Reasoning with Graphs: Structuring Implicit Knowledge to Enhance LLMs Reasoning

Haoyu Han[1], Yaochen Xie[2], Hui Liu[2,1], Xianfeng Tang[2], Sreyashi Nag[2], William Headden[2], Yang Li[2], Chen Luo[2], Shuiwang Ji[3], Qi He[2], Jiliang Tang[1]
[1]Michigan State University, [2]Amazon, [3]Texas A&M University
{hanhaoy1, liuhui7, tangjili}@msu.edu, sji@tamu.edu
{yaochx, liunhu, xianft, sreyanag, headdenw, limyng, cheluo, qih}@amazon.com

Abstract

Large language models (LLMs) have demonstrated remarkable success across a wide range of tasks; however, they still encounter challenges in reasoning tasks that require understanding and inferring relationships between distinct pieces of information within text sequences. This challenge is particularly pronounced in tasks involving multi-step processes, such as logical reasoning and multi-hop question answering, where understanding implicit relationships between entities and leveraging multi-hop connections in the given context are crucial. Graphs, as fundamental data structures, explicitly represent pairwise relationships between entities, thereby offering the potential to enhance LLMs' reasoning capabilities. External graphs have proven effective in supporting LLMs across multiple tasks. However, in many reasoning tasks, no pre-existing graph structure is provided. Can we structure implicit knowledge derived from context into graphs to assist LLMs in reasoning? In this paper, we propose Reasoning with Graphs (RwG) by first constructing explicit graphs from the context and then leveraging these graphs to enhance LLM reasoning performance on reasoning tasks. Extensive experiments demonstrate the effectiveness of the proposed method in improving both logical reasoning and multi-hop question answering tasks.

1 Introduction

In recent years, Large Language Models (LLMs) have demonstrated remarkable capabilities in a variety of tasks, such as question answering (Zhuang et al., 2024; Lan et al., 2022), summarization (Pu et al., 2023), and language understanding (Zhao et al., 2023). Despite these successes, LLMs still face significant challenges in certain areas (Zhao et al., 2023; Minaee et al., 2024). A key limitation lies in their struggle with reasoning tasks (Yang et al., 2024; Huang et al., 2023), particularly with

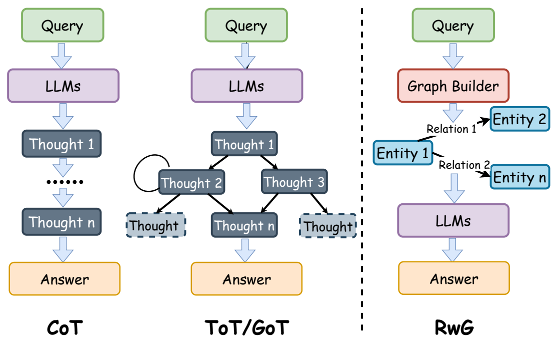


Figure 1: Comparison of Reasoning with Graph (RwG) to other prompting methods.

logical reasoning (Nezhurina et al., 2024), which requires models to infer missing relationships between distinct pieces of information, and multi-hop reasoning (Yang et al., 2024), where they must trace a reasoning path or follow some structures through the context to arrive at the correct answer.

To enhance the reasoning capabilities of LLMs, several prompting methods have been proposed. Chain-of-Thought (CoT) prompting (Wei et al., 2022; Kojima et al., 2022) aids LLMs in reasoning by generating intermediate steps that lead to the final answer. CoT has shown significant improvement in certain reasoning tasks without requiring model tuning. Building on CoT, the Self-Consistency method (Wang et al., 2022b) further enhances reasoning by generating multiple CoT pathways and selecting the most consistent one. Additionally, Tree of Thought (ToT) (Yao et al., 2024) and Graph of Thoughts (GoT) (Besta et al., 2024) extend this approach by structuring the LLMs' thought process using trees and graphs, respectively. These methods prompt LLMs to generate initial thoughts and organize them into various structures. Despite the successes of these approaches, they still face challenges in handling complex reasoning tasks, such as logical reasoning (Nezhurina et al., 2024) and multi-hop question answering (Yang et al., 2024).

Findings of the Association for Computational Linguistics: ACL 2025, pages 25698–25714

For these complex reasoning tasks, LLMs need to figure out the relationships between entities in the context and infer missing components. Take the AIW+ problem (Nezhurina et al., 2024) as one example, LLMs are asked to solve problems such as *Alice has 3 sisters. Her mother has 1 sister who does not have children - she has 7 nephews and nieces and also 2 brothers. Alice's father has a brother who has 5 nephews and nieces in total, and who has also 1 son. How many cousins does Alice's sister have?* In this problem, LLMs need to infer the relationships between each character, such as the relation between Alice and her mother's sister. Additionally, LLMs must infer the missing roles in the question, such as identifying the nephews and nieces that Alice's father's brother has. These types of questions pose significant challenges for LLMs, with many popular models achieving nearly zero accuracy on these tasks (Nezhurina et al., 2024). Typically, LLMs treat the information as a sequence. However, a human solving such a problem would naturally draw a graph to represent relationships between characters and infer missing links based on that structure. This is because graphs provide a fundamental data structure for representing relationships between entities, making them well-suited as reasoning graphs for reasoning tasks.

Several works have shown the effectiveness of leveraging external graphs to help LLMs in reasoning, such as improving retrieval quality using graph structures (He et al., 2024; Tian et al., 2024) or reasoning on an external graph (Jin et al., 2024; Luo et al., 2023). However, these methods rely on pre-existing graph structures. In most common reasoning tasks, only textual sequences are available. Therefore, a natural question arises: "Can LLMs enhance their reasoning abilities by structuring implicit knowledge into explicit graphs?"

In this work, we aim to explore reasoning with graphs by constructing explicit graph structures from the context. Unlike previous prompting approaches, which construct trees or graphs based on LLMs' thoughts, our Reasoning with Graphs (RwG) method directly constructs explicit graphs from the context, where nodes are the entities in the context. The comparison is shown in Figure 1. Specifically, we first design a graph construction method with multiple rounds of verification to generate a graph from the given context for the reasoning problem. We then assess the LLMs' reasoning abilities with the constructed graph. Experimental results demonstrate that the proposed RwG significantly improves the performance of various LLMs on both logical reasoning and multi-hop question answering tasks. RwG showcases the potential of leveraging explicit graph structures derived from the context to enhance LLM reasoning capabilities, offering a promising new direction for incorporating structured knowledge into LLM-driven tasks.

2 Related Works

2.1 Reasoning of Large Language Models

Reasoning is a fundamental aspect of human intelligence, crucial for problem solving, decision making, and critical thinking. Recent advancements in LLMs, such as GPT-4 (Achiam et al., 2023) and LLaMA-3 (Touvron et al., 2023), suggest that the ability for reasoning is already embedded within these large-scale models. Various prompting methods have been proposed to better utilize the reasoning capabilities of LLMs. Chain-of-Thought (CoT) (Wei et al., 2022) is one of the most popular methods, prompting LLMs to generate reasoning paths. Building on this concept, Tree-of-Thought (ToT) (Yao et al., 2024) and Graph-of-Thought (GoT) (Besta et al., 2024) similarly model different reasoning paths using tree or graph structures. In addition to designing prompts, adopting additional strategies, such as incorporating verifiers, has contributed to enhancing the reasoning abilities of large language models. For instance, self-consistency (Wang et al., 2022b) improves LLMs' reasoning by using majority voting among multiple generated paths. Studies by (Weng et al., 2022) and (Stechly et al., 2024) demonstrate that LLMs can benefit from self-verification or external verification methods. Additionally, other techniques have been introduced to enhance LLMs' reasoning abilities, such as in-context learning (Lampinen et al., 2022), fine-tuning (Rajani et al., 2019), and retrieval-augmented generation (RAG)(Huang and Chang, 2022; Qiao et al., 2022; Gao et al., 2023). Recent studies (Wang and Zhou, 2024) reveal that CoT reasoning paths can be elicited from pre-trained LLMs simply by altering the decoding process without explicit prompting. This demonstrates that the effectiveness of CoT lies in guiding LLMs toward different decoding paths; for example, CoT can choose longer and more reliable paths instead of relying on greedy decoding. In this paper, we explore a different approach to prompting LLMs' reasoning abilities. Rather than leveraging multiple generated thoughts, we model the reasoning problem as graphs, where the nodes represents entities in the question, and test the LLMs' ability to reason directly with these graph structures.

2.2 Graphs for LLMs

Graphs, which represent relationships between entities, are popular data structures widely used across various domains (Ma and Tang, 2021). Recently, numerous studies have explored the integration of graphs with LLMs (Chen et al., 2024; Han et al., 2024). Specifically, several works have sought to enhance LLMs' reasoning abilities using graphs by retrieving relevant information. These methods typically involve extracting ego subgraphs based on related nodes and edges (Zhang et al., 2022b; Tian et al., 2024) or paths within knowledge graphs (Luo et al., 2023). Furthermore, GraphReason (Cao, 2023) constructs a graph based on LLMs' outputs and then verifies the output using the graph. However, **these methods rely on external graphs or generate graphs based on LLMs' reasoning paths**; they do not explore the effects of directly constructing a graph from the reasoning problems. Two related approaches are worth mentioning: GE-Reasoning (Park et al., 2024), which decomposes multi-hop questions into sub-questions to form a graph and prompts LLMs to answer based on the chronological order of the graph and Structure-Guided Prompting (Cheng et al., 2024), which builds a graph from text to solve graph-based tasks. In this paper, we construct graphs from the context of complex reasoning questions and use these graphs to assist LLMs in their reasoning processes.

3 Reasoning with Graph

Many reasoning tasks involve inferring missing entities and relationships that are not explicitly presented in the question. Graphs provide an explicit structure to represent relationships between key entities and serve as a useful tool for inferring missing connections. However, reasoning problems typically do not come with explicit graph representations. Reasoning with Graph (RwG) teaches large language models to tackle complex reasoning questions by structuring the implicit knowledge within the questions into explicit graph representations and leveraging these graph structures to solve the problems. This mirrors how humans often solve complex reasoning problems — by organizing information in a structured way, such as drawing diagrams to clarify connections between concepts. In RwG, no additional or external graph information is used.

We roughly decompose the process of proposed RwG into two key stages: (1) **Graph Construction**: The graph construction prompt guides LLMs to build an explicit graph based on the context of the reasoning question. We expect the graph to meet different requirements depending on the tasks, which are detailed in sections 4. (2) **Reasoning with graph**: Once the graph is constructed, the reasoning question is answered by leveraging the information encoded in the graph structure. Next, we will provide a detailed explanation of each stage.

3.1 Graph Construction

The goal of this step is to construct a graph from unstructured reasoning problems, representing the relationships between the entities mentioned in the reasoning question. However, there may be missing entities or relationships that are not explicitly stated in the context. These missing elements could be critical to solve the reasoning question. Therefore, we should refine the constructed graph by inferring additional relationships or entities, ensuring that it satisfies the requirements of the reasoning problem.

Take the context of AIW+ problem as an example: *Alice has 3 sisters. Her mother has 1 sister who does not have children—she has 7 nephews and nieces and also 2 brothers. Alice's father has a brother who has 5 nephews and nieces in total, and who also has 1 son.* In the constructed graph, all the characters mentioned in the context, such as Alice, and Alice's father's brother should be included. Additionally, it should include missing roles and relationships. For instance, Alice and her sisters account for only 4 of the 5 nephews and nieces of her father's brother, implying that there is one missing individual, which should be included in the graph.

There are several traditional methods for graph construction, such as entity and relation extraction (Zhong et al., 2023). With the advancements in LLMs, recent works (Edge et al., 2024; Zhang and Soh, 2024) have also leveraged LLMs to automatically detect entities and relationships for graph construction. These methods can be used to generate an initial graph for reasoning questions.

However, the constructed initial graphs may only capture the entities and relationships explicitly mentioned in the context, which may not fully meet the requirements for different tasks. For example,

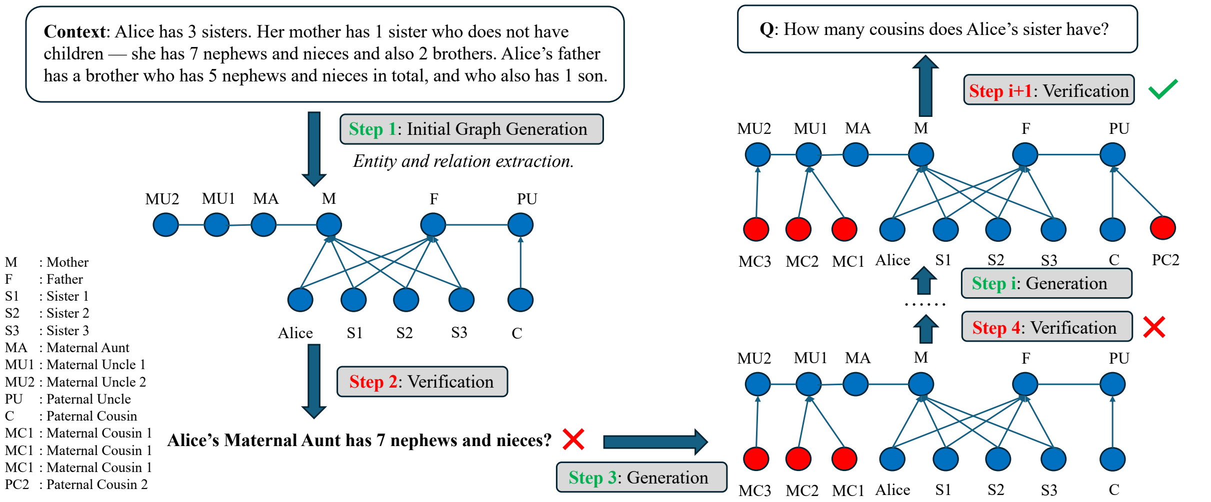


Figure 2: The procedure of RwG for the AIW+ example. Blue nodes represent entities explicitly mentioned in the context and included in the initial graph, while red nodes denote inferred entities added during the graph generation and verification processes. The node names are based on their relationship to Alice.

in the logical reasoning tasks, there might be some constraints in the context, such as "Alice's father's brother has 5 nephews and nieces". In multi-hop question-answering tasks, crucial relationships between entities may be missing, which are essential for reasoning. To address this, we propose an iterative graph construction method that updates the graph repeatedly to meet the specific requirements for different tasks. Specifically, this process mainly involves two steps: graph generation and graph verification. The graph generation step aims to construct a graph based on the context, previous graph and feedback from the verifier. The graph verification step verifies whether the generated graph meets the requirements.

We begin by prompting the LLMs to generate an initial graph based on the given query. Next, we ask the LLMs to verify whether the graph satisfies the requirements. If the graph does not meet the requirements, the LLMs are prompted to add the missing entities or relations to update the graph. This process of graph verification and graph generation is repeated until the graph satisfies all the requirements or the maximum number of iterations is reached. After construction, the graph is represented as a list of triples, with each triple consisting of a (Head Entity, Relation, Tail Entity).

The process of constructing a graph for the AIW+ example is illustrated in Figure 2. We first prompt the LLMs to extract entities and relationships from the context to generate an initial graph. The blue nodes represent entities explicitly stated in the context and are shown in the initial graph, while the red nodes are inferred during the multiple rounds of graph verification and generation. During the graph generation and verification steps, LLMs can better understand the context and infer missing relationships by utilizing the explicit graph structure. Once the graph is complete, it can then be used to answer the questions.

3.2 Reasoning with graphs

Reasoning with the graph involves answering reasoning questions using both the constructed graph and the given context. Many existing methods leverage external graphs, such as training a graph-based encoder (Tian et al., 2024; Zhang et al., 2022b), retrieving subgraphs (He et al., 2024; Zhang et al., 2022a), or reasoning along a path within the graph (Luo et al., 2023; Sun et al., 2023). The constructed graph can also be utilized in these ways. However, in this paper, we focus on having LLMs directly solve the reasoning question by leveraging the graph and context. We prompt LLMs to answer reasoning questions based on the constructed graph and context. Additional approaches to utilizing the constructed graph can be explored as future work.

4 Experiments

Our framework is inherently task-agnostic, designed to accommodate a wide range of tasks with versatility. To evaluate whether the proposed RwG approach can enhance LLMs' reasoning and grounded generation capabilities, we test it on two distinct reasoning tasks: logical reasoning and multi-hop question answering. In all experiments, we follow a zero-shot setting.

Table 1: The results on the AIW and AIW+ datasets. Since the AIW+ dataset contains many possible relationships, there are no RwG + Relation results.

Datasets	AIW				AIW+			
Methods	Claude	GPT-4o	Llama3.1-8B	Llama3.1-70B	Claude	GPT-4o	Llama3.1-8B	Llama3.1-70B
Vanilla	**0.026**	0.066	0.053	0.013	0.0588	0.1176	0	0.2941
CoT	0.013	0.5733	**0.066**	0.053	0	0.2352	**0.058**	0.3529
ToT	0	0.2800	0	0.066	0.0588	0.2941	0	0.3529
GoT	0	0.4533	0.040	0.093	0.0588	0.2352	0	0.2941
Self-Consistency	0	0.053	0	0	0.0588	0.0588	0	0.1176
Least-to-Most	0	0.4533	0.0266	0.053	0	0.1764	0	0.1764
RwG	**0.026**	0.6266	0	0.12	**0.2941**	**0.5294**	0	**0.4545**
RwG + Relation	**0.026**	**0.8666**	0.0266	**0.5733**	-	-	-	-

Table 2: The results on LogiQA and AR-LSAT datasets.

Datasets	LogiQA				AR-LSAT			
Methods	Claude	GPT-4o	Llama3.1-8B	Llama3.1-70B	Claude	GPT-4o	Llama3.1-8B	Llama3.1-70B
Vanilla	0.387	0.5698	0.1827	0.5698	0.2565	0.3608	0.1217	0.313
CoT	0.3978	0.5483	0.3548	0.5053	0.213	0.3565	0.1782	0.2434
ToT	0.4494	0.6021	0.3225	0.3978	0.2043	0.3304	0.2521	0.2913
GoT	0.3656	0.6236	0.3225	0.4838	0.2565	0.3782	0.1826	0.3217
Self-Consistency	0.3871	0.5806	0.172	0.5483	0.2608	0.3521	0.1217	0.2826
Least-to-Most	0.3225	0.5806	0.2795	0.5483	0.2652	0.3565	0.1695	0.2695
RwG	**0.4516**	**0.6344**	**0.3871**	**0.5913**	0.2782	0.4043	0.1826	0.3173
RwG + Self-Consistency	0.4408	0.6451	0.3548	0.5591	**0.3086**	**0.4521**	**0.2086**	**0.3217**

4.1 Task1: Logical Reasoning

Logical reasoning is a crucial aspect of human reading comprehension and question answering. A typical logical reasoning problem consists of a paragraph of facts and a question that requires the testee to draw a valid conclusion based on those facts. To generate a correct answer, a machine must not only understand the facts but also recognize the relationships between the different components in the question. By constructing a graph for the logical reasoning question, we explicitly extract the key entities and their relationships, while also inferring any missing entities and relations — an essential step for effective logical reasoning. The general generation and verification process for logical reasoning task in RwG is as follows: (1) **Generation**: Generate a graph based on the context by updating the previous graph, inferring missing entities and relations; (2) **Verification**: Verify whether the graph meets all requirements outlined in context.

4.1.1 Datasets

We selected four popular logical question answering datasets: AIW, AIW+(Nezhurina et al., 2024), LogiQA (Liu et al., 2020) and AR-LSAT (Wang et al., 2022a). Specifically, the AIW and AIW+ datasets mainly focus on answering questions related to Alice and her family. The LogiQA dataset includes various types of reasoning questions. The AR-LSAT dataset is a complex logical reasoning dataset that tests the ability to analyze a scenario governed by a set of constraints and determine which option satisfies or conflicts with those constraints. For these datasets, where the answers are numbers or options, we use accuracy as the evaluation metric. For more details on these datasets and pre-processing, please refer to Appendix A.1.1.

4.1.2 Baselines

We evaluate our method on four widely used LLMs: GPT-4o (Achiam et al., 2023), Claude 3-sonnet (Anthropic, 2024), LLaMA3.1 8B, and LLaMA3.1 70B (Touvron et al., 2023). Additionally, we compare our results with several representative baselines, such as the Chain-of-Thought (CoT) (Wei et al., 2022), Tree-of-Thought (ToT) (Yao et al., 2024), Graph-of-Thought (GoT) (Besta et al., 2024), Least-to-Most Prompting (Zhou et al., 2022), and Self-Consistency (Wang et al., 2022b). For the AIW dataset, there are only 3 different relations between entities, i.e., brother-brother, brother-sister, sister-sister. We introduce a variant of RwG called RwG-Relation, where explicit relationships are provided during the graph generation process. For RwG, we set the maximum number of graph generation and verification steps to 5. The prompts in the proposed RwG for these dataset are shown in Appendix A.2.

4.1.3 Results

The results of different LLMs on the AIW and AIW+ datasets are presented in Table 1, while the results for the LogiQA and LSAT datasets are shown in Table 2. From these results, we can make the following observations:

- Reasoning with graphs (RwG) generally enhances the logical reasoning capabilities of various LLMs on all datasets.
- RwG outperforms the ToT and GoT methods, which generate trees or graphs where nodes represent the thoughts of LLMs.
- Stronger models, such as GPT-4 and LLaMA 70B, tend to benefit more from RwG. However, if the original model struggles to solve the problem, as seen with LLaMA 3.1-8B on the AIW and AIW+ datasets, applying RwG usually does not yield significant improvements.
- When explicit relationships are provided, as in RwG-Relation for the AIW problem, the reasoning ability is further enhanced.
- The proposed RwG can be incorporated with other methods, such as Self-Consistency, and combining these approaches may achieve even better results, such as on AR-LSAT dataset.

4.1.4 Case studies

To understand why the proposed RwG improves performance on logical reasoning questions, we conduct case studies. Detailed results and additional examples are provided in Appendix A.4. Here, we analyze the behavior of RwG with GPT-4o using the example shown in Figure 2. The step-by-step procedure of RwG is presented in Figure 3. In the first phase, RwG generates an initial graph with the explicit entities mentioned in the question. The graph is then updated if it fails verification. For example, RwG adds more entities to the maternal and paternal parts during the first and second rounds of verification, respectively. In the third round, RwG identifies an incorrect relation from the second round and corrects it, successfully passing verification. Finally, the LLMs can answer the question correctly based on the complete graph. For other baselines, the LLMs may fail due to incomplete information, as demonstrated in Appendix A.4.1.

4.1.5 Analysis

Based on the case studies, the effectiveness of RwG may stem from its ability to infer missing entities and relationships. To validate this assumption, we designed an experiment where we manually added the missing roles and relationships in the AIW+ problem. Specifically, we completed the graph by incorporating the missing relationships. For example, we add *One of Alice's mother's brother has 1 son while another has 2 sons. Alice's father also have another brother who has 1 son.* to the AIW+ example. We refer to this dataset as AIW+ Complete.

AIW+ example with RwG

Initial Graph
Alice's Mother – Alice
Alice's Mother – Alice's Sister 1
...
1st Round Verification and Generation
Alice's Maternal Uncle 1–Maternal Cousin 1
Alice's Maternal Uncle 1–Maternal Cousin 2
Alice's Maternal Uncle 2–Maternal Cousin 3
2nd Round Verification and Generation
Alice's Paternal Uncle's Wife – Paternal Nephew 1
3rd Round Verification and Generation
~~*Alice's Paternal Uncle's Wife – Paternal Nephew 1*~~
Alice's Paternal Uncle – Paternal Cousin 2
The graph passes verification.
Answer: Total Cousins
3 (Maternal) + 2 (Paternal) = 5 Cousins

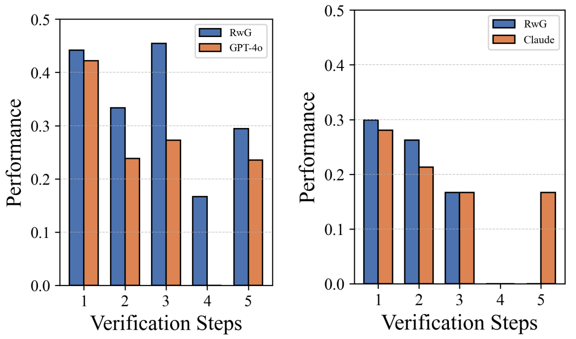
Figure 3: The graph updating procedure of RwG applied to the AIW+ example using GPT-4o.

The results are shown in Table 3. From these results, we observe that all models, except for LLaMA 3.1-8B, perform well on this dataset. The performance difference between the AIW+ and AIW+ Complete datasets demonstrates that missing entities and relationships in the questions are a major barrier to LLM reasoning. The proposed RwG addresses this issue by inferring the missing entities and relationships during the graph verification and generation processes, thereby improving performance. Additionally, RwG continues to improve performance on the AIW+ Complete dataset, demonstrating that explicit graph structures can assist LLMs with this task.

Table 3: The results of AIW+ Complete dataset.

	Claude	GPT-4o	Llama3.1-8B	Llama3.1-70B
Vanilla	0.5882	0.8823	0	0.7058
CoT	0.5294	0.9411	0	0.8823
RwG	0.7058	1	0.058	1

We further analyze the performance gain of the proposed RwG with respect to the number of verification steps. The number of verification and generation steps required to obtain the final graph varies depending on the question. If a question contains most of the entities and relationships, fewer verification steps are needed to construct the final graph. In contrast, if many verification and generation steps are required, the question is likely missing many entities and relationships, making it more difficult to solve. We select the AR-LSAT dataset and compare the performance of the proposed RwG with vanilla models, as shown in Figure 4.



(a) GPT-4o (b) Cluade

Figure 4: Comparison of performances under different verification steps.

We observe that when the verification step is 1, meaning the initial graph passes verification, the performance gap between RwG and the vanilla models is small. However, as more generation steps are required to pass verification, the performance gap increases, which aligns with our assumption.

4.2 Task 2: Multi-hop Question Answering

Multi-hop question answering typically provides several paragraphs of knowledge and requires answering a question that involves a sequence of interdependent reasoning steps leading to the final answer. These reasoning steps and their dependencies can often be represented as a directed acyclic graph (DAG). Therefore, the proposed RwG aims to extract such reasoning graphs from given context to answer the multi-hop question. Since the given context can be lengthy and LLMs struggle to comprehend large graphs (Dai et al., 2024), we build only a subgraph related to the question rather than constructing the entire graph. The general generation and verification process for the multi-hop question answering task in RwG is as follows: (1) **Generation:** Generate a graph related to the question by updating the previous graph, inferring missing relations, or adding more entities and relations from the context. (2) **Verification:** Verify whether the graph contains enough information to answer the multi-hop question.

4.2.1 Datasets

We selected four widely used multi-hop reasoning datasets: 2WikiMultihopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), HotpotQA (Yang et al., 2018) and Clutrr (Sinha et al., 2018). More details are shown in Appendix A.1.2

4.2.2 Baselines

We evaluate the proposed RwG for multi-hop question answering using two LLMs: Claude 3-sonnet (Anthropic, 2024) and GPT-4o (Achiam et al., 2023). Additionally, we choose the following baselines: CoT (Wei et al., 2022), ToT (Yao et al., 2024), GoT (Besta et al., 2024), Least-to-Most (Zhou et al., 2022), Structure-Guided Prompting (Cheng et al., 2024). The detailed prompts can be found in Appendix A.2.

4.2.3 Results

The overall performance on the selected datasets is shown in Table 4. Additionally, we evaluate the performance of different hop questions for the MuSiQue and Clutrr datasets, with results presented in Appendix A.3. Specifically, we illustrate the performance on different hop questions for the Clutrr dataset using Claude in Figure 5.

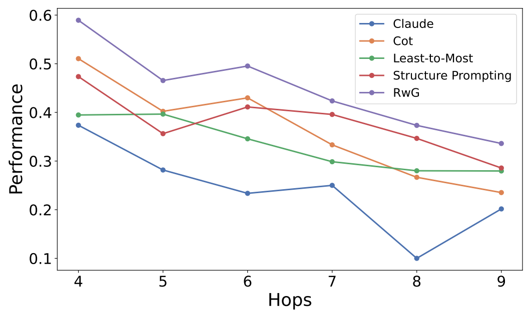


Figure 5: Performance on different hop questions in the Clutrr dataset.

From these results, we can make the following observations:

- The proposed RwG outperforms all baselines on these representative multi-hop question answering datasets.
- The performance of all methods tends to decrease as the number of hops increases in both the MuSiQue and Clutrr datasets. However, the proposed RwG performs well across both low and high hop questions.
- Chain-of-Thought (CoT) tends to perform well when the number of hops is low, but its performance declines with higher hop questions, especially when using Claude.

Table 4: Comparison of different models on the Multi-hop Question Answering datasets

Dataset	Hotpot		MuSiQue		2WikiMultihopQA		Clutrr	
Multi-hop QA	Claude	GPT-4o	Claude	GPT-4o	Claude	GPT-4o	Claude	GPT-4o
Vanilla	0.700	0.7219	0.5008	0.6131	0.6608	0.8493	0.2488	0.5485
CoT	0.6941	0.7294	0.5492	0.6064	0.8160	0.8660	0.3721	0.6594
ToT	0.7211	0.7589	0.5961	0.6452	0.8076	0.8859	0.2941	0.5764
GoT	0.7223	0.7666	0.5509	0.6539	0.7276	0.8826	0.2721	0.5135
Least-to-Most	0.6943	0.7452	0.5799	0.6331	0.8160	0.8859	0.3385	0.6503
Structure Prompting	0.6547	0.7435	0.5594	0.6094	0.7594	0.8660	0.3834	0.6413
RwG	**0.7399**	**0.7742**	**0.6395**	**0.7187**	**0.8202**	**0.9040**	**0.4558**	**0.6911**

4.2.4 Case studies

In this section, we aim to understand why the proposed RwG improves performance on multi-hop question answering tasks by analyzing several representative cases.

Case 1: We select one example from the Clutrr dataset, which requires LLMs to infer multi-hop family relationships between *Christian* and *Jeff*. The procedure of RwG with GPT-4o is shown in Figure 6. The proposed RwG would infer missing relationships during the graph construction. For example, after the first round of verification and generation, the LLMs inferred an edge between *Jason* and *Jeff*, reducing the reasoning path length between *Christian* and *Jeff* from 4 to 2.

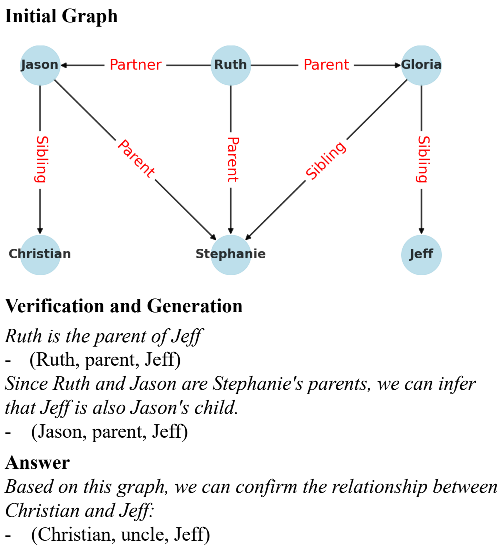


Figure 6: The illustration of Case 1.

More examples can be found in Appendix A.4. Based on the case studies, the proposed RwG aids LLMs in multi-hop question answering from two key perspectives: (1) The constructed graph reduces irrelevant information while maintaining an explicit reasoning structure; (2) The graph shortens the reasoning path length for the question.

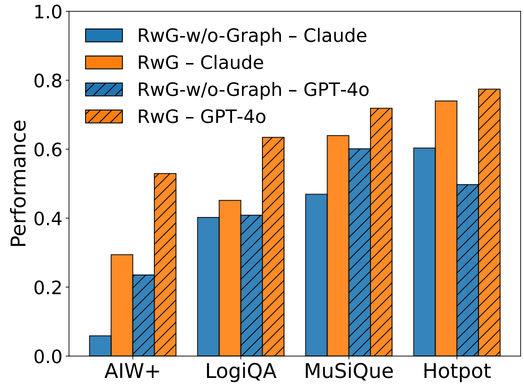


Figure 7: The comparison between variants of RwG

4.3 Analysis

We further analyze the necessity of using graphs to solve reasoning problems. To do so, we adopt a variant of RwG that follows the same verification and generation process but lacks an explicit graph structure, referred to as RwG-w/o-Graph. The results on both Reasoning and Multi-hop QA datasets are shown in Figure 7. We observe that RwG consistently outperforms RwG-w/o-Graph across all datasets, highlighting the importance of incorporating graph structures in the proposed RwG. Additionally, we analyze the computational complexity of RwG in Appendix A.5 and examine the effectiveness of the iterative generation and verification process in Appendix A.6. Furthermore, we evaluate RwG on other types of tasks, as shown in Appendix A.7.

5 Conclusion

In this paper, we propose a novel Reasoning with Graphs (RwG) method to structure implicit knowledge to enhance the reasoning capabilities of LLMs. Our method constructs graphs through multiple rounds of generation and verification, leveraging these graphs to answer complex questions. We evaluate our approach on both logical reasoning and multi-hop question-answering tasks using several widely recognized datasets. Experimental results demonstrate that RwG significantly improves the performance of various LLMs across both tasks.

6 Acknowledgment

Haoyu Han, Jiliang Tang are supported by the National Science Foundation (NSF) under grant numbers CNS2321416, IIS2212032, IIS2212144, IOS2107215, DUE2234015, CNS2246050, DRL2405483 and IOS2035472, US Department of Commerce, Gates Foundation, the Michigan Department of Agriculture and Rural Development, Amazon, Meta, and SNAP.

7 Limitations

In this paper, we aim to improve the reasoning ability of LLMs by modeling input as a graph structure, which mirrors the way humans often approach reasoning tasks. We conducted experiments on four popular LLMs: GPT-4o, Claude, Llama 3.1-8B, and Llama 3.1-70B. However, more LLMs can be tested with the proposed RwG in future studies. Additionally, while we explored why explicit graph structures can aid LLM reasoning primarily through experimental results and case studies, a more rigorous theoretical analysis is an interesting direction for future work. Furthermore, our evaluation focused on logical reasoning and multi-hop question answering tasks, but other tasks can also be explored to assess the broader applicability of RwG.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17682–17690.

Lang Cao. 2023. Enhancing reasoning capabilities of large language models: A graph-based verification approach. *arXiv preprint arXiv:2308.09267*.

Wenhu Chen, Xinyi Wang, and William Yang Wang. 2021. A dataset for answering time-sensitive questions. *arXiv preprint arXiv:2108.06314*.

Zhikai Chen, Haitao Mao, Hang Li, Wei Jin, Hongzhi Wen, Xiaochi Wei, Shuaiqiang Wang, Dawei Yin, Wenqi Fan, Hui Liu, et al. 2024. Exploring the potential of large language models (llms) in learning on graphs. *ACM SIGKDD Explorations Newsletter*, 25(2):42–61.

Kewei Cheng, Nesreen K Ahmed, Theodore Willke, and Yizhou Sun. 2024. Structure guided prompt: Instructing large language model in multi-step reasoning by exploring graph structure of the text. *arXiv preprint arXiv:2402.13415*.

Xinnan Dai, Qihao Wen, Yifei Shen, Hongzhi Wen, Dongsheng Li, Jiliang Tang, and Caihua Shan. 2024. Revisiting the graph reasoning ability of large language models: Case studies in translation, connectivity and shortest path. *arXiv preprint arXiv:2408.09529*.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Haoyu Han, Yu Wang, Harry Shomer, Kai Guo, Jiayuan Ding, Yongjia Lei, Mahantesh Halappanavar, Ryan A Rossi, Subhabrata Mukherjee, Xianfeng Tang, et al. 2024. Retrieval-augmented generation with graphs (graphrag). *arXiv preprint arXiv:2501.00309*.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *arXiv preprint arXiv:2402.07630*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*.

Jie Huang and Kevin Chen-Chuan Chang. 2022. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798*.

Bowen Jin, Chulin Xie, Jiawei Zhang, Kashob Kumar Roy, Yu Zhang, Suhang Wang, Yu Meng, and Jiawei Han. 2024. Graph chain-of-thought: Augmenting large language models by reasoning on graphs. *arXiv preprint arXiv:2404.07103*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in* *neural information processing systems*, 35:22199–22213.

Andrew K Lampinen, Ishita Dasgupta, Stephanie CY Chan, Kory Matthewson, Michael Henry Tessler, Antonia Creswell, James L McClelland, Jane X Wang, and Felix Hill. 2022. Can language models learn from explanations in context? *arXiv preprint arXiv:2204.02329*.

Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Complex knowledge base question answering: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 35(11):11196–11215.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124*.

Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. 2023. Reasoning on graphs: Faithful and interpretable large language model reasoning. *arXiv preprint arXiv:2310.01061*.

Yao Ma and Jiliang Tang. 2021. *Deep learning on graphs*. Cambridge University Press.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

Marianna Nezhurina, Lucia Cipolina-Kun, Mehdi Cherti, and Jenia Jitsev. 2024. Alice in wonderland: Simple tasks showing complete reasoning breakdown in state-of-the-art large language models. *arXiv preprint arXiv:2406.02061*.

Jinyoung Park, Ameen Patel, Omar Zia Khan, Hyunwoo J Kim, and Joo-Kyung Kim. 2024. Graph elicitation for guiding multi-step reasoning in large language models. *arXiv preprint arXiv:2311.09762v2*.

Xiao Pu, Mingqi Gao, and Xiaojun Wan. 2023. Summarization is (almost) dead. *arXiv preprint arXiv:2309.09558*.

Shuofei Qiao, Yixin Ou, Ningyu Zhang, Xiang Chen, Yunzhi Yao, Shumin Deng, Chuanqi Tan, Fei Huang, and Huajun Chen. 2022. Reasoning with language model prompting: A survey. *arXiv preprint arXiv:2212.09597*.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. *arXiv preprint arXiv:1906.02361*.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pages 31210–31227. PMLR.

Koustuv Sinha, Shagun Sodhani, William L Hamilton, and Joelle Pineau. 2018. Compositional language understanding with text-based relational reasoning. *arXiv preprint arXiv:1811.02959*.

Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. 2024. On the self-verification limitations of large language models on reasoning and planning tasks. *arXiv preprint arXiv:2402.08115*.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Heung-Yeung Shum, and Jian Guo. 2023. Think-on-graph: Deep and responsible reasoning of large language model with knowledge graph. *arXiv preprint arXiv:2307.07697*.

Yijun Tian, Huan Song, Zichen Wang, Haozhu Wang, Ziqing Hu, Fang Wang, Nitesh V Chawla, and Panpan Xu. 2024. Graph neural prompting with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19080–19088.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Siyuan Wang, Zhongkun Liu, Wanjun Zhong, Ming Zhou, Zhongyu Wei, Zhumin Chen, and Nan Duan. 2022a. From lsat: The progress and challenges of complex reasoning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022b. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Xuezhi Wang and Denny Zhou. 2024. Chain-of-thought reasoning without prompting. *arXiv preprint arXiv:2402.10200*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. 2022. Large language models are better reasoners with self-verification. *arXiv preprint arXiv:2212.09561*.

Sohee Yang, Elena Gribovskaya, Nora Kassner, Mor Geva, and Sebastian Riedel. 2024. Do large language models latently perform multi-hop reasoning? *arXiv preprint arXiv:2402.16837*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Bowen Zhang and Harold Soh. 2024. Extract, define, canonicalize: An llm-based framework for knowledge graph construction. *arXiv preprint arXiv:2404.03868*.

Jing Zhang, Xiaokang Zhang, Jifan Yu, Jian Tang, Jie Tang, Cuiping Li, and Hong Chen. 2022a. Subgraph retrieval enhanced model for multi-hop knowledge base question answering. *arXiv preprint arXiv:2202.13296*.

Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D Manning, and Jure Leskovec. 2022b. Greaselm: Graph reasoning enhanced language models for question answering. *arXiv preprint arXiv:2201.08860*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Lingfeng Zhong, Jia Wu, Qian Li, Hao Peng, and Xindong Wu. 2023. A comprehensive survey on automatic knowledge graph construction. *ACM Computing Surveys*, 56(4):1–62.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

Yuchen Zhuang, Yue Yu, Kuan Wang, Haotian Sun, and Chao Zhang. 2024. Toolqa: A dataset for llm question answering with external tools. *Advances in Neural Information Processing Systems*, 36.

A Appendix

A.1 Datasets

In this section, we introduce the used datasets in the logical reasoning task and multi-hop question answering task.

A.1.1 Logical Reasoning Task

For the logical reasoning task, we select 4 datasets, i.e., AIW, AIW+ (Nezhurina et al., 2024), LogiQA (Liu et al., 2020) and AR-LSAT (Wang et al., 2022a). The details of each dataset are as follows:

AIW: AIW dataset contains a set of "Alice in Wonderland Problems", which typically follow the format: "Alice has N brothers and she also has M sisters. How many sisters does Alice's brother have?". This dataset is popular to evaluate the reasoning abilities of LLMs.

AIW+: The AIW+ dataset is an extension of the AIW problem, describing a more complex family structure. It introduces additional hierarchy and distractors when depicting relational family structures, making the reasoning task more challenging. In the AIW+ problem, multiple solutions could arise if the model assumes that Alice's parents have additional children, which is also a feasible solution. To eliminate this ambiguity, we added a constraint to the problem: Alice's parents do not have any other children.

LogiQA: LogiQA is a widely used logical reasoning dataset that includes questions involving various types of reasoning, such as categorical reasoning, sufficient conditional reasoning, necessary conditional reasoning, disjunctive reasoning, and conjunctive reasoning. The dataset is divided into training, validation, and test sets. Since we do not train or fine-tune the LLMs, we selected 100 samples from the test set to evaluate different methods.

AR-LSAT: AR-LSAT is a dataset collected from the Law School Admission Test (LSAT). There are three dominant game types in LSAT: ordering games, grouping games, and assignment games. In ordering games, participants must be ordered based on given facts and rules. Grouping games involve separating participants into groups according to specific facts and rules. Assignment games require assigning characteristics to participants, such as scheduling tasks for individuals, while adhering to given rules. We use all the test data to evaluate the proposed method and baselines.

A.1.2 Multi-hop Question Answering Task

For the multi-hop question answering task, we select 4 widely used datasets, i.e., 2WikiMultihopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), HotpotQA (Yang et al., 2018) and Clutrr (Sinha et al., 2018). The details of each dataset are as follows:

2WikiMultihopQA: The 2WikiMultihopQA dataset is built from Wikipedia and Wikidata. It contains several related paragraphs and one question, with various types of multi-hop questions such as comparison, inference, compositional, and bridge-comparison questions. We randomly sampled 100 questions from the test set of 2WikiMultihopQA for the experiments.

MuSiQue: The MuSiQue dataset contains multi-hop questions via single-hop question composition. Like 2WikiMultihopQA, it includes several related paragraphs per question. The dataset features 2-hop, 3-hop, and 4-hop questions. For each hop type, we randomly sampled 100 questions. Detailed results for each hop can be found in Appendix A.3.

HotpotQA: HotpotQA is a widely used multi-hop question dataset. It provides 10 paragraphs to answer a single question. There are different difficulty levels, and the easier questions are typically solvable by LLMs. We randomly selected a subset of 100 hard bridging questions from the development set of HotpotQA.

Clutrr: The Clutrr (Compositional Language Understanding with Text-based Relational Reasoning) dataset differs from the other three multi-hop question datasets. It primarily contains a single paragraph that describes relationships between family members, and the task is to infer the relationship between two specified members. The dataset includes different path lengths between the predicted family members. For our experiments, we selected path lengths from 4 to 9, as shorter paths are generally easier for LLMs to solve. We report the overall performance in Table 4 while the detailed results for each hop can be found in Appendix A.3.

A.2 Prompts of RwG

In this section, we provide the prompt of the proposed RwG for logical reasoning and multi-hop question answering tasks. The system prompt of the proposed RwG is shown in Tabel 5.

A.2.1 Logical Reasoning

There are mainly three steps in the proposed RwG, i.e., initial graph generation, graph verification and graph generation. During the experiments, we merge the graph verification and graph generation into one prompt for convenient. The initial graph generation prompt is shown in Table 6. The graph verification and generation prompt is shown in Table 7, and the question answering prompt is shown in Table 8 .

A.2.2 Multi-hop Question Answering

The prompts for Multi-hop Question Answering are similar to those used for logical reasoning. However, during graph generation, we only extract the entities and relationships relevant to the question to prevent the graph from becoming too large. During the verification stage, we check whether the current graph contains all the necessary information to answer the question. Specifically, the graph initialization prompt is shown in Table 9, the graph verification and generation prompt is shown in Table 10, and the question answering prompt is shown in Table 11 .

A.3 More results

We provide detailed results for the different hop questions in the MuSiQue and Clutrr datasets. Specifically, the MuSiQue dataset contains 2, 3, and 4-hop questions, and the results are shown in Table 12. Additionally, the results for 4 to 9-hop questions in the Clutrr dataset are provided in Table 13. From the results, we observe that the proposed RwG outperforms all baselines across all hop levels.

A.4 Case Studies

In this section, we present additional cases generated by GPT-4o. Due to the lengthy context generated by GPT-4o, we have removed some non-essential content.

A.4.1 Failure case of AIW+ problem with Chain-of-Thought prompting

In section 4.1.4, we analyzed how the proposed RwG solves the AIW+ example. Here, we present a failure case using GPT-4o with Chain-of-Thought prompting.

Question: Alice has 3 sisters. Her mother has 1 sister who does not have children - she has 7 nephews and nieces and also 2 brothers. Alice's father has a brother who has 5 nephews and nieces in total, and who has also 1 son. Alice's parents have no other children. How many cousins does Alice's sister have? Please think step by step and explain your reasoning.

Response of GPT-4o with CoT:
To solve this problem, we need to determine the number of cousins Alice and her sisters have. We'll approach this step by step:

1. Alice's Family Structure:
- Alice has 3 sisters.
- Alice's parents have no other children, so Alice and her 3 sisters are the only children in their immediate family.

2. Maternal Side:
- Alice's mother has 1 sister and 2 brothers.
- The mother's sister has no children.
- The mother's 2 brothers collectively have 7 nephews and nieces.
- Since Alice and her 3 sisters (4 children in total) are part of these 7 nephews and nieces, the remaining 3 must be Alice's cousins from her maternal uncles.

3. Paternal Side:
- Alice's father has 1 brother.
- This brother has 5 nephews and nieces in total.
- Alice and her 3 sisters are 4 of these nephews and nieces.
- Therefore, there must be 1 more nephew or niece, who is Alice's cousin from her paternal uncle.
- This paternal uncle also has 1 son, who is Alice's cousin.

4. Total Cousins:
- Alice has 3 cousins from her maternal side.
- Alice has 1 cousin from her paternal side.
- Therefore, Alice has a total of 3 + 1 = 4 cousins.

From the response, we can see that GPT-4o fails to infer one cousin from the paternal side, highlighting the effectiveness of the proposed RwG.

A.4.2 Case from LogiQA dataset with RwG

Context: There are nine cards, one to nine. A, B, C, D four people take cards, each take 2. It is known that the sum of the two cards taken by A is 10; the difference between the two cards taken by B is 1; the product of the two cards taken by C is 24; and the quotient of the two cards taken by D is 3.

Question: Which of the following is the left

Table 5: System prompt of RwG.

You are an expert of knowledge graphs. Knowledge Graphs aim to represent the relationships between entities. You are good at reasoning based on the graph. When you are asked to output the graph, please write [latest graph] first, followed by all triples, such as (entity1, relation, entity2) in the graph.

Table 6: Initial graph generation prompt for logical reasoning task.

Please build a knowledge graph based on the given context: {Context} and question: {Question}. The graph aim to answer the question. The nodes represent entities while edges represent their relations.

Table 7: Graph verification and generation prompt for logical reasoning task.

Does the latest graph meet all the requirements? Please first define each relationships in the context. And then carefully verify all the requirements. If the old graph meets all the requirements, please write [YES] at the end. If the old graph is wrong, please update the graph by inferring missing relations and nodes -> Write the [latest graph] with new edge list first, followed by [No].

Table 8: Answer generation prompt for logical reasoning task.

Please answer the following question based on the latest graph and context: {Question}.

Table 9: Initial graph generation prompt for multi-hop question answering task.

2WikiMultihopQA, MuSiQue, HotpotQA	There are multiple paragraphs in the given context: {Context}. Please first find all paragraphs that may related to the question: {Question}. Please extract all the entities and relations of these paragraphs. Then build a knowledge graph based on these entities and relations.
Clutrr	Please build a family relation knowledge graph based on the context sentence by sentence. Nodes represent roles, and edges represent relationships. The graph should be bidirectional, including ([entity1], relation, [entity2]) and ([entity2], reverse relation, [entity1]).

Table 10: Graph verification and generation prompt for multi-hop question answering task.

2WikiMultihopQA, MuSiQue, HotpotQA	Does the graph include all the entities and relations related to the questions: {Question}? Please recursively add new entities and relations after you have new entities. If the old graph meets the requirement, please write [YES] at the end. If the old graph can not, please update the graph by retrieving more entities and relations from the given contexts. Add these information to form a new graph. -> Write the [latest graph] with new edge list first, followed by [No].
Clutrr	Can the [latest graph] contains enough information to answer the question: {Question}? Please confirm your conclusion. If yes, please write [YES] at the end. If not, update the graph by inferring missing relations between entities as many as possible based on the graph to form a new graph. Then, provide the [latest graph], followed by [No]. Please think step by step and explain your reasoning.

Table 11: Answer generation prompt for multi-hop question answering task.

Please answer the following question: {Question} based on the latest graph and context: {Context}.

Table 12: The results of different hop questions in MuSiQue dataset

Hops	MuSiQue 2		MuSiQue 3		MuSiQue 4	
Methods	Claude	GPT-4o	Claude	GPT-4o	Claude	GPT-4o
Vanilla	0.6011	0.7302	0.4937	0.5608	0.4076	0.5484
Cot	0.6472	0.7091	0.4967	0.5893	0.5039	0.5208
ToT	0.6944	0.7437	0.5770	0.6342	0.5171	0.5577
GoT	0.6495	0.7494	0.5365	0.6248	0.4667	0.5876
Least-to-Most	0.6900	0.7340	0.5222	0.6288	0.5275	0.5365
Structure Prompting	0.6325	0.7216	0.5572	0.5952	0.4886	0.5115
RwG	**0.7488**	**0.8126**	**0.6346**	**0.7032**	**0.5353**	**0.6403**

Table 13: The results of different hop questions in Clutrr dataset

Hops	Clutrr 4		Clutrr 5		Clutrr 6		Clutrr 7		Clutrr 8		Clutrr 9	
Methods	Claude	GPT-4o	Claude	GPT-4o	Claude	GPT-4o	Claude	GPT-4o	Claude	GPT-4o	Claude	GPT-4o
Vanilla	0.3736	0.5263	0.2816	0.6091	0.2336	0.5981	0.2500	0.5902	0.1000	0.4800	0.2016	0.4873
Cot	0.5105	0.7894	0.4022	0.6839	0.4299	0.6915	0.3333	0.625	0.2666	0.5466	0.2352	0.5714
Least-to-Most	0.3947	0.7894	0.3965	0.7068	0.3457	0.7102	0.2986	0.5694	0.2800	0.5400	0.2796	0.5294
Structure Prompting	0.4736	0.7421	0.3563	0.7183	0.4112	0.6635	0.3958	0.6041	0.3466	0.5466	0.2857	0.5126
RwG	**0.5894**	**0.8315**	**0.4655**	**0.7471**	**0.4953**	**0.7102**	**0.4236**	**0.6597**	**0.3733**	**0.5533**	**0.3361**	**0.5798**

card? 0. Six.; 1. Three.; 2. Seven.; 3. Four.

Initial Graph Generation The possible cards assignments:

A: (1, 9), (2, 8), (3, 7), (4, 6)

B: (2, 3), (3, 4), (4, 5), (5, 6), (6, 7), (7, 8), (8, 9)

C: (3, 8), (4, 6)

D: (3, 1), (6, 2), (9, 3)

Initial Graph:
- A - 1 ; A - 9
- B - 8 ; B - 7
- C - 4 ; C - 6
- D - 3 ; D - 9

First Round Verification and generation

There is overlap using card 9. New graph:
- A - 1 ; A - 9
- B - 2 ; B - 3
- C - 4 ; C - 6
- D - 6 ; D - 2

Second Round Verification and generation

Still found overlap. New graph:
- A - 1 ; A - 9
- B - 4 ; B - 5
- C - 3 ; C - 8
- D - 6 ; D - 2

Third Round Verification and generation

Pass the verification.

Answer: The left card is 7.

A.4.3 Case from AR-LSAT dataset with RwG

Context: A maintenance company that takes service requests from three clients—Image, Solide, and Truvest—plans to set targets for its average service response times. Service targets will be set at 3 days, 2 days, or 1 day. Two service targets are set for each client—one for requests received through the maintenance company's website and one for requests received by voicemail. The six targets are set according to the following conditions: None of the clients can have a website target that is longer than its voicemail target. Image's voicemail target must be shorter than the other clients' voicemail targets. Solide's website target must be shorter than Truvest's website target.

Initial Graph Generation

1. Identify the variables:
- There are six targets:
- Image's website target (IW) and voicemail target (IV).
- Solide's website target (SW) and voicemail target (SV).
- Truvest's website target (TW) and voicemail target (TV).

The generated graph:
- IW ≤ IV
- IV < SV
- IV < TV
- SW < TW
- TW < TV
- SW < TV

Question: If Truvest's website target is shorter than its voicemail target, which one of the following must be true? 0. Image's voicemail target is 2 days.; 1. Image's website target is 2 days.; 2. Image's website target is 1 day.; 3. Solide's website target is 2 days.; 4. Solide's website target is 1 day.
Answer:
- Option 0: Image's voicemail target is 2 days.
This option doesn't directly relate to the constraints provided. The constraints don't specify a numeric value for IV, just that IV < SV and IV < TV. Therefore, this option is not necessarily true based on the graph.

- Option 1: Image's website target is 2 days.
Similar to Option 0, there is no specific constraint that pins IW to a numeric value. The constraint IW $\leq$ IV doesn't specify that IW must be 2 days. Hence, this option is not necessarily true. - Option 2: Image's website target is 1 day.
If IW is 1 day, it satisfies IW $\leq$ IV. However, there's no direct constraint requiring IW to be exactly 1 day. Therefore, this option is not necessarily true.

- Option 3: Solide's website target is 2 days.
For SW to be 2 days, while it satisfies SW < TW, the constraints don't dictate that SW must be exactly 2 days. No direct necessity is derived from the constraints for SW to be 2 days. Thus, this option is not necessarily true.

- Option 4: Solide's website target is 1 day.
Given SW < TW and SW < TV, if SW is set to 1 day, it is possible to meet the condition of SW being the shortest among SW, TW, and TV, considering that TW > SW and TV > SW because of the constraints. Setting SW to 1 day ensures that it is less than TW and TV, which aligns with the necessary condition SW < TW and SW < TV. This option aligns well with the constraints and scenario.

A.4.4 Case from MuSiQue dataset with RwG

We select an example from the MuSiQue dataset, where GPT-4o provided incorrect answers. In this example, 19 paragraphs are given, and the question is: *When was the death penalty abolished in the country which, along with Eisenhower's VP's country, recognized Gaddafi's government early on?* The graph constructed by GPT-4o is in Table 14. This graph provides a condensed representation of the entire context, containing essential reasoning structure to answer the question. By focusing on this explicit knowledge graph, which includes only the necessary information, LLMs can avoid irrelevant details that might otherwise interfere with their response generation (Shi et al., 2023).

Table 14: The illustration of Case in MuSiQue dataset

(Dwight D. Eisenhower) –[was President of]–> (United States)
(Richard Nixon) –[was Vice President of]–> (United States)
(United States) –[recognized government of]–> (Muammar Gaddafi)
(United Kingdom) –[recognized government of]–> (Muammar Gaddafi)
(Capital Punishment) –[abolished in]–> (Great Britain) [in 1965]
(Capital Punishment) –[abolished in]–> (Northern Ireland) [in 1973]
(Capital Punishment) –[abolished in]–> (United Kingdom) [in 1998]

A.4.5 Case from 2WikiMultihopQA dataset with RwG

In the 2WikiMultihopQA dataset, most questions require only 2 hops, making it easier to extract the relevant entities in a single step.

Question: Which film has the director born earlier, Haiducii (Film) or My Wife'S Best Friend?

Initial Graph Generation
- (Haiducii, directed_by, Dinu Cocea)
- (My Wife's Best Friend, directed_by, Richard Sale)
- (Dinu Cocea, birthdate, 22 September 1929)
- (Richard Sale, birthdate, December 17, 1911)

Answer: My Wife's Best Friend

A.4.6 Case from MuSiQue dataset with RwG

In the MuSiQue dataset, we selected a 4-hop question answering example for this case study.

Question: Based on population alone, what is the ranking of the city having the Courthouse of the birthplace of James Glisson in the country winning the rugby 7 in Las Vegas?

Initial Graph Generation:
- (James Glisson) –born in–> (Jackson County, Florida)
- (Jacksonville, Florida) –located in–> (Jackson County, Florida)
- (Bermuda) –participated in–> (Las Vegas 7's)
- (Las Vegas, Nevada) –hosted–> (Las Vegas 7's)
- (Jacksonville, Florida) –is ranked as most populous in–> (Florida)

First Round Verification and Generation
- (James Glisson) –born in–> (Jackson County, Florida)
- (Jacksonville, Florida) –located in–> (Jackson County, Florida)
- (Bermuda) –won–> (Las Vegas 7's)
- (Las Vegas, Nevada) –hosted–> (Las Vegas 7's)
- (Jacksonville, Florida) –is ranked 12th most populous city in–> (United States)

Answer: 12th

Table 15: Average number of iterations for different datasets.

	AIW+	LogiQA	LSAT	MuSiQue2	MuSiQue3	MuSiQue4
GPT-4o	3.29	1.53	1.65	1.43	1.81	1.97
Claude	2.01	1.37	1.42	2.19	2.56	2.75

Table 16: The peformance comparision between RwG and RwG-1 on AIW+ and LogiQA dataset.

	AIW+		LogiQA	
	GPT-4o	Llama3.1-70B	GPT-4o	Llama3.1-70B
RwG - 1	0.2352	0.1176	0.5925	0.5606
RwG	0.5294	0.4545	0.6344	0.5913

A.5 The Computational Complexity of RwG

Compared to vanilla LLMs, RwG does require additional resources due to the iterative process of graph construction. To assess the extent of resource consumption, we calculated the average number of iterations required to generate the graph across different datasets as shown in Table 15. From the results, we observe that the average number of iterations required to generate the graph is relatively low, indicating that the resource consumption of RwG is manageable. Interestingly, for the MuSiQue dataset, we note that as the reasoning complexity (number of hops) increases, the proposed RwG requires slightly more steps to generate the final graph.

It is worth noting that other baselines, such as self-consistency, also require multiple generations. Similarly, methods like GoT and ToT involve generating additional thoughts for reasoning, which can result in an exponential increase in LLM calls as the depth of reasoning increases.

A.6 The Effectiveness of Verification and Generation Process in RwG

To demonstrate the effectiveness of verification and generation Process in RwG, we evaluated the performance of the graph generated one step before it fully meets all requirements on the AIW+ and LogiQA datasets. For cases requiring only one iteration to generate the final graph, the final graph is used. We denote this method as RwG-1. The peformance comparision between RwG and RwG-1 on AIW+ and LogiQA dataset is shown in Table 16. From the results, we observe that the performance of RwG-1, which uses the last but one graph, is significantly worse than that of RwG, which uses the final graph that passes the verification.

A.7 Results on Time-Sensitive-QA dataset

To further evaluate the versatility of the proposed RwG method, we added a temporal reasoning dataset, Time-Sensitive-QA (Chen et al., 2021), which involves reasoning about fact evolution over time. The performance comparison between the proposed RwG with baselines are shown in Table 17. The results further demonstrate the effectiveness of the proposed RwG method.

Table 17: The performance comparison on Time-Sensitive-QA dataset

Method	GPT-4o	Llama3.1-70B
Vanilla	0.7161	0.5277
CoT	0.7541	0.7179
ToT	0.7483	0.5542
GoT	0.7301	0.6524
RwG	0.7862	0.7712