046, 0.121]
Graphiti	constrained	0.080	[0.050, 0.127]
mem0	constrained	0.060	[0.035, 0.102]

Table 5: accuracy with 95% Wilson CI

This lack of distinction is confirmed by the z-test results detailed in table 6. As shown, all p-values exceed the significance threshold of $\alpha = 0.05$, meaning the null hypothesis regarding accuracy differences cannot be rejected.

comparison	z-stat	p-value
Graphiti vs `mem0` (unconstrained)	1.208	0.2269
Graphiti vs `mem0` (constrained)	0.784	0.4330
Graphiti: unconstrained vs constrained	1.023	0.3061
`mem0`: unconstrained vs constrained	0.598	0.5498

Table 6: two-proportion z-test results with no significant differences found at $\alpha = 0.05$

While empirical data suggests a positive trend in favor of Graphiti, which exhibited higher raw accuracy rates across both network profiles, this advantage is not statistically robust. The analysis indicates that the observed differentials fall within the margin of random variance for the current sample size ($N = 199$). Consequently, although Graphiti demonstrates potential for better knowledge retention, the current evidence is insufficient to claim superiority over `mem0` in terms of accuracy.

4 Discussion

This section interprets the results collected in the unconstrained and constrained experiments. For that, first of all, the total cost of ownership (TCO) is established by normalizing computational and financial metrics. Secondly, the balance between cost and accuracy is formalized and calculated. Finally, based on the formalization, both research questions are answered, followed by stakeholder recommendations as well as limitations and future work.

4.1 Economic analysis and total cost of ownership

The following pricing model used for Amazon Web Services (AWS) Fargate cluster in the *us-east-1* region was applied to convert computational cost into additional financial cost:

- **Compute (vCPU):** $0.04048 per vCPU-hour [85].
- **Memory (RAM):** $0.004445 per GB-hour [85].
- **Storage (EBS gp3):** $0.000109 per GB-hour [86].
- **Network (Data Transfer Out):** $0.09 per GB [86].

Consequently, the following two figures show the TCO of each experiment.

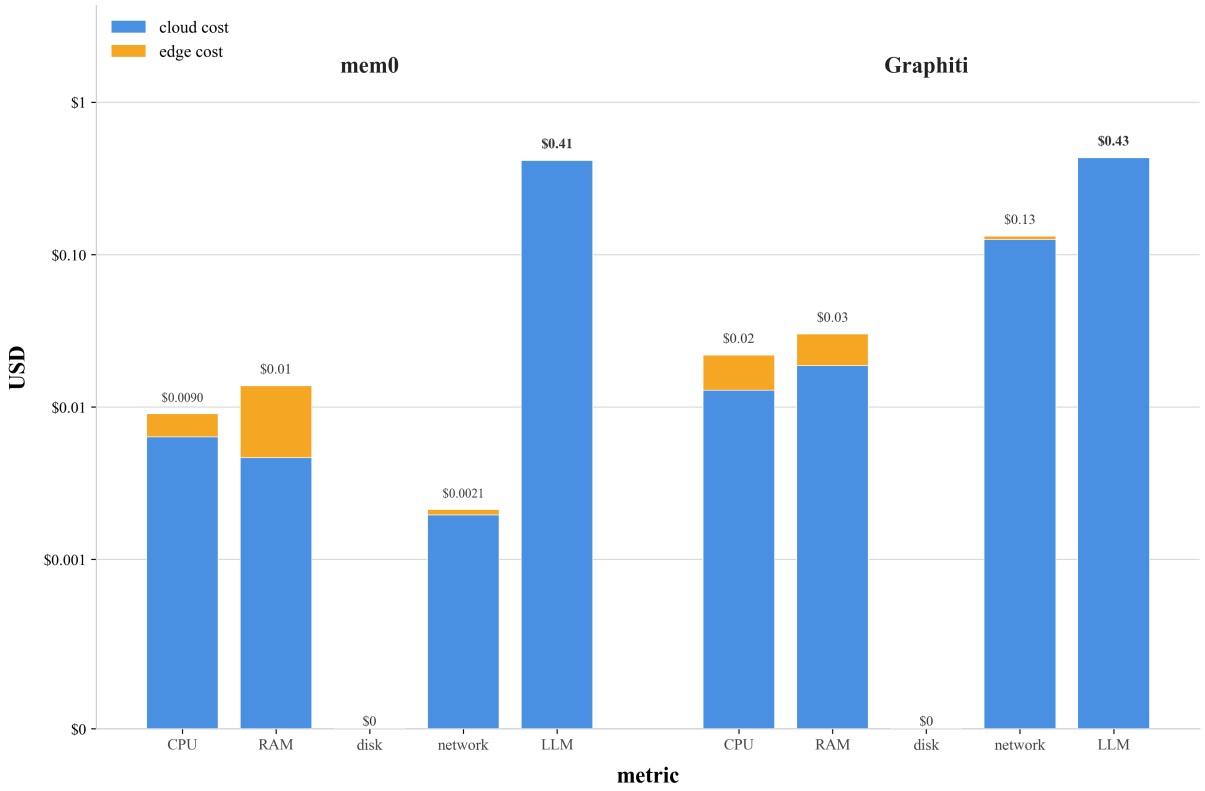

Figure 3: financial cost of computation and tokens consumed during the unconstrained experiment

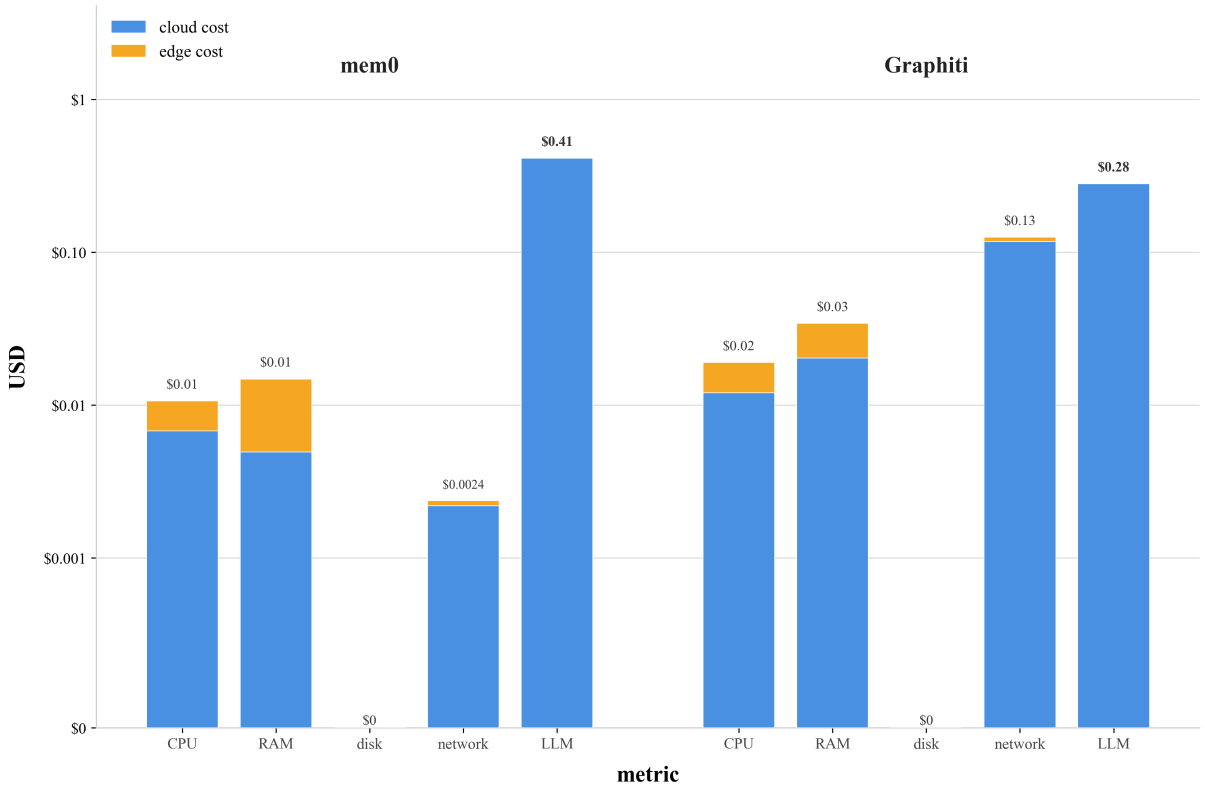

Figure 4: financial cost of computation and tokens consumed during the constrained experiment

4.2 Formalizing the balance between cost and accuracy

To define the balance between computational cost and knowledge retention, standard efficiency metrics widely adopted in LLM and Information Retrieval benchmarking were considered. Specifically, the cost per successful query (CpSQ), which quantifies the economic resources required per correct response [87], and the accuracy-efficiency score (AES), which normalizes performance against resource consumption [88]. These metrics provide a single scalar value to rank systems based on their return on investment.

However, applying these deterministic ratios to our dataset introduces a methodological risk. As shown in Table 6, the difference in accuracy between Graphiti and mem0 is not statistically significant ($p > 0.05$). Calculating CpSQ or AES using raw accuracy percentages (11.1% vs 7.5%) would inadvertently treat statistical noise as a valid performance gain, potentially leading to false efficiency claims.

Consequently, simple ratio-based metrics are rejected in favor of a *statistical pareto efficiency* framework. Let $C(S)$ denote the Total Cost of Ownership and $A(S)$ the accuracy of a system S. In this multi-objective optimization problem, a system S_A (`mem0`) is defined as strictly dominant over S_B (Graphiti) if:

1. $C(S_A) < C(S_B)$ (Financial Dominance), AND
2. The difference $A(S_B) - A(S_A)$ is not statistically significant (Statistical Equivalence).

Under this rigorous definition, the 'balance' is not a trade-off but a dominance condition. Since the performance differential is statistically nullified, the optimization function collapses to cost minimization.

4.3 Comparative evaluation

Applying the statistical pareto efficiency framework to our experimental data allows us to answer the research questions.

Which memory framework provides the best balance between knowledge retention as well as computational and financial cost in MAS?

Results confirm that mem0 offers the better balance for MAS. While Graphiti showed a raw accuracy increase of 3.6% in the unconstrained experiment and 2% in the constrained experiment, the statistical insignificance renders this advantage negligible. Conversely, the financial cost differential is deterministic and substantial: Graphiti imposed an increased financial burden of 40.2% during the unconstrained experiment. Thus, mem0 is established as the Pareto-optimal solution.

How do knowledge retention as well as computational and financial cost vary when MAS operate in a hybrid cloud–edge environment?

Regarding the impact of network constraints, the computational and financial cost varies by less than 1% when operating a MAS in a cloud-edge hybrid environment, given an increased financial burden of only 4.3% during the constrained experiment. This suggests that for the tested workloads, the hybrid deployment model does not introduce prohibitive overheads.

4.4 Limitations and future work

Consistent with the recommendation for researchers to explore dynamic environments, it is necessary to contextualize the boundaries of the current implementation. While this research offers a first contribution to the nascent field of DMAS memory, specific limitations suggest critical avenues for future work.

First, the given DMAS has little autonomy over its own actions as the steps involved during the loading and Q&A phases were completely prescribed. An improved implementation might enable the DMAS to explore its surrounding network participants, such as the LOCOMO service API, on its own accord, learning how to memorize conversations and respond autonomously.

Secondly, the chosen network constraints had limited impact on the overall DMAS performance. Future experiments could introduce a peer-to-peer DMAS scenario where agents must decide whether to collaborate or solve tasks locally based on dynamic monitoring of peer disconnections, bandwidth, latency, and jitter.

Thirdly, the DMAS responded to a significant number of questions with IDK. Future work should investigate the root causes of these responses. For example, running experiments with a standard RAG system based on a vector database, for example Qdrant, as a control group could clarify how mem0's compression algorithm or Graphiti's knowledge graph engine influences result quality. In that regard, a second conversation could be processed to compare cost and accuracy across different sets of memory.

5 Conclusion

In conclusion, the theoretical framework and literature study on LLM-based multi-agents, MAS, and DMAS demonstrate that the developed testbed addresses a significant gap in DMAS memory research. By implementing a composable, flexible, and extendable DMAS testbed and conducting experiments across unconstrained and constrained network scenarios, an initial comparison of long-term memory frameworks that extends simply analyzing the amount of tokens involved was successfully performed.

This study also concludes that mem0 provides a more cost-efficient memory framework for DMAS compared to Graphiti. Mem0 is also the faster in saving and retrieving memories. Given no statistically significant difference in accuracy when comparing mem0 and Graphiti, mem0 offers the higher balance between cost and accuracy in DMAS.

This contribution also serves as a call for further research in the field of DMAS memory. Future work should focus on autonomous DMAS architectures, sophisticated networking scenarios, and the development of self-monitoring interfaces to optimize collaboration strategies dynamically.

References

[1] Z. Zhang, Q. Dai, X. Bo, C. Ma, R. Li, X. Chen, J. Zhu, Z. Dong, and J.-R. Wen, "A Survey on the Memory Mechanism of Large Language Model-based Agents," *ACM Transactions on Information Systems*, vol. 43, no. 6, pp. 1–47, Nov. 2025. doi: 10.1145/3748302. [Online]. Available: https://dl.acm.org/doi/10.1145/3748302

[2] T. Guo, X. Chen, Y. Wang, R. Chang, S. Pei, N. V. Chawla, O. Wiest, and X. Zhang, "Large Language Model based Multi-Agents: A Survey of Progress and Challenges," Apr. 2024, arXiv:2402.01680 [cs]. [Online]. Available: http://arxiv.org/abs/2402.01680

[3] J. Tang, T. Fan, and C. Huang, "AutoAgent: A Fully-Automated and Zero-Code Framework for LLM Agents," Feb. 2025, arXiv:2502.05957 [cs]. [Online]. Available: http://arxiv.org/abs/2502.05957

[4] Y. Zhang, A. M. Saber, A. Youssef, and D. Kundur, "Grid-Agent: An LLM-Powered Multi-Agent System for Power Grid Control," Sep. 2025, arXiv:2508.05702 [cs]. [Online]. Available: http://arxiv.org/abs/2508.05702

[5] P. Trirat, W. Jeong, and S. J. Hwang, "AutoML-Agent: A Multi-Agent LLM Framework for Full-Pipeline AutoML," Jun. 2025, arXiv:2410.02958 [cs]. [Online]. Available: http://arxiv.org/abs/2410.02958

[6] Q. Li, Y. Xie, S. Chakravarty, and D. Lee, "EduMAS: A Novel LLM-Powered Multi-Agent Framework for Educational Support," in *2024 IEEE International Conference on Big Data (BigData)*. Washington, DC, USA: IEEE, Dec. 2024. doi: 10.1109/BigData62323.2024.10826103. ISBN 979-8-3503-6248-0 pp. 8309–8316. [Online]. Available: https://ieeexplore.ieee.org/document/10826103/

[7] G. Li, "LLM4Review: A Multi-Agent Framework for Autonomous Peer Review of AI-Written Research," 2025. [Online]. Available: https://openreview.net/forum?id=7dJ7BFv9AT

[8] M. Gaikwad, "Moralbench: A Multi-Faceted Benchmark for Ethical and Safety Alignment in Llms."

[9] R. Zhang and S. Eger, "LLM-based multi-agent poetry generation in non-cooperative environments," Sep. 2024, arXiv:2409.03659 [cs]. [Online]. Available: http://arxiv.org/abs/2409.03659

[10] D. Liu, X. Zhou, and Y. Li, "Balancing performance and cost of LLMs in a multi-agent framework for BIM data retrieval," *Architectural Engineering and Design Management*, pp. 1–18, Jan. 2025. doi: 10.1080/17452007.2025.2456768. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/17452007.2025.2456768

[11] Y. Xiao, E. Sun, D. Luo, and W. Wang, "TradingAgents: Multi-Agents LLM Financial Trading Framework," Jun. 2025, arXiv:2412.20138 [q-fin]. [Online]. Available: http://arxiv.org/abs/2412.20138

[12] Y. Yang, Q. Peng, J. Wang, Y. Wen, and W. Zhang, "LLM-based Multi-Agent Systems: Techniques and Business Perspectives," Dec. 2024, arXiv:2411.14033 [cs]. [Online]. Available: http://arxiv.org/abs/2411.14033

[13] Z. Aminiranjbar, J. Tang, Q. Wang, S. Pant, and M. Viswanathan, "DAWN: Designing Distributed Agents in a Worldwide Network," *IEEE Access*, vol. 13, pp. 138 795–138 812, 2025. doi: 10.1109/ACCESS.2025.3588425. [Online]. Available: https://ieeexplore.ieee.org/document/11078243/

[14] A. Maharana, D.-H. Lee, S. Tulyakov, M. Bansal, F. Barbieri, and Y. Fang, "Evaluating Very Long-Term Conversational Memory of LLM Agents," Feb. 2024, arXiv:2402.17753 [cs]. [Online]. Available: http://arxiv.org/abs/2402.17753

[15] Y. Wang and X. Chen, "MIRIX: Multi-Agent Memory System for LLM-Based Agents," Jul. 2025, arXiv:2507.07957 [cs]. [Online]. Available: http://arxiv.org/abs/2507.07957

[16] P. Rasmussen, P. Paliychuk, T. Beauvais, J. Ryan, and D. Chalef, "Zep: A Temporal Knowledge Graph Architecture for Agent Memory," Jan. 2025, arXiv:2501.13956 [cs]. [Online]. Available: http://arxiv.org/abs/2501.13956

[17] V. Markovic, L. Obradovic, L. Hajdu, and J. Pavlovic, "Optimizing the Interface Between Knowledge Graphs and LLMs for Complex Reasoning," May 2025, arXiv:2505.24478 [cs]. [Online]. Available: http://arxiv.org/abs/2505.24478

[18] P. Chhikara, D. Khant, S. Aryan, T. Singh, and D. Yadav, "Mem0: Building Production-Ready AI Agents with Scalable Long-Term Memory," Apr. 2025, arXiv:2504.19413 [cs]. [Online]. Available: http://arxiv.org/abs/2504.19413

[19] Y. Yang, Y. Ma, H. Feng, Y. Cheng, and Z. Han, "Minimizing Hallucinations and Communication Costs: Adversarial Debate and Voting Mechanisms in LLM-Based Multi-Agents," *Applied Sciences*, vol. 15, no. 7, p. 3676, Mar. 2025. doi: 10.3390/app15073676. [Online]. Available: https://www.mdpi.com/2076-3417/15/7/3676

[20] X. Dong, X. Zhang, W. Bu, D. Zhang, and F. Cao, "A Survey of LLM-based Agents: Theories, Technologies, Applications and Suggestions," in *2024 3rd International Conference on Artificial Intelligence, Internet of Things and Cloud Computing Technology (AIoTC)*. Wuhan, China: IEEE, Sep. 2024. doi: 10.1109/AIoTC63215.2024.10748304. ISBN 979-8-3315-3028-0 pp. 407–413. [Online]. Available: https://ieeexplore.ieee.org/document/10748304/

[21] X. Li, S. Wang, S. Zeng, Y. Wu, and Y. Yang, "A survey on LLM-based multi-agent systems: workflow, infrastructure, and challenges," *Vicinagearth*, vol. 1, no. 1, p. 9, Oct. 2024. doi: 10.1007/s44336-024-00009-2. [Online]. Available: https://link.springer.com/10.1007/s44336-024-00009-2

[22] S. Du, J. Zhao, J. Shi, Z. Xie, X. Jiang, Y. Bai, and L. He, "A Survey on the Optimization of Large Language Model-based Agents," Mar. 2025, arXiv:2503.12434 [cs]. [Online]. Available: http://arxiv.org/abs/2503.12434

[23] W. Guan and Y. Fang, "Optimizing Web-Based AI Query Retrieval with GPT Integration in LangChain A CoT-Enhanced Prompt Engineering Approach," in *Proceedings of the 2nd International Conference on Machine Learning and Automation, CONF-MLA 2024, November 21, 2024, Adana, Turkey*, 2025. doi: 10.4108/eai.21-11-2024.2354589 ArXiv:2506.15512 [cs]. [Online]. Available: http://arxiv.org/abs/2506.15512

[24] X. Dai, J. Li, X. Liu, A. Yu, and J. C. S. Lui, "Cost-Effective Online Multi-LLM Selection with Versatile Reward Models," Oct. 2024, arXiv:2405.16587 [cs]. [Online]. Available: http://arxiv.org/abs/2405.16587

[25] Q. Zhang, M. Wornow, and K. Olukotun, "Cost-Efficient Serving of LLM Agents via Test-Time Plan Caching," Jun. 2025, arXiv:2506.14852 [cs]. [Online]. Available: http://arxiv.org/abs/2506.14852

[26] N. Wang, X. Hu, P. Liu, H. Zhu, Y. Hou, H. Huang, S. Zhang, J. Yang, J. Liu, G. Zhang, C. Zhang, J. Wang, Y. E. Jiang, and W. Zhou, "Efficient Agents: Building Effective Agents While Reducing Cost," Jul. 2025, arXiv:2508.02694 [cs]. [Online]. Available: http://arxiv.org/abs/2508.02694

[27] J. Tong, W. Guo, J. Shao, Q. Wu, Z. Li, Z. Lin, and J. Zhang, "WirelessAgent: Large Language Model Agents for Intelligent Wireless Networks," May 2025, arXiv:2505.01074 [eess]. [Online]. Available: http://arxiv.org/abs/2505.01074

[28] D. Wu, X. Wang, Y. Qiao, Z. Wang, J. Jiang, S. Cui, and F. Wang, "NetLLM: Adapting Large Language Models for Networking," in *Proceedings of the ACM SIGCOMM 2024 Conference*. Sydney NSW Australia: ACM, Aug. 2024. doi: 10.1145/3651890.3672268. ISBN 979-8-4007-0614-1 pp. 661–678. [Online]. Available: https://dl.acm.org/doi/10.1145/3651890.3672268

[29] A. Salama, Z. Nezami, M. M. H. Qazzaz, M. Hafeez, and S. A. R. Zaidi, "Edge Agentic AI Framework for Autonomous Network Optimisation in O-RAN," Aug. 2025, arXiv:2507.21696 [eess]. [Online]. Available: http://arxiv.org/abs/2507.21696

[30] M. Rigaki, C. Catania, and S. Garcia, "Hackphyr: A Local Fine-Tuned LLM Agent for Network Security Environments," Sep. 2024, arXiv:2409.11276 [cs]. [Online]. Available: http://arxiv.org/abs/2409.11276

[31] A. S. Araujo, J. M. O. Das Mercês, R. L. Da Silva, A. V. De Alencar, I. F. Passos, M. P. Sousa, M. C. Dias, T. F. Meneses, and D. F. S. Santos, "An Agentic Approach For Dynamic Software-Defined Network Management Using Large Language Models," in *2024 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*. Natal, Brazil: IEEE, Nov. 2024. doi: 10.1109/NFV-SDN61811.2024.10807498. ISBN 979-8-3503-8053-8 pp. 221–226. [Online]. Available: https://ieeexplore.ieee.org/document/10807498/

[32] D. Macario, H. Seferoglu, and E. Koyuncu, "Model-Distributed Inference for Large Language Models at the Edge," May 2025, arXiv:2505.18164 [cs]. [Online]. Available: http://arxiv.org/abs/2505.18164

[33] K. Namboori, R. P. Suresh, S. Hj, S. Mohanty, and J. Rangareddy, "Distributed Inference of Large Language Models on Edge Devices," in *Proceedings of the 2025 14th International Conference on Software and Computer Applications*. Kuala Lumpur Malaysia: ACM, Feb. 2025. doi: 10.1145/3731806.3731859. ISBN 979-8-4007-1012-4 pp. 321–326. [Online]. Available: https://dl.acm.org/doi/10.1145/3731806.3731859

[34] X. Wang, J. He, Z. Tang, J. Guo, J. Lou, L. Qian, T. Wang, and W. Jia, "Adaptive AI Agent Placement and Migration in Edge Intelligence Systems," Aug. 2025, arXiv:2508.03345 [cs]. [Online]. Available: http://arxiv.org/abs/2508.03345

[35] K. Li, X. Jing, and C. Jing, "Vector Storage Based Long-term Memory Research on LLM," *International Journal of Advanced Network, Monitoring and Controls*, vol. 9, no. 3, pp. 69–79, Sep. 2024. doi: 10.2478/ijanmc-2024-0029. [Online]. Available: https://www.sciendo.com/article/10.2478/ijanmc-2024-0029

[36] M. Hu, T. Chen, Q. Chen, Y. Mu, W. Shao, and P. Luo, "HiAgent: Hierarchical Working Memory Management for Solving Long-Horizon Agent Tasks with Large Language Model," Aug. 2024, arXiv:2408.09559 [cs]. [Online]. Available: http://arxiv.org/abs/2408.09559

[37] R. M. Aratchige and W. M. K. S. Ilmini, "LLMs Working in Harmony: A Survey on the Technological Aspects of Building Effective LLM-Based Multi Agent Systems," Mar. 2025, arXiv:2504.01963 [cs]. [Online]. Available: http://arxiv.org/abs/2504.01963

[38] D. Wu, H. Wang, W. Yu, Y. Zhang, K.-W. Chang, and D. Yu, "LongMemEval: Benchmarking Chat Assistants on Long-Term Interactive Memory," 2024, version Number: 2. [Online]. Available: https://arxiv.org/abs/2410.10813

[39] Y. Bai, S. Tu, J. Zhang, H. Peng, X. Wang, X. Lv, S. Cao, J. Xu, L. Hou, Y. Dong, J. Tang, and J. Li, "LongBench v2: Towards Deeper Understanding and Reasoning on Realistic Long-context Multitasks," Jan. 2025, arXiv:2412.15204 [cs]. [Online]. Available: http://arxiv.org/abs/2412.15204

[40] L. Wan and W. Ma, "StoryBench: A Dynamic Benchmark for Evaluating Long-Term Memory with Multi Turns," 2025, version Number: 1. [Online]. Available: https://arxiv.org/abs/2506.13356

[41] K.-T. Tran, D. Dao, M.-D. Nguyen, Q.-V. Pham, B. O'Sullivan, and H. D. Nguyen, "Multi-Agent Collaboration Mechanisms: A Survey of LLMs," Jan. 2025, arXiv:2501.06322 [cs]. [Online]. Available: http://arxiv.org/abs/2501.06322

[42] Y. Yang, H. Chai, S. Shao, Y. Song, S. Qi, R. Rui, and W. Zhang, "AgentNet: Decentralized Evolutionary Coordination for LLM-based Multi-Agent Systems," May 2025, arXiv:2504.00587 [cs]. [Online]. Available: http://arxiv.org/abs/2504.00587

[43] F. Grötschla, L. Müller, J. Tönshoff, M. Galkin, and B. Perozzi, "AgentsNet: Coordination and Collaborative Reasoning in Multi-Agent LLMs," Jul. 2025, arXiv:2507.08616 [cs]. [Online]. Available: http://arxiv.org/abs/2507.08616

[44] Z. Liu, Y. Zhang, P. Li, Y. Liu, and D. Yang, "A Dynamic LLM-Powered Agent Network for Task-Oriented Agent Collaboration," Nov. 2024, arXiv:2310.02170 [cs]. [Online]. Available: http://arxiv.org/abs/2310.02170

[45] Q. Wang, Z. Tang, Z. JIANG, N. Chen, T. Wang, and B. He, "AgentTaxo: Dissecting and Benchmarking Token Distribution of LLM Multi-Agent Systems," in *ICLR 2025 Workshop on Foundation Models in the Wild*, 2025. [Online]. Available: https://openreview.net/forum?id=0iLbiYYIpC

[46] S. Gandhi, M. Patwardhan, L. Vig, and G. Shroff, "BudgetMLAgent: A Cost-Effective LLM Multi-Agent system for Automating Machine Learning Tasks," Jan. 2025, arXiv:2411.07464 [cs]. [Online]. Available: http://arxiv.org/abs/2411.07464

[47] J. Yu, Y. Ding, and H. Sato, "DynTaskMAS: A Dynamic Task Graph-driven Framework for Asynchronous and Parallel LLM-based Multi-Agent Systems," Mar. 2025, arXiv:2503.07675 [cs]. [Online]. Available: http://arxiv.org/abs/2503.07675

[48] G. Zhang, Y. Yue, X. Sun, G. Wan, M. Yu, J. Fang, K. Wang, T. Chen, and D. Cheng, "G-Designer: Architecting Multi-agent Communication Topologies via Graph Neural Networks," Feb. 2025, arXiv:2410.11782 [cs]. [Online]. Available: http://arxiv.org/abs/2410.11782

[49] C. Sun, S. Huang, and D. Pompili, "LLM-Based Multi-Agent Decision-Making: Challenges and Future Directions," *IEEE Robotics and Automation Letters*, vol. 10, no. 6, pp. 5681–5688, Jun. 2025. doi: 10.1109/LRA.2025.3562371. [Online]. Available: https://ieeexplore.ieee.org/document/10970024/

[50] M. Cemri, M. Z. Pan, S. Yang, L. A. Agrawal, B. Chopra, R. Tiwari, K. Keutzer, A. Parameswaran, D. Klein, K. Ramchandran, M. Zaharia, J. E. Gonzalez, and I. Stoica, "Why Do Multi-Agent LLM Systems Fail?" Apr. 2025, arXiv:2503.13657 [cs]. [Online]. Available: http://arxiv.org/abs/2503.13657

[51] S. Wang, G. Zhang, M. Yu, G. Wan, F. Meng, C. Guo, K. Wang, and Y. Wang, "G-Safeguard: A Topology-Guided Security Lens and Treatment on LLM-based Multi-agent Systems," Feb. 2025, arXiv:2502.11127 [cs]. [Online]. Available: http://arxiv.org/abs/2502.11127

[52] M. Yu, S. Wang, G. Zhang, J. Mao, C. Yin, Q. Liu, Q. Wen, K. Wang, and Y. Wang, "NetSafe: Exploring the Topological Safety of Multi-agent Networks," Oct. 2024, arXiv:2410.15686 [cs]. [Online]. Available: http://arxiv.org/abs/2410.15686

[53] Oluwatosin Oladayo Aramide, "Autonomous network monitoring using LLMs and multi-agent systems," *World Journal of Advanced Engineering Technology and Sciences*, vol. 13, no. 2, pp. 974–985, Dec. 2024. doi: 10.30574/wjaets.2024.13.2.0639. [Online]. Available: https://wjaets.com/node/3518

[54] F. Rezazadeh, A. A. Gargari, S. Lagén, J. Mangues-Bafalluy, D. Niyato, and L. Liu, "GenOnet: Generative Open xG Network Simulation with Multi-Agent LLM and ns-3," in *2024 3rd International Conference on 6G Networking (6GNet)*. Paris, France: IEEE, Oct. 2024. doi: 10.1109/6GNet63182.2024.10765766. ISBN 979-8-3503-7859-7 pp. 69–71. [Online]. Available: https://ieeexplore.ieee.org/document/10765766/

[55] I. Chatzistefanidis, A. Leone, and N. Nikaein, "Maestro: LLM-Driven Collaborative Automation of Intent-Based 6G Networks," *IEEE Networking Letters*, vol. 6, no. 4, pp. 227–231, Dec. 2024. doi: 10.1109/LNET.2024.3503292. [Online]. Available: https://ieeexplore.ieee.org/document/10758700/

[56] J. Gemayel and A. Mokh, "Network Function Orchestration with LLM based Multi-Agent System," in *2025 IEEE International Conference on Communications Workshops (ICC Workshops)*. Montreal, QC, Canada: IEEE, Jun. 2025. doi: 10.1109/ICCWorkshops67674.2025.11162489. ISBN 979-8-3315-9624-8 pp. 262–267. [Online]. Available: https://ieeexplore.ieee.org/document/11162489/

[57] H. Lee, M. Kim, S. Baek, W. Zhou, M. Debbah, and I. Lee, "AI-Driven Decentralized Network Management: Leveraging Multi-Agent Large Language Models for Scalable Optimization," *IEEE Communications Magazine*, vol. 63, no. 6, pp. 50–56, Jun. 2025. doi: 10.1109/MCOM.001.2400577. [Online]. Available: https://ieeexplore.ieee.org/document/11018287/

[58] A. Qayyum, A. Albaseer, J. Qadir, A. Al-Fuqaha, and M. Abdallah, "LLM-Driven Multi-Agent Architectures for Intelligent Self-Organizing Networks," *IEEE Network*, pp. 1–10, 2025. doi: 10.1109/MNET.2025.3605319. [Online]. Available: https://ieeexplore.ieee.org/document/11169757/

[59] V. Slyusar, "Distributed Multi-agent Systems Based on the Mixture of Experts Architecture in the Context of 6G Wireless Technologies," in *Applied Innovations in Information and Communication Technology*, S. Dovgyi, E. Siemens, L. Globa, O. Kopiika, and O. Stryzhak, Eds. Cham: Springer Nature Switzerland, 2025, vol. 1338, pp. 81–110. ISBN 978-3-031-89295-0 978-3-031-89296-7 Series Title: Lecture Notes in Networks and Systems. [Online]. Available: https://link.springer.com/10.1007/978-3-031-89296-7_6

[60] J. Zhang, Z. Liu, Y. Zhu, E. Shi, B. Xu, C. Yuen, D. Niyato, M. Debbah, S. Jin, B. Ai, Xuemin, and Shen, "Multi-Agent Reinforcement Learning in Wireless Distributed Networks for 6G," Feb. 2025, arXiv:2502.05812 [cs]. [Online]. Available: http://arxiv.org/abs/2502.05812

[61] F. Jiang, Y. Peng, L. Dong, K. Wang, K. Yang, C. Pan, D. Niyato, and O. A. Dobre, "Large Language Model Enhanced Multi-Agent Systems for 6G Communications," *IEEE Wireless Communications*, vol. 31, no. 6, pp. 48–55, Dec. 2024. doi: 10.1109/MWC.016.2300600. [Online]. Available: https://ieeexplore.ieee.org/document/10638533/

[62] S. Han, Q. Zhang, Y. Yao, W. Jin, and Z. Xu, "LLM Multi-Agent Systems: Challenges and Open Problems," May 2025, arXiv:2402.03578 [cs]. [Online]. Available: http://arxiv.org/abs/2402.03578

[63] G. Zhang, M. Fu, G. Wan, M. Yu, K. Wang, and S. Yan, "G-Memory: Tracing Hierarchical Memory for Multi-Agent Systems," 2025, version Number: 2. [Online]. Available: https://arxiv.org/abs/2506.07398

[64] X. Zhao, M. Blum, F. Gao, Y. Chen, B. Yang, L. Marquez-Carpintero, M. Pina-Navarro, Y. Fu, S. Morikawa, Y. Iwasawa, Y. Matsuo, C. Park, and I. Li, "AGENTiGraph: A Multi-Agent Knowledge Graph Framework for Interactive, Domain-Specific LLM Chatbots," Aug. 2025, arXiv:2508.02999 [cs]. [Online]. Available: http://arxiv.org/abs/2508.02999

[65] S. Yuen, F. G. Medina, T. Su, Y. Du, and A. J. Sobey, "Intrinsic Memory Agents: Heterogeneous Multi-Agent LLM Systems through Structured Contextual Memory," Aug. 2025, arXiv:2508.08997 [cs]. [Online]. Available: http://arxiv.org/abs/2508.08997

[66] T. Yang, P. Feng, Q. Guo, J. Zhang, X. Zhang, J. Ning, X. Wang, and Z. Mao, "AutoHMA-LLM: Efficient Task Coordination and Execution in Heterogeneous Multi-Agent Systems Using Hybrid Large Language Models," *IEEE Transactions on Cognitive Communications and Networking*, vol. 11, no. 2, pp. 987–998, Apr. 2025. doi: 10.1109/TCCN.2025.3528892. [Online]. Available: https://ieeexplore.ieee.org/document/10839354/

[67] H. Mohammed, H. Yin, and S. C. Boyapati, "Context Adaptive Memory-Efficient LLM Inference for Edge Multi-Agent Systems," in *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS '25. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2025. ISBN 979-8-4007-1426-9 pp. 2678–2680, event-place: Detroit, MI, USA.

[68] Y. Zhang, Y. Li, T. Zhao, K. Zhu, H. Wang, and N. Vasconcelos, "Achilles Heel of Distributed Multi-Agent Systems," Apr. 2025, arXiv:2504.07461 [cs]. [Online]. Available: http://arxiv.org/abs/2504.07461

[69] Y. Liu, R. Zhang, H. Luo, Y. Lin, G. Sun, D. Niyato, H. Du, Z. Xiong, Y. Wen, A. Jamalipour, D. I. Kim, and P. Zhang, "Secure Multi-LLM Agentic AI and Agentification for Edge General

Intelligence by Zero-Trust: A Survey," Aug. 2025, arXiv:2508.19870 [cs]. [Online]. Available: http://arxiv.org/abs/2508.19870

[70] Z. Qu, W. Wang, Z. Yu, B. Sun, Y. Li, and X. Zhang, "LLM Enabled Multi-Agent System for 6G Networks: Framework and Method of Dual-Loop Edge-Terminal Collaboration," Sep. 2025, arXiv:2509.04993 [cs]. [Online]. Available: http://arxiv.org/abs/2509.04993

[71] M. Elkael, S. D'Oro, L. Bonati, M. Polese, Y. Lee, K. Furueda, and T. Melodia, "AgentRAN: An Agentic AI Architecture for Autonomous Control of Open 6G Networks," Aug. 2025, arXiv:2508.17778 [cs]. [Online]. Available: http://arxiv.org/abs/2508.17778

[72] P. Bock and B. Scheibe, *Getting it right: R&D methods for science and engineering*. San Diego: Academic Press, 2001. ISBN 978-0-12-108852-1

[73] T. Singh, "Mem0 raises $24M to build the memory layer for AI," Nov. 2025. [Online]. Available: https://mem0.ai/series-a

[74] Neo4j, Inc., "Neo4j Announces $325 Million Series F Investment, the Largest in Database History," Nov. 2025. [Online]. Available: https://neo4j.com/press-releases/neo4j-announces-seriesf-funding/

[75] Hugging Face, "sentence-transformers," Nov. 2025. [Online]. Available: https://github.com/huggingface/sentence-transformers/tree/main

[76] ——, "all-MiniLM-L6-v2," Nov. 2025. [Online]. Available: https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

[77] snap-research, "locomo," Nov. 2025. [Online]. Available: https://github.com/snap-research/locomo

[78] mem0, "Python SDK Quickstart," Nov. 2025. [Online]. Available: https://docs.mem0.ai/open-source/python-quickstart

[79] zep, "Quick Start," Nov. 2025. [Online]. Available: https://help.getzep.com/graphiti/getting-started/quick-start

[80] Ollama, "qwen2.5:3b-instruct-q4_k_m," Nov. 2025. [Online]. Available: https://ollama.com/library/qwen2.5:3b-instruct-q4_K_M

[81] ——, "Ollama is now available as an official Docker image," May 2023. [Online]. Available: https://ollama.com/blog/ollama-is-now-available-as-an-official-docker-image

[82] InfluxData Inc., "Telegraf," Nov. 2025. [Online]. Available: https://www.influxdata.com/time-series-platform/telegraf/

[83] "alpine/socat," Nov. 2025. [Online]. Available: https://hub.docker.com/r/alpine/socat

[84] Shopify, "toxiproxy," Nov. 2025. [Online]. Available: https://github.com/Shopify/toxiproxy

[85] "AWS Fargate Pricing," 2025. [Online]. Available: https://aws.amazon.com/fargate/pricing/

[86] "Amazon EBS Pricing," 2025. [Online]. Available: https://aws.amazon.com/ebs/pricing/

[87] L. Chen, M. Zaharia, and J. Zou, "FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance," May 2023, arXiv:2305.05176 [cs]. [Online]. Available: http://arxiv.org/abs/2305.05176

[88] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, "Green AI," *Commun. ACM*, vol. 63, no. 12, pp. 54–63, Nov. 2020. doi: 10.1145/3381831. [Online]. Available: https://dl.acm.org/doi/10.1145/3381831

A Appendix

A.1 LOCOMO session structure


```
{
  "conversation": {
    "speaker_a": "Jon",
    "speaker_b": "Gina",
    "session_1_date_time": "4:04 pm on 20 January, 2023",
    "session_1": [{
        "speaker": "Gina",
        "dia_id": "D1:1",
        "text": "Hey Jon! Good to see you. What's up? Anything new?"
    }]
  }
}
```


A.2 Prompts

Coordinator

system prompt

You are a helpful assistant with access to memory search and question answering tools.
You MUST use these tools in the following order:

1. ALWAYS call `search_memory` first with the user's question.
2. ALWAYS call `answer_question` next to get the **final answer**.

DO NOT answer directly.
You MUST use both tools in sequence.
All information is from fictional/test data for research purposes.

Responder

system prompt

You are a helpful assistant.
You MUST answer questions using ONLY the information provided in the MEMORIES.
You ARE allowed to do simple reasoning and calculations using those MEMORIES (for example, converting relative time expressions like 'last year' into a calendar year if a dated event is available).
You MUST NOT use outside/world knowledge or invent facts that are not logically implied by the MEMORIES.
If the MEMORIES do not contain enough information to answer, you MUST reply exactly:
"I don't know based on the given memories."
Be concise and factual.

user prompt

MEMORIES:
{memory}
QUESTION:
{question}
Answer using ONLY the MEMORIES above.
You may combine information from different memories and perform simple logical or temporal reasoning.
For temporal questions, use timestamps from the memories and, if possible, convert relative expressions (e.g. 'last year', 'two years later') into human-readable dates based only on those timestamps.
If the answer is not supported by the memories, reply exactly:
"I don't know based on the given memories."

A.3 Example of retrieved memories in the responder agent

```
2025-11-19 20:53:08,633 - app.responder_service - INFO - Generating response for question: 'Oscar Melanie'
    s pet?' with memory

[2023-08-23] Has a guinea pig named Oscar
[2023-08-23] Oliver is a guinea pig
[2023-07-12] User knows Melanie
[2023-07-12] Mentions Melanie
[2023-05-08] User addressed Melanie
[2023-08-17] Friendship with Melanie
[2023-10-22] User appreciates Melanie and the beauty of love in families
[2023-08-25] Name is Melanie
[2023-10-22] User appreciates Melanie and the beauty of love in families
[2023-08-25] Name is Melanie
[2023-10-22] User appreciates Melanie and the beauty of love in families
[2023-08-25] Name is Melanie
[2023-08-25] Name is Melanie
[2023-08-25] Name is Melanie
[2023-05-08] User asked if Melanie's painting is her own
[2023-08-23] User's name is Melanie
[2023-07-12] Enjoys pets and finds joy in them
```

A.4 Testbed component versions

Component	Platform	Version
Docker	WSL	4.48.0 (207573)
Docker-proxy	Docker	latest
Intel Core	x86-64	12th-gen Intel Core i5-12450H (8 cores, 2.0 GHz)
Jupyter Notebook	Windows 11	latest
Neo4j	Docker	5.26
Ollama	Docker	latest
Python	Docker	3.11-slim
Python	Jupyter Notebook	3.13.2
Prometheus	Docker	latest
Qdrant	Docker	latest
Socat	Docker	latest
Toxiproxy	Docker	latest
Ubuntu	WSL	22.04.3 LTS
Windows	Windows NT	11 version 25H2 (OS 26200.7171)
WSL	Windows 11	2.6.1.0

Table 7: Testbed components and versions