AriGraph: Learning Knowledge Graph World Models with Episodic Memory for LLM Agents

Petr Anokhin[1], **Nikita Semenov**[2], **Artyom Sorokin**[1], **Dmitry Evseev**[2], **Andrey Kravchenko**[4], **Mikhail Burtsev**[3] and **Evgeny Burnaev**[2,1]
[1]AIRI, Moscow, Russia
[2]Skoltech, Moscow, Russia
[3]London Institute for Mathematical Sciences, London, UK
[4]University of Oxford, UK
anokhin@airi.net

Abstract

Advancements in the capabilities of Large Language Models (LLMs) have created a promising foundation for developing autonomous agents. With the right tools, these agents could learn to solve tasks in new environments by accumulating and updating their knowledge. Current LLM-based agents process past experiences using a full history of observations, summarization, retrieval augmentation. However, these unstructured memory representations do not facilitate the reasoning and planning essential for complex decision-making. In our study, we introduce AriGraph, a novel method wherein the agent constructs and updates a memory graph that integrates semantic and episodic memories while exploring the environment. We demonstrate that our Ariadne LLM agent, consisting of the proposed memory architecture augmented with planning and decision-making, effectively handles complex tasks within interactive text game environments difficult even for human players. Results show that our approach markedly outperforms other established memory methods and strong RL baselines in a range of problems of varying complexity. Additionally, AriGraph demonstrates competitive performance compared to dedicated knowledge graph-based methods in static multi-hop question-answering.

1 Introduction*

Impressive language generation capabilities of large language models (LLMs) has sparked substantial interest in their application as core components for creating autonomous agents capable of interacting with dynamic environments and executing complex tasks. Over the past year, the research community has explored general architectures and core modules for such LLM agents [Wang *et al.*, 2024; Sumers *et al.*, 2024; Cheng *et al.*, 2024]. A crucial property of a general cognitive agent is its ability to accumulate and use knowledge. A long-term memory allows an agent to store and recall past experiences and knowledge, enabling it to learn from previous encounters and make informed decisions. However, the question of the best way to equip an agent with these capabilities remains open. Despite the constraints inherent in transformer architectures, contemporary methods enable LLMs to manage contexts encompassing millions of tokens [Ding *et al.*, 2024b]. However, this approach proves inefficient for agents required to maintain continuous interaction with their environment. Such agents must hold an entire historical context in memory to perform actions, which is not only costly but also limited in handling complex logic hidden in vast amounts of information. Research into alternative frameworks like Recurrent Memory Transformer [Bulatov *et al.*, 2022; Bulatov *et al.*, 2024] and MAMBA [Gu and Dao, 2023] seeks to provide long-term memory solutions, though these models are still in their infancy.

Currently, the most popular solution for incorporating memory to LLM agents is the Retrieval-Augmented Generation (RAG) approach. RAG in a form of vector retrieval leverages an external database to enhance the model's prompt with relevant information. This technique is commonly used in memory architectures for LLM agents, often to recall specific observations or learned skills. However, it suffers from unstructured nature, greatly reducing the ability to retrieve related information, which may be scattered throughout the agent's memory. These limitations can be overcome by using knowledge graphs as database. This approach has also experienced a resurgence with the advent of LLMs [Pan *et al.*, 2024]. However, for a robust memory architecture, integrating both structured and unstructured data is essential. In cognitive science, this integration parallels the concepts of semantic and episodic memories. Semantic memory encompasses factual knowledge about the world, whereas episodic memory pertains to personal experiences, which often contain richer and more detailed information. Though traditionally considered separate due to their distinct neurological representations, recent studies suggest these memory types are interconnected [Wong Gonzalez, 2018]. Semantic knowledge is built upon the foundation of episodic memory and subsequently provides a structured base for associative memory. This allows for the integration of various memory aspects,

*An extended version of the paper, including appendices, is available at arXiv:2407.04363.

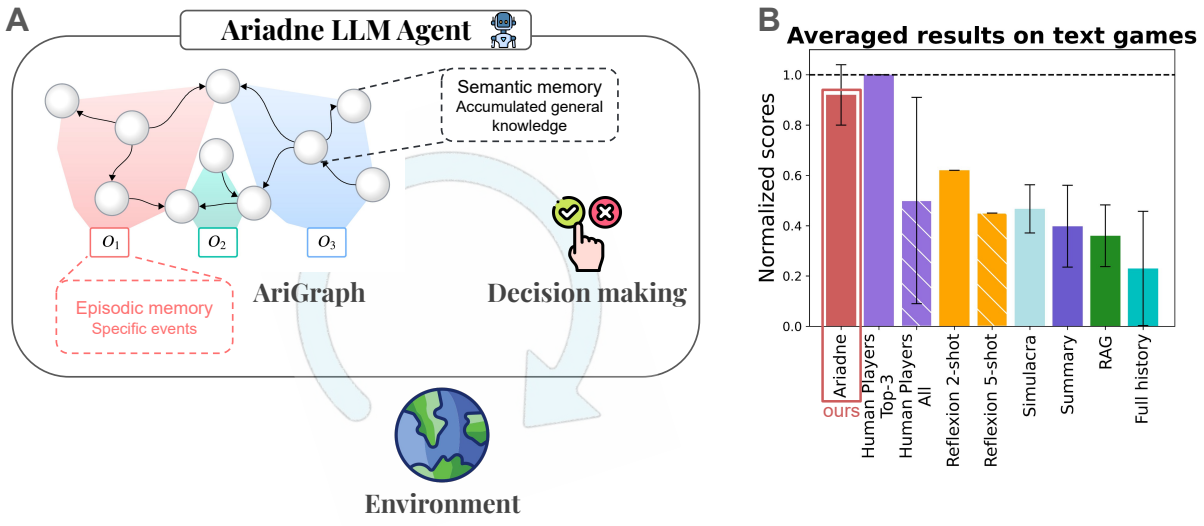

Figure 1: (A) The architecture of our Ariadne agent, equipped with AriGraph memory. AriGraph integrates both semantic knowledge graph and past experiences. Memory in the form of a semantic knowledge graph extended with episodic vertices and edges significantly enhances the performance of LLM-agent in text-based games. (B) The average performance of our agent on text games, compared to various baselines including human players and other LLM memory implementations. The LLM-agents differ only in the memory module, while the decision-making component remains identical across all versions. The results for the agents are displayed for the top three out of five runs. For human players, the results are presented as both the top three and the average across all participants.

including episodic memories themselves.

In our research, we have developed a memory architecture called Ariadne's Graph (AriGraph), that integrates semantic and episodic memories within a memory graph framework. A knowledge graph represents a network of interconnected semantic knowledge, while episodic memories are depicted as episodic edges that can connect multiple relations within the graph. As an agent interacts with environment, it learns joint semantic and episodic world model by updating and extending knowledge graph based memory. This architecture not only serves as a foundational memory framework but also aids in environmental modeling, improving spatial orientation and exploration capabilities. For the general framework of our LLM agent called Ariadne, we employed pipeline of memory retrieval, planing and decision making. For evaluation of proposed methods we set up experiments to study two research questions.

RQ1. Can LLM based agents learn useful structured world model from scratch via interaction with an environment?

RQ2. Does structured knowledge representation improve retrieval of relevant facts from memory and enable effective exploration?

We evaluated our agent in complex interactive tasks in Textworld and NetHack environments [Côté *et al.*, 2018; Küttler *et al.*, 2020]. Experimental results demonstrate that our agent Ariadne can effectively learn through interactions with environment and significantly outperforms other memory approaches for LLMs such as full history, summarization, RAG, Simulacra [Park *et al.*, 2023] and Reflexion [Shinn *et al.*, 2023]. We also show that our method outperforms existing reinforcement learning (RL) baselines. We also evaluated our approach on the classical roguelike game NetHack, where our agent with local observations achieved scores comparable to an agent with ground-truth knowledge. Although AriGraph was originally designed for an agent interacting with the environment, it also demonstrates competitive performance on multi-hop question answering tasks.

2 AriGraph World Model

Memory graph structure. AriGraph world model $G = (V_s, E_s, V_e, E_e)$ consists of *semantic* (V_s, E_s) and *episodic memory* (V_e, E_e) vertices and edges (see Figure 2). At each step t agent receives observation o_t and sends action a_t back to the environment. The environment also returns rewards r_t that are not visible to the LLM agent but are used to evaluate its performance. The agent continuously learns world model G by extracting semantic triplets $(object_1, relation, object_2)$ from textual observations o_t.

- V_s is a set of semantic vertices. Semantic vertices correspond to objects extracted from triplets.
- E_s is a set of semantic edges. Semantic edge is a tuple (v, rel, u), where u, v are semantic vertices and rel is a relationship between them. Semantic edges essentially represent triplets integrated in the *semantic memory*.
- V_e is a set of episodic vertices. Each episodic vertex corresponds to an observation received from the environment at the respective step $v_e^t = o_t$.
- E_e is a set of episodic edges. Each episodic edge $e_e^t = (v_e^t, E_s^t)$ connects all semantic triplets E_s^t extracted from o_t with each other and corresponding episodic ver-

Algorithm 1: Memory Graph Search

Input: set of queries Q, V_s, E_s, V_e, E_e, number of episodic vertices k, semantic search depth d and width w
Result: retrieved episodic vertices V_e^Q, retrieved semantic triplets E_s^Q

$E_s^Q \leftarrow \emptyset$,
foreach q in Q **do**
  $E'_s \leftarrow \texttt{SemanticSearch}(q, V_s, E_s, d, w)$
  $E_s^Q \leftarrow E_s^Q \cup E'_s$
end
$V_e^Q \leftarrow \texttt{EpisodicSearch}(E_s^Q, V_e, E_e, k)$
return E_s^Q, V_e^Q

tex v_e^t. In other words episodic edges represent temporal relationship "happened at the same time".[1]

Constructing AriGraph. Interaction with the environment can provide the agent with an information about the world to create new or update previously acquired knowledge. Given new observation o_t, LLM agent extracts new triplets as semantic vertices V_s^t and edges E_s^t. To find already existing knowledge about the objects mentioned in o_t a set of all semantic edges E_s^{rel} incident to vertices V_s^t is filtered out. Then outdated edges in E_s^{rel} are detected by comparing them with E_s^t and removed from the graph. After clearing outdated knowledge we expand semantic memory with V_s^t and E_s^t. Episodic memory is updated by simply adding new episodic vertex v_e^t containing o_t and new episodic edge that connect all edges in E_s^t with v_e^t. Episodic nodes store agent's past history and episodic edges connects all knowledge received at the same step. See Appendix E for prompts used to extract new triplets and detect outdated knowledge.

Retrieval from AriGraph. For successful decision-making in a partially observable environment, the agent needs to be able to retrieve relevant knowledge. Retrieval from the AriGraph memory consists of two procedures: (1) a semantic search returns the most relevant triplets (semantic edges) and (2) an episodic search that, given extracted triplets, returns the most relevant episodic vertices V_e. The pseudo-code for the search is presented in the Algorithm 1.

Semantic search relies on semantic similarity and semantic graph structure to recall the most relevant triplets. Given a query, the retriever (pre-trained Contriever model [Izacard *et al.*, 2022]) selects the most relevant semantic triplets. Then, the set of vertices incident to the found triplets is used to recursively retrieve new edges from the graph. Depth and breadth of the search can be controlled by respective hyperparameters d and w. For details see Appendix A.

Episodic search starts with the results of the semantic search as an input. Episodic edges link the input triplets with past episodic vertices representing observations. The number of input triplets associated with a particular episodic vertex is used to calculate their relevance:

$$rel(v_e^i) = \frac{n_i}{max(N_i, 1)} \log(max(N_i, 1)), \qquad (1)$$

where n_i is a number of input triplets incident to episodic edge e^i, N_i is a total number triplets (semantic edges) incident to e^i and $\log(max(N_i, 1))$ is a weighting factor to prevent high scores for low information observations. k most relevant episodic vertices (containing respective observations) are returned as a result of the episodic search.

3 Ariadne Cognitive Architecture

To test utility of AriGraph world modelling method we propose an agentic architecture called Ariadne. Ariadne agent interacts with an unknown environment to accomplish a goal set by a user. Throughout this process, at each time step, the agent learns a world model, plans and executes actions. Ariadne has *long-term* memory stored as AriGraph and *working* memory containing information for current planning and decision making.

Given an observation the agent updates world model and retrieves semantic and episodic knowledge from AriGraph to working memory. Working memory is also populated with a final goal description, current observation, history of recent observation and actions. At the planning stage, Ariadne agent uses content of working memory to create new or update existing plan as a series of task-relevant sub-goals, each accompanied by a concise description. The planning module also evaluates the outcomes of actions based on feedback from the environment after each action at step $t-1$, adjusting the plan accordingly.

The revised plan is added to the working memory which is accessed by the decision-making module, tasked with selecting the most suitable action aligned with the current plan's objectives. This module adheres to the ReAct [Yao *et al.*, 2023] framework, requiring the agent to articulate the rationale behind an action before execution. Separation of planning from decision-making enables LLMs to focus on distinct cognitive processes. In text-based environments an agent selects an action from the list of valid actions. Our agent can also use graph specific function for navigation utilizing its memory module. It extends its action space with "go to location" type commands and infers an optimal route to a target location using spatial relations stored in a semantic graph.

4 Experimental Setup

4.1 TextWorld Interactive Environments

We compared Ariadne agent with alternative methods in a series of text based games involving spatial navigation, object collection and tool manipulation. All these games can be considered Partially Observable MDPs (POMDPs). Such games have long been benchmarks for researching agents capable of effectively remembering information and establishing long-term dependencies [Parisotto *et al.*, 2020; Pleines *et al.*, 2022; Sorokin *et al.*, 2022].

Treasure Hunting. The primary objective is to retrieve the hidden treasure, with a series of rooms providing keys and clues leading to the final goal. The basic variation has 12

[1] Strictly speaking, episodic edges cannot be called edges or even hyperedges, because they connect vertices with multiple graph edges, but for simplicity we call them edges or episodic edges.

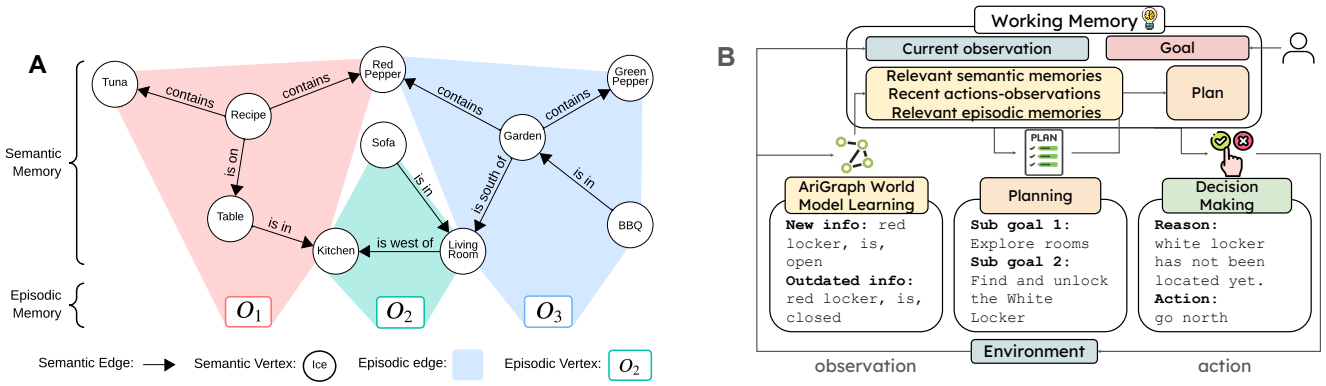

Figure 2: AriGraph world model and Ariadne cognitive architecture. (A) AriGraph learns episodic and semantic knowledge during interaction with unknown environment. At each time step t new episodic vertex (containing full textual observation o_t) is added to the episodic memory. Then LLM model parses observation o_t to extract relevant relationships in a form of triplets $(object_1, relation, object_2)$. These triplets are used to update semantic memory graph. The connection between episodic and semantic memory occurs through episodic edges that link each episodic vertex with all triplets extracted from respective observation. (B) Ariadne agent explores the environment and accomplishes tasks with AriGraph. User sets goal to the agent. Working memory is populated with recent history of observations and actions, relevant semantic and episodic knowledge retrieved from the AirGraph world model. Planing LLM module uses content of working memory to generate new or update existing plan. Results of planning are stored back in working memory. Finally, a ReAct-based module reads memory content and selects one of possible actions to be executed in the environment. Every observation triggers learning that updates agent's world model.

rooms and 4 keys, hard one has 16 rooms and 5 keys and hardest contains 36 rooms, 7 keys and additional distract items in every room.

Cleaning. The goal is to clean a house by identifying and returning misplaced items to their correct locations. Environment consists of 9 rooms (kitchen, pool, etc.) and contains 11 misplaced items (among many other items). To solve the problem, the agent needs to memorize the location of rooms and objects, as well as reason about objects placement.

Cooking. The goal is to prepare and consume a meal by following a recipe, selecting the correct ingredients, and using appropriate tools, while navigating in multi-room house. Basic difficulty task features 9 locations and 3 ingredients and hard task features 12 locations and 4 ingredients, while hardest task also features closed doors and inventory management.

For baselines we used Ariadne's planning and decision making module with one of the following types of memory instead of AriGraph model: full history of observations and actions, iterative summarization, RAG, RAG with Reflexion [Shinn *et al.*, 2023], and Simulacra - memory implementation from [Park *et al.*, 2023].

Full history involves retaining a complete record of all observations and actions to inform decision-making at every step. *Summarization*, as an alternative to storing the full history, focuses on retaining only the necessary information while discarding the rest. The standard *RAG* baseline retrieves top-k memories based on their similarity score to the current observation and plan. *Simulacra* features a scoring mechanism that integrates recency, importance, and relevance, alongside reflections on the extracted memories. The *Reflexion* baseline differs from other methods in its approach, as it operates over multiple trials. After failing a trial, the agent reflects on its trajectories to document information that may assist in solving the task in subsequent trials. We used the gpt-4-0125-preview as LLM backbone for AriGraph and other LLM-based baselines.

Additionally, we tested our architecture on a variation of the cooking test from [Adhikari *et al.*, 2021] to compare it with RL baselines. These tasks have 4 levels of difficulty, however, they are significantly simpler than our main tasks, having fewer locations, ingredients, and required actions (Appendix F).

For *RL* baselines, we collect the best results reported by [Adhikari *et al.*, 2021; Tuli *et al.*, 2022; Basu *et al.*, 2024] for the GATA, LTL-GATA, and EXPLORER architectures on the Cooking task with four difficulties levels from [Adhikari *et al.*, 2021].

To estimate human performance in the same games, we developed a graphical user interface , allowing volunteers to play basic versions of the Treasure Hunt, The Cleaning, and the Cooking. After collecting the data, we excluded sessions where the game was not completed.

4.2 NetHack Environment

NetHack [Küttler *et al.*, 2020] is a classic roguelike adventure game featuring procedurally generated multi-level dungeon (see Figure 13 in Appendix for a dungeon level example). It poses significant challenges for both LLM-based and RL-based approaches, requiring complex exploration, resource management, and strategic planning.

We based our experiments on NetPlay [Jeurissen *et al.*, 2024] agent, which demonstrates state-of-the-art performance among LLM agents that do not rely on finetuning or RL. In NetPlay agent receives textual observations containing all information about current explored dungeon level. These observations (*Level obs*) effectively function as handcrafted memory oracle for the agent.

To evaluate our Ariadne agent, we restricted textual observations to agent's current room or corridor (*Room Obs*),

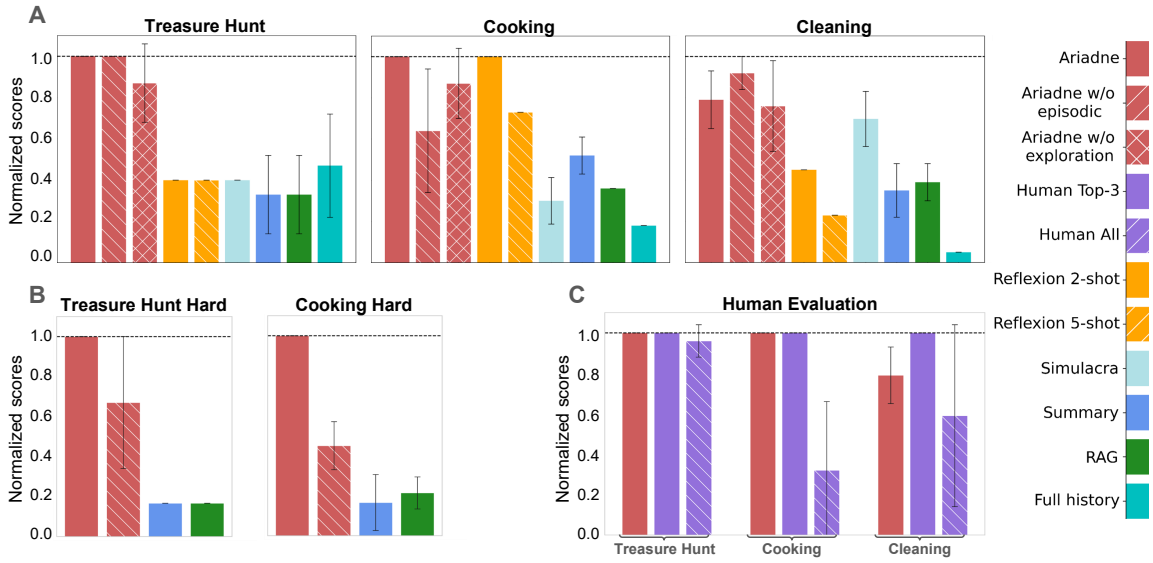

Figure 3: AriGraph world model enables Ariadne agent to successfully solve variety of text games. (A) Ariadne outperform baseline agents with alternative types of memory. (B) Ariadne with episodic and semantic memory scales to harder environments without losing performance. (C) Ariadne shows performance comparable to the best human players. The Y-axis shows the normalized score, which is calculated relative to the maximum possible points that can be obtained in each environment. Error bars show standard deviation. The number of max steps is set to 60 in the Cooking and to 150 in other games.

testing whether AriGraph world model could compensate for this restriction by remembering all relevant level information.

We compare three agents. The first is *NetPlay [Room obs]* with restricted textual observations, the second is our *Ariadne [Room obs]* agent that receives Room Obs and updates AriGraph, and the last is *NetPlay [Level obs]* with access to information about explored level.

4.3 Multi-hop Q&A

Although our memory architecture was originally designed for an agent interacting with the environment, we evaluated its performance on standard multi-hop Q&A benchmarks — Musique [Trivedi *et al.*, 2022] and HotpotQA [Yang *et al.*, 2018] to show its robustness and efficiency in more standard retrieval tasks. We made slight adjustments to the promts and replaced Contriever model with BGE-M3[Chen *et al.*, 2024], as it is a better fit for general text encoding. We used 200 random samples from both datasets similar to [Li *et al.*, 2024a]. We compared the performance of our approach against Graphreader [Li *et al.*, 2024a], ReadAgent [Lee *et al.*, 2024], HOLMES [Panda *et al.*, 2024], GraphRAG [Edge *et al.*, 2024] and RAG baselines provided in [Li *et al.*, 2024a].

5 Results

5.1 TextWorld

Every LLM based agent had five attempts to solve each game. The normalized score of one means that an agent completed the game, and score less than one represents intermediate progress. Results on text-based games are shown on the Figure 3 (for dynamics see Appendix G). We estimate performance as average of three best runs. Ariadne successfully remembers and uses information about state of the world for all three tasks. Baseline agents are unable to solve the Treasure Hunt, and fail to find even second key in the Treasure Hunt Hardest. On the other hand, Ariadne successfully solves the Treasure Hunt in about fifty steps, maintains robust performance in the Treasure Hunt Hard, and is able to complete the Treasure Hunt Hardest with more then double amount of rooms compared to Hard version, additional keys and distractors (see Appendix G).

Compared to the Treasure Hunt, the Cleaning game it is more important to properly filter outdated information about object locations, than not to lose any information. This is evident from the reduced usefulness of Episodic Memory in Ariadne agent and Full history baseline, since both memory modules focus on retaining long-term information. Overall Ariadne notably outperforms alternatives in this game. Moreover, Ariadne also outperforms Reflexion, which has additional information between episodes [Shinn *et al.*, 2023]. This baseline shows markable performance growth (in comparison wuth RAG) at the second try, but degrades with following tries.

The Cooking game has the highest difficulty, because any error at intermediate step prevents completion of the whole game. All baseline agents (except Reflexion 2-shot with obvious advantage over other methods) fail to complete cooking tasks due to insufficient or misused information. In this game, episodic memory is particularly important, allowing the agent to recall useful observations such as the content of the recipe or cooking instructions. For token usage of every method see Table 3, Appendix D.

Comparison with RL baselines on variation of the Cooking task is shown in Figure 4. We run Ariadne and GPT-4 with Full history on 4 difficulty levels from the cooking benchmark

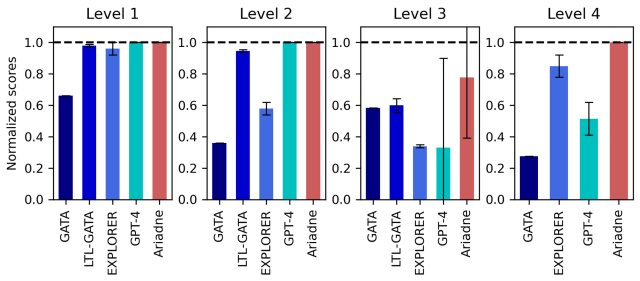

Figure 4: Ariadne LLM agent shows top performance compared to RL alternatives. Comparison of Ariadne and Full History baseline (GPT-4) with RL baselines in the cooking benchmark. Ariadne demonstrates superior performance across all 4 difficulty levels

[Adhikari *et al.*, 2021]. Ariadne shows superior performance to RL-agents on all 4 levels, especially harder ones. GPT-4 agent with Full history solves only first two levels which is consistent with previous result as the Cooking from Figure 3.A is harder than level 4.

Human evaluation. Comparison with respect to human players is shown in Figure 3.C. *All Humans* is the average score of all valid (completed) human trials. *Human Top-3* is the average score of three best plays for each task. Ariadne outperforms average human player from our sample on all tasks, and scores similarly to the best human plays in the Cooking and the Treasure Hunt, but underperforms in the Cleaning.

Graph quality. We measured AriGraph's growth rate and update rate during gameplay (see Figure 5). The graph actively grows during the exploration phase and flattens once the agent becomes familiar with the environment. We argue that this indicates that agent can generalize to long interactions with the environment despite constant updates to the semantic graph. Additional results in Appendix C demonstrate that the growth rate of the graph decreases with the increase in quality of LLM backbone.

Overal results demonstrate clear advantage of Ariadne agent over LLM based and RL baselines. Semantic memory enables the Ariadne Agent to build and update knowledge about the current state of the POMDP environment, which is crucial for navigation, exploration and capturing relevant details in interactive environments. On the other hand, episodic memory assists the agent in retrieving detailed long-term information that may not be captured in semantic memory, as demonstrated by the results in the Cooking task.

5.2 NetHack

The results are presented in Table 1. Scores column shows average game score across 3 runs, Levels column shows average number of dungeon levels completed by an agent. GPT-4o was used for all agents. Underscoring the importance of memory in this task, *NetPlay [Level obs]* with access to memory oracle achieved the highest scores, while *NetPlay [Room obs]* with only current room observations performed the worst. *Ariadne [Room obs]* successfully utilized AriGraph word model, achieving performance comparable to the baseline with memory oracle.

Method	Score	Levels
Ariadne (Room obs)	593.00 ± 202.62	6.33 ± 2.31
NetPlay (Room obs)	341.67 ± 109.14	3.67 ± 1.15
NetPlay (Level obs)	675.33 ± 130.27	7.33 ± 1.15

Table 1: Ariadne with obscured partial observations performs comparable to NetPlay agent full level information.

5.3 Multi-hop Q&A

We compared AriGraph with the latest LLM-based approaches that employ knowledge graph construction and retrieval techniques for question answering over documents (Table 2). Our memory architecture, adapted from the Ariadne TextWorld agent, utilizing both GPT-4 and GPT-4o-mini outperformed baseline methods like ReadAgent (GPT-4), GPT-4 RAG, GPT-4 full context and GraphReader (GPT-4). GraphRAG served as a strong GPT-4o-mini baseline, due to its extremely hight costs. ArigGraph (GPT-4o-mini) showed weaker performance on Musique, but outperformed GraphRAG on HotpotQA. Notably, our approach is more then 10x cheaper in comparison to GraphRAG (Table 3, Appendix D).

The best performance using GPT-4 was achieved by HOLMES, but AriGraph (GPT-4) exhibited comparable results. Notably, all baseline methods were specifically designed for Q&A tasks, incorporating task-specific prompt tuning and additional architectural enhancements. Both GraphRAG and HOLMES employ hyper-relations in their graphs to connect source data with extracted entities, similar to our method. However, these approaches lack mechanisms for updates in dynamic environments, a key advantage of AriGraph.

Method	MuSiQue		HotpotQA	
	EM	F1	EM	F1
BM25(top-3)	25.0	31.1	45.7	58.5
Ada-002(top-3)	24.5	32.1	45.0	58.1
GPT-4 full context	33.5	42.7	53.0	68.4
GPT-4 + supporting facts	45.0	56.0	57.0	73.8
ReadAgent(GPT-4)	35.0	45.1	48.0	62.0
GraphReader(GPT-4)	38.0	47.4	55.0	70.0
HOLMES(GPT-4)	**48.0**	**58.0**	66.0	**78.0**
AriGraph(GPT-4)	45.0	57.0	**68.0**	74.7
GraphRAG(GPT-4o-mini)	<u>40.0</u>	<u>53.5</u>	58.7	63.3
AriGraph(GPT-4o-mini)	36.5	47.9	<u>60.0</u>	<u>68.6</u>

Table 2: AriGraph memory demonstrates competitive performance on Multi-Hop Q&A datasets. Even in non interactive tasks AriGraph is comparable to strong QA baseline agents. The best results with the base GPT-4o and GPT-4o-mini are shown in bold and underline respectively.

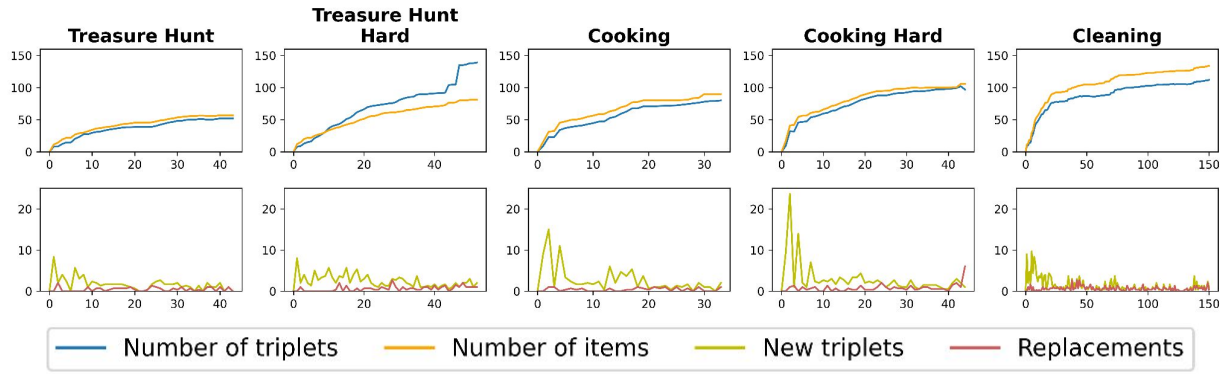

Figure 5: AriGraph demonstrate good scaling during learning and with environment size. A size of the knowledge graph quickly saturates during exploration and learning phase. KG grows moderately when the Treasure Hunt and the Cooking games include more rooms and objects in their hard versions.

6 Related Work

Voyager [Wang *et al.*, 2023a], Ghost in the Minecraft [Zhu *et al.*, 2023] and Jarvis-1 [Wang *et al.*, 2023b] are advanced, open-ended LLM agents that show significantly better performance in Minecraft compared to earlier techniques. These agents feature memory capabilities through a library of learned skills, summaries of successful actions, and episodic memory with plans for successful task execution. However, they fall short in representing knowledge with semantic structure and depend heavily on the LLM's extensive Minecraft knowledge. Generative agents [Park *et al.*, 2023] mimic human behavior in multi-agent environments and were among the pioneers in introducing an advanced memory system for LLM agents. Reflexion [Shinn *et al.*, 2023] and CLIN [Majumder *et al.*, 2023] enables agents to reflect on past trajectories, allowing them to store relevant insights about completed actions in a long-term memory module, but has no structural representation of knowledge and episodic memories. LARP [Yan *et al.*, 2023] utilizes the concepts of episodic and semantic memories but treats them as separate instances and lacks a structural representation of knowledge.

Considerable research is dedicated to leveraging established knowledge graphs for enhancing Q&A [Baek *et al.*, 2023; Li *et al.*, 2024b] systems to address the factual knowledge deficiency observed in LLMs. The latest research demonstrating best performance in Q&A tasks includes Graphreader [Li *et al.*, 2024a], HOLMES [Panda *et al.*, 2024], HippoRAG [Gutiérrez *et al.*, 2024], GraphRAG [Edge *et al.*, 2024] which all employ the technique of building knowledge graphs from texts. However, these studies do not address the context of functioning within an interactive environment, nor do they take into account the updates to knowledge graphs prompted by new experiences.

Text-based environments [Côté *et al.*, 2018; Hausknecht *et al.*, 2019; Shridhar *et al.*, 2021; Wang *et al.*, 2022] were originally designed to evaluate reinforcement learning (RL) agents [Guo *et al.*, 2020; Yao *et al.*, 2020; Ammanabrolu *et al.*, 2020; Ammanabrolu and Hausknecht, 2020; Tuli *et al.*, 2022; Adhikari *et al.*, 2021]. Multiple experiments have already explored the potential of LLMs in these complex scenarios [Tsai *et al.*, 2023; Tan *et al.*, 2023; Momennejad *et al.*, 2023; Ding *et al.*, 2024a]. However raw LLMs show poor results in these games without proper agentic architecture and memory.

7 Conclusions

In this paper, we introduced AriGraph, a novel knowledge graph world model tailored for LLM agents. AriGraph uniquely integrates semantic and episodic memories from textual observations, providing a structured and dynamic representation of knowledge. We evaluated this approach across a range of interactive text-based games and multi-hop Q&A benchmarks, comparing it against existing memory architectures. To test its capabilities comprehensively, we developed a cognitive architecture called Ariadne, which combines AriGraph with planning and decision-making components.

Our results demonstrate that AriGraph significantly outperforms other memory systems in tasks requiring long-term memory, such as decision-making, planning, and exploration in partially observable environments. The structured knowledge representation provided by AriGraph enables efficient retrieval and reasoning, accelerating learning and task completion. Additionally, AriGraph's scalability was evident as it maintained high performance even when the complexity of tasks increased, involving more objects and locations. In multi-hop Q&A benchmarks, AriGraph exhibited competitive performance, underscoring its robustness and adaptability beyond interactive environments.

While promising, our approach can be further enhanced by incorporating multi-modal observations, procedural memories, and more sophisticated graph search methods.

Acknowledgments

This work was supported by the Ministry of Economic Development of the Russian Federation (code 25-139-66879-1-0003).

References

[Adhikari *et al.*, 2021] Ashutosh Adhikari, Xingdi Yuan, Marc-Alexandre Côté, Mikuláš Zelinka, Marc-Antoine Rondeau, Romain Laroche, Pascal Poupart, Jian Tang, Adam Trischler, and William L. Hamilton. Learning dynamic belief graphs to generalize on text-based games, 2021.

[Ammanabrolu and Hausknecht, 2020] Prithviraj Ammanabrolu and Matthew Hausknecht. Graph constrained reinforcement learning for natural language action spaces, 2020.

[Ammanabrolu *et al.*, 2020] Prithviraj Ammanabrolu, Ethan Tien, Matthew Hausknecht, and Mark O. Riedl. How to avoid being eaten by a grue: Structured exploration strategies for textual worlds, 2020.

[Baek *et al.*, 2023] Jinheon Baek, Alham Fikri Aji, and Amir Saffari. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering, 2023.

[Basu *et al.*, 2024] Kinjal Basu, Keerthiram Murugesan, Subhajit Chaudhury, Murray Campbell, Kartik Talamadupula, and Tim Klinger. Explorer: Exploration-guided reasoning for textual reinforcement learning, 2024.

[Bulatov *et al.*, 2022] Aydar Bulatov, Yuri Kuratov, and Mikhail S. Burtsev. Recurrent memory transformer, 2022.

[Bulatov *et al.*, 2024] Aydar Bulatov, Yuri Kuratov, Yermek Kapushev, and Mikhail S. Burtsev. Scaling transformer to 1m tokens and beyond with rmt, 2024.

[Chen *et al.*, 2024] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation, 2024.

[Cheng *et al.*, 2024] Yuheng Cheng, Ceyao Zhang, Zhengwen Zhang, Xiangrui Meng, Sirui Hong, Wenhao Li, Zihao Wang, Zekai Wang, Feng Yin, Junhua Zhao, and Xiuqiang He. Exploring large language model based intelligent agents: Definitions, methods, and prospects, 2024.

[Côté *et al.*, 2018] Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Ruo Yu Tao, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. Textworld: A learning environment for text-based games. *CoRR*, abs/1806.11532, 2018.

[Ding *et al.*, 2024a] Peng Ding, Jiading Fang, Peng Li, Kangrui Wang, Xiaochen Zhou, Mo Yu, Jing Li, Matthew Walter, and Hongyuan Mei. MANGO: A benchmark for evaluating mapping and navigation abilities of large language models, 2024.

[Ding *et al.*, 2024b] Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. Longrope: Extending llm context window beyond 2 million tokens, 2024.

[Edge *et al.*, 2024] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. From local to global: A graph rag approach to query-focused summarization, 2024.

[Gu and Dao, 2023] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2023.

[Guo *et al.*, 2020] Xiaoxiao Guo, Mo Yu, Yupeng Gao, Chuang Gan, Murray Campbell, and Shiyu Chang. Interactive fiction game playing as multi-paragraph reading comprehension with reinforcement learning, 2020.

[Gutiérrez *et al.*, 2024] Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. Hipporag: Neurobiologically inspired long-term memory for large language models, 2024.

[Hausknecht *et al.*, 2019] Matthew Hausknecht, Prithviraj Ammanabrolu, Côté Marc-Alexandre, and Yuan Xingdi. Interactive fiction games: A colossal adventure. *CoRR*, abs/1909.05398, 2019.

[Izacard *et al.*, 2022] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning, 2022.

[Jeurissen *et al.*, 2024] Dominik Jeurissen, Diego Perez-Liebana, Jeremy Gow, Duygu Cakmak, and James Kwan. Playing nethack with llms: Potential and limitations as zero-shot agents, 2024.

[Küttler *et al.*, 2020] Heinrich Küttler, Nantas Nardelli, Alexander Miller, Roberta Raileanu, Marco Selvatici, Edward Grefenstette, and Tim Rocktäschel. The nethack learning environment. *Advances in Neural Information Processing Systems*, 33:7671–7684, 2020.

[Lee *et al.*, 2024] Kuang-Huei Lee, Xinyun Chen, Hiroki Furuta, John Canny, and Ian Fischer. A human-inspired reading agent with gist memory of very long contexts, 2024.

[Li *et al.*, 2024a] Shilong Li, Yancheng He, Hangyu Guo, Xingyuan Bu, Ge Bai, Jie Liu, Jiaheng Liu, Xingwei Qu, Yangguang Li, Wanli Ouyang, Wenbo Su, and Bo Zheng. Graphreader: Building graph-based agent to enhance long-context abilities of large language models, 2024.

[Li *et al.*, 2024b] Yihao Li, Ru Zhang, Jianyi Liu, and Gongshen Liu. An enhanced prompt-based llm reasoning scheme via knowledge graph-integrated collaboration, 2024.

[Majumder *et al.*, 2023] Bodhisattwa Prasad Majumder, Bhavana Dalvi Mishra, Peter Jansen, Oyvind Tafjord, Niket Tandon, Li Zhang, Chris Callison-Burch, and Peter Clark. Clin: A continually learning language agent for rapid task adaptation and generalization, 2023.

[Momennejad *et al.*, 2023] Ida Momennejad, Hosein Hasanbeig, Felipe Vieira Frujeri, Hiteshi Sharma, Nebojsa Jojic, Hamid Palangi, Robert Ness, and Jonathan Larson. Evaluating cognitive maps and planning in large language models with cogeval. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[Pan *et al.*, 2024] Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, page 1–20, 2024.

[Panda *et al.*, 2024] Pranoy Panda, Ankush Agarwal, Chaitanya Devaguptapu, Manohar Kaul, and Prathosh A P. Holmes: Hyper-relational knowledge graphs for multi-hop question answering using llms, 2024.

[Parisotto *et al.*, 2020] Emilio Parisotto, Francis Song, Jack Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant Jayakumar, Max Jaderberg, Raphael Lopez Kaufman, Aidan Clark, Seb Noury, et al. Stabilizing transformers for reinforcement learning. In *International conference on machine learning*, pages 7487–7498. PMLR, 2020.

[Park *et al.*, 2023] Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior, 2023.

[Pleines *et al.*, 2022] Marco Pleines, Matthias Pallasch, Frank Zimmer, and Mike Preuss. Memory gym: Partially observable challenges to memory-based agents. In *The eleventh international conference on learning representations*, 2022.

[Shinn *et al.*, 2023] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023.

[Shridhar *et al.*, 2021] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[Sorokin *et al.*, 2022] Artyom Sorokin, Nazar Buzun, Leonid Pugachev, and Mikhail Burtsev. Explain my surprise: Learning efficient long-term memory by predicting uncertain outcomes. *Advances in Neural Information Processing Systems*, 35:36875–36888, 2022.

[Sumers *et al.*, 2024] Theodore R. Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas L. Griffiths. Cognitive architectures for language agents, 2024.

[Tan *et al.*, 2023] Qinyue Tan, Ashkan Kazemi, and Rada Mihalcea. Text-based games as a challenging benchmark for large language models, 2023.

[Trivedi *et al.*, 2022] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 2022.

[Tsai *et al.*, 2023] Chen Feng Tsai, Xiaochen Zhou, Sierra S. Liu, Jing Li, Mo Yu, and Hongyuan Mei. Can large language models play text games well? current state-of-the-art and open questions, 2023.

[Tuli *et al.*, 2022] Mathieu Tuli, Andrew C. Li, Pashootan Vaezipoor, Toryn Q. Klassen, Scott Sanner, and Sheila A. McIlraith. Learning to follow instructions in text-based games, 2022.

[Wang *et al.*, 2022] Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. Scienceworld: Is your agent smarter than a 5th grader?, 2022.

[Wang *et al.*, 2023a] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023.

[Wang *et al.*, 2023b] Zihao Wang, Shaofei Cai, Anji Liu, Yonggang Jin, Jinbing Hou, Bowei Zhang, Haowei Lin, Zhaofeng He, Zilong Zheng, Yaodong Yang, Xiaojian Ma, and Yitao Liang. Jarvis-1: Open-world multi-task agents with memory-augmented multimodal language models, 2023.

[Wang *et al.*, 2024] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Jirong Wen. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6), March 2024.

[Wong Gonzalez, 2018] Daniela Wong Gonzalez. *The Relationship Between Semantic and Episodic Memory: Exploring the Effect of Semantic Neighbourhood Density on Episodic Memory*. PhD thesis, Electronic Theses and Dissertations, 2018. Paper 7585.

[Yan *et al.*, 2023] Ming Yan, Ruihao Li, Hao Zhang, Hao Wang, Zhilan Yang, and Ji Yan. Larp: Language-agent role play for open-world games, 2023.

[Yang *et al.*, 2018] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering, 2018.

[Yao *et al.*, 2020] Shunyu Yao, Rohan Rao, Matthew Hausknecht, and Karthik Narasimhan. Keep calm and explore: Language models for action generation in text-based games, 2020.

[Yao *et al.*, 2023] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023.

[Zhu *et al.*, 2023] Xizhou Zhu, Yuntao Chen, Hao Tian, Chenxin Tao, Weijie Su, Chenyu Yang, Gao Huang, Bin Li, Lewei Lu, Xiaogang Wang, Yu Qiao, Zhaoxiang Zhang, and Jifeng Dai. Ghost in the minecraft: Generally capable agents for open-world environments via large language models with text-based knowledge and memory, 2023.