M^{3}GQA: A Multi-Entity Multi-Hop Multi-Setting Graph Question Answering Benchmark

Boci Peng[1] **Yongchao Liu**[2]* **Xiaohe Bo**[3] **Jiaxin Guo**[1]
Yun Zhu[4] **Xuanbo Fan**[1] **Chuntao Hong**[2] **Yan Zhang**[1]*
[1]School of Intelligence Science and Technology, Peking University
[2]Ant Group
[3]Gaoling School of Artificial Intelligence, Renmin University of China
[4]College of Computer Science and Technology, Zhejiang University
{bcpeng,guojiaxin,xuanbo.fan}@stu.pku.edu.cn, {yongchao.ly,chuntao.hct}@antgroup.com,
xiaohe@ruc.edu.cn, zhuyun_dcd@zju.edu.cn, zhyzhy001@pku.edu.cn

Abstract

Recently, GraphRAG systems have achieved remarkable progress in enhancing the performance and reliability of large language models (LLMs). However, most previous benchmarks are template-based and primarily focus on few-entity queries, which are monotypic and simplistic, failing to offer comprehensive and robust assessments. Besides, the lack of ground-truth reasoning paths also hinders the assessments of different components in GraphRAG systems. To address these limitations, we propose M^{3}GQA, a complex, diverse, and high-quality GraphRAG benchmark focusing on multi-entity queries, with six distinct settings for comprehensive evaluation. In order to construct diverse data with semantically correct ground-truth reasoning paths, we introduce a novel reasoning-driven four-step data construction method, including tree sampling, reasoning path backtracking, query creation, and multi-stage refinement and filtering. Extensive experiments demonstrate that M^{3}GQA effectively reflects the capabilities of GraphRAG methods, offering valuable insights into the model performance and reliability. By pushing the boundaries of current methods, M^{3}GQA establishes a comprehensive, robust, and reliable benchmark for advancing GraphRAG research. Our code and dataset are available at https://github.com/pengboci/M3GQA.

1 Introduction

In recent years, Retrieval-Augmented Generation (RAG) (Fan et al., 2024) has demonstrated significant potential in enhancing the performance and mitigating hallucinations in LLMs by retrieving relevant external knowledge. However, traditional RAG methods often struggle to answer global-level or complex questions effectively (Edge et al., 2024), as they fail to capture essential relational and contextual knowledge. To address these limitations,

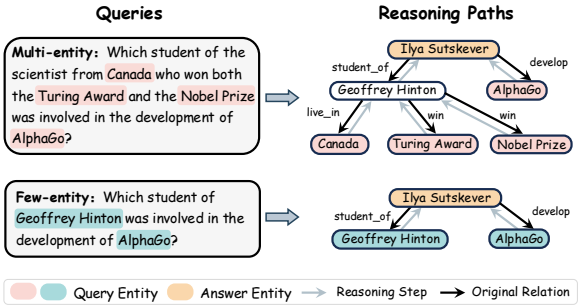


Figure 1: Comparison of examples between multi-entity queries and few-entity queries.

GraphRAG methods (Edge et al., 2024; Guo et al., 2024b; Li et al., 2024; Peng et al., 2024b) are proposed to enhance LLMs by incorporating relational and contextual knowledge between key entities, integrating relevant structural information to improve response accuracy and comprehension.

As the field of GraphRAG continues to advance rapidly, effectively evaluating the capabilities of GraphRAG methods has become essential for driving further research. However, previous benchmarks, as listed in Table 1, are monotypic and simplistic, failing to offer comprehensive and robust assessments: (1) They lack complex, multi-entity queries, which are common in real-world scenarios and challenging to models. Although CWQ (Talmor and Berant, 2018), GrailQA (Gu et al., 2021), and LC-QUAD 2.0 (Dubey et al., 2019) include few multi-entity queries, they mainly focus on cases with only three entities, and the quantity is quite limited. (2) The reliance on templates or manually annotated QA pairs in these benchmarks leads to high labor costs and restricted query formats, which may cause the model overfitting to specific query structures, limiting the models' robustness and adaptability. (3) These benchmarks lack ground-truth reasoning paths, which impedes a comprehensive evaluation of various components in GraphRAG systems. While earlier

*Corresponding authors

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 30594–30620
July 27 - August 1, 2025

	Multi Hop	Multi Entity	Multi Setting	Non-Template or Automic	Reasoning Paths
WebQ (Berant et al., 2013)	✓	✗	✗	✗	✗
SimpleQ (Bordes et al., 2015)	✗	✗	✗	✗	✗
WebQSP (Yih et al., 2016)	✓	✗	✓	✗	✗
MetaQA (Zhang et al., 2018)	✓	✗	✗	✗	✗
GrailQA (Gu et al., 2021)	✓	✓✗	✓	✗	✗
LC-QUAD2.0 (Dubey et al., 2019)	✓	✓✗	✓	✗	✗
QALD10-en (Perevalov et al., 2022)	✓	✗	✗	✗	✗
CWQ (Talmor and Berant, 2018)	✓	✓✗	✓	✗	✗
TextGraphs2024 (Sakhovskiy et al., 2024)	✓	✗	✗	✗	✓✗
M^3GQA (ours)	✓	✓	✓	✓	✓

Table 1: Comparisons between M^3GQA and previous datasets. In "Multi Entity" column, "✓✗" indicates the data contains less than 10% multi-entity queries or only contains queries with no more than three entities. "Reasoning Paths" indicates whether the dataset provides the ground-truth reasoning paths, and "✓✗" denotes that using heuristic rules to generate reasoning paths.

studies (Sakhovskiy et al., 2024) use shortest paths between query and answer entities as the ground-truth, these approaches have inherent limitations: First, reasoning paths for a given query may not necessarily be the shortest. Second, multiple shortest paths with different semantics may exist, making it difficult to ensure the semantic correctness.

To solve these challenges, we propose **M^3GQA**, a complex, diverse, and high-quality GraphRAG benchmark, with six distinct settings to comprehensively evaluate GraphRAG systems. Unlike previous **query-first** data construction methods, we propose a novel **reasoning-first** approach. Here, we first sample two sets of nodes with certain relations, serving as query and answer entities. Then we construct the ground-truth reasoning paths with a backtracking algorithm. After that, an LLM is utilized to generate corresponding queries based on previous information. Due to the flexibility of sampling, this method generates more complex and diverse QA data, while ensuring the semantic correctness of reasoning paths. Moreover, our method is independent of specific graph schemas or expert knowledge, making it easily adaptable to new domains. During the query generation process, since the factual information, such as the reasoning path, query entity, and answer entity, is derived directly from knowledge graphs, issues such as hallucination and bias in LLMs are effectively mitigated. Nevertheless, to further improve the data quality, we introduce a multi-stage refinement and filtering process, and manually curate the validation and test sets. In this way, M^3GQA offers a comprehensive and reliable evaluation for GraphRAG systems.

Extensive experimental results across six settings, focusing on two key aspects: the effectiveness and efficiency of graph retrieval, and the quality of answer generation, demonstrate that our benchmark accurately reflects the capabilities of GraphRAG methods, providing valuable insights into the model performance and reliability, thereby advancing the GraphRAG research.

Our contributions can be summarized as follows:

- To the best of our knowledge, M^3GQA is the first GraphRAG benchmark focusing on multi-entity queries, a highly practical yet challenging aspect in GraphRAG systems.
- To automatically generate complex, diverse, high-quality multi-entity query data with semantically correct reasoning paths, we propose an innovative four-step reasoning-first data construction method. The method is also easily adaptable to other domains.
- Extensive experiments show that M^3GQA accurately reflects the capabilities of GraphRAG methods, providing valuable insights into the model performance and reliability, thereby advancing GraphRAG research.

2 Related Work

GraphRAG Methods. GraphRAG is an emerging technique that enhances Vallina RAG by incorporating graph data, with three main steps: graph-based indexing, graph-guided retrieval, and graph-enhanced generation (Peng et al., 2024b). The indexing phase focuses on constructing and indexing knowledge graphs from text corpus (Edge et al., 2024; Guo et al., 2024b). This paper primarily focuses on the latter two stages, i.e. the retrieval of relevant graph information and the answer generation. Early methods for retrieval and generation mainly rely on GNNs (Lin et al., 2019; Yasunaga et al., 2021; Dong et al., 2023; Taunk et al., 2023) or small language models (Sun et al., 2019; Zhang et al., 2022; Jiang et al., 2023b; Huang et al., 2023; Li et al., 2023; Peng et al., 2024a) (e.g., BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019)). Recently, due to the outstanding reasoning and text generation capabilities of LLMs (OpenAI, 2024; Zhao et al., 2024), various LLM-based GraphRAG methods have been proposed. Some methods (Jiang et al., 2023a; Sun et al., 2024; Luo et al., 2024; Jin et al., 2024; Jiang et al., 2024a) utilize LLMs for both graph retrieval and

answer generation, while others leverage graph algorithms (Salnikov et al., 2023; He et al., 2024; Wang et al., 2024; Jiang et al., 2024b; Wen et al., 2024) or smaller models (Wu et al., 2023; Guo et al., 2024a; Hu et al., 2024; Mavromatis and Karypis, 2024) as graph retrievers, considering the time costs and expenses. Previous methods have achieved remarkable success on traditional datasets and a more challenging benchmark is needed for further development of GraphRAG.

GraphRAG Datasets. Various benchmarks (Berant et al., 2013; Bordes et al., 2015; Yih et al., 2016; Mihaylov et al., 2018; Talmor and Berant, 2018; Zhang et al., 2018; Talmor et al., 2019; Dubey et al., 2019; Gu et al., 2021; Perevalov et al., 2022; Cao et al., 2022; Sen et al., 2022; Mallen et al., 2023; Sakhovskiy et al., 2024) have been introduced to evaluate the performance of GraphRAG systems. Table 1 lists several widely used datasets along with their features, showing that current datasets mainly focus on few-entity queries. Although CWQ (Talmor and Berant, 2018), GrailQA (Gu et al., 2021), and LC-QUAD 2.0 (Dubey et al., 2019) contain several multi-entity queries, the number is relatively small and primarily concentrated in the case of three entities. Furthermore, previous datasets are created based on templates or manually annotated QA pairs, which may cause restricted query formats and model overfit. Also, reasoning paths corresponding to the QA pairs are not available in previous datasets. Therefore, we propose a novel reasoning-first data construction method and create a multi-entity multi-hop multi-setting benchmark named M^{3}GQA.

3 Task Formulation

Graph Question Answering (GQA) requires models to answer natural language queries according to a given graph, which is an important application of GraphRAG. The graph can be formally defined as $G = (V, E, \{x_v\}, \{x_e\})$, where V and E are node and edge sets, $\{x_v\}$ and $\{x_e\}$ represent their corresponding test attributes. In this paper, for the convenience of data construction, we use Knowledge Graphs to provide structured information, where attributes of nodes and edges are their names. Given a query q and a graph G, models need to first locate the nodes $V_q \subset V$ in the query (query entities), and then determine a node or a node set $V_a \subset V$ (answer entities) in the graph G as the answer.

In M^{3}GQA, we focus primarily on multi-entity queries, which are defined here as queries that involve at least three nodes from the graph, i.e. $|V_q| \geq 3$. We first introduce 4 general settings with different complexities, which are as follows:

Single-hop Setting: Queries in the single-hop setting can be answered with a single reasoning step. We ensure there is only one answer node, meaning that in the reasoning path, the answer node v_a is at a distance of 1 from all query nodes. Therefore, the reasoning path can be shaped like: $\{v_q \xrightarrow{e} v_a | v_q \in V_q\}$. This setting is theoretically the simplest, designed to assess the model's basic ability to handle multi-entity queries.

Multi-hop Setting: Queries in this setting require starting from multiple query entities and reaching a single answer node using multi-hop reasoning. The number of reasoning steps from different query entities may vary, so the reasoning path can be defined as: $\{v_q \xrightarrow{e_1} v_m \xrightarrow{e_2} \cdots \xrightarrow{e_k} v_a | v_q \in V_q\}$, where $k > 1$ represents the hop count, which may vary with different query entities. The answering process can be intuitively understood as starting from the child nodes at different levels of a tree (query entities) and searching for the root node (answer entity). Due to the highly variable number and distribution of query entities, the reasoning process can be significantly more complex than few-entity multi-hop questions.

Set Setting: The main difference between the set setting and the two settings mentioned above is that the answer to each query is a set of nodes, that is $|V_a| > 1$. The primary focus of the setting is to assess the model's ability to obtain a complete answer without any omissions.

Aggregation Setting: Building on the set setting, we introduce additional constraints (e.g., maximum value, minimum value, etc.) to the set of answer nodes, so that the final answer is a single node from the node set satisfying the given conditions. In this setting, the model first needs to identify the original answer set, and then, by incorporating information from neighbor nodes, compare and reason across multiple similar entities. Therefore, the reasoning path in the aggregation setting can be denoted as: $\{v_q \xrightarrow{e_1} v_m \xrightarrow{e_2} \cdots \xrightarrow{e_k} v_a \xleftarrow{e} v_n \, | v_q \in V_q, v_n \in V_n\}$, where $k \geq 1$ and V_n is a subset of v_a's neighbor nodes.

Beyond the general settings above, we innovatively propose other two special settings:

Editing Setting: Considering that many queries can be answered by LLMs solely based on their parameterized knowledge, we introduce the editing

setting. Specifically, based on the single-hop setting and multi-hop setting, we modify the answer node v_a in the original graph, replacing it with a new node v'_a generated by LLMs that have different textual attributes. The model needs to respond with the new answer node v'_a, rather than the original one. The performance under this setting highly depends on the retrieval quality. It also enables us to investigate the potential applications of GraphRAG in the context of knowledge updating for LLMs.

Answerability Setting: In the answerability setting, we first construct the data following single and multi-hop settings, and then randomly select a subset of the data, remove the answer nodes and their neighbors from G to create a new graph G'. Given a query q and graph G', the model is required to determine whether G' contains the answer to the query. This setting accesses the model's ability to perform graph retrieval and evaluate its sensitivity to the information in the retrieved subgraphs.

4 Data Construction

In this section, we will introduce our reasoning-first data construction method. Our ultimate goal is to obtain diverse queries with multiple entities, their corresponding answers, and reasoning paths. To achieve this, we first obtain the graphical representation of the query, i.e., reasoning paths, and then map it to the textual representation, i.e., QA pairs of natural language. Specifically, we first propose a tree sampling algorithm to sample two sets of nodes with certain relations, serving as query and answer nodes respectively. Since directly generating queries from query and answer nodes is challenging, we then utilize a backtracking algorithm to obtain the original reasoning paths. Afterward, an LLM is leveraged to create queries based on query nodes, answer nodes, and corresponding reasoning paths. Finally, we refine reasoning paths and filter low-quality data with a multi-stage strategy. Figure 2 shows the overview of our method.

4.1 Tree Sampling

To sample query nodes and answer nodes[1] with certain relationships, we propose a novel tree sampling method. Specifically, we randomly select a node $v_a \in V$ as the starting point and perform a constrained breadth-first search (C-BFS), where both the search width and depth are limited. All search paths form a tree with the selected node v_a as the root node, which then serves as the answer node. The process can be formulated as:

$$G_{\mathcal{T}_{v_a}} = \text{C-BFS}(v_a, G, D_{\text{max}}, W_{\text{max}}), \quad (1)$$

where D_{max} is the maximum search depth and W_{max} is the maximum search width. $G_{\mathcal{T}_{v_a}} = (V_{\mathcal{T}_{v_a}}, E_{\mathcal{T}_{v_a}})$ denotes the search tree with v_a as the root node, which is a subgraph of G.

Next, we randomly sample n ($n \geq 3$) nodes from the non-root nodes of the tree, serving as the query nodes. Afterwards, we can obtain multiple (v_a, V_q) pairs, where $V_q = \{v_{q_i}\}_{i=1}^{n}$ denotes the query node set. This process can be further formulated as $\{(v_a, V_q) | v_a \in V, V_q \subset V_{\mathcal{T}_{v_a}} \backslash \{v_a\}\}$. Considering that each setting in M^{3}GQA has its unique characteristics and constraints, we then sequentially explain how to adapt the tree sampling algorithm to these different settings. A detailed introduction with strict definitions and mathematical notation can be found in the Appendix D.

Single-hop Setting: The main characteristic of the single-hop setting is that the hop count is single, that is $D_{\text{max}} = 1$. Besides, to ensure the uniqueness of the answer node, we examine all other common neighboring nodes of query nodes to ensure that no other node is linked to query nodes with the same relations.

Multi-hop Setting: In the multi-hop setting, the search depth must be at least 2, i.e. $D_{\text{max}} > 1$. Additionally, to ensure the answer's uniqueness, we examine the first level ancestor nodes (the root node at level 0) of each query node, using the same detection method as the single-hop setting.

Set Setting: The data construction method in the set setting is more complex. Since the answer is a node set, we need to consider the initialization of the answer nodes and the process for conducting the search. For answer node selection, we first adopt the same searching approach as the single-hop setting. Here, we additionally restrict that the relation remains consistent during the search process. Then we treat the query node in the single-hop setting as the answer nodes in the set setting, ensuring that answer nodes share some similar properties. As for the search process, a straightforward approach is to use the answer nodes as the starting points and apply multi-source BFS. However, this may result in a disconnected search tree, which makes it hard to form a query. To address this issue, we first detect all the shared

[1]For simplicity, we first consider the case where there is only a single answer node.

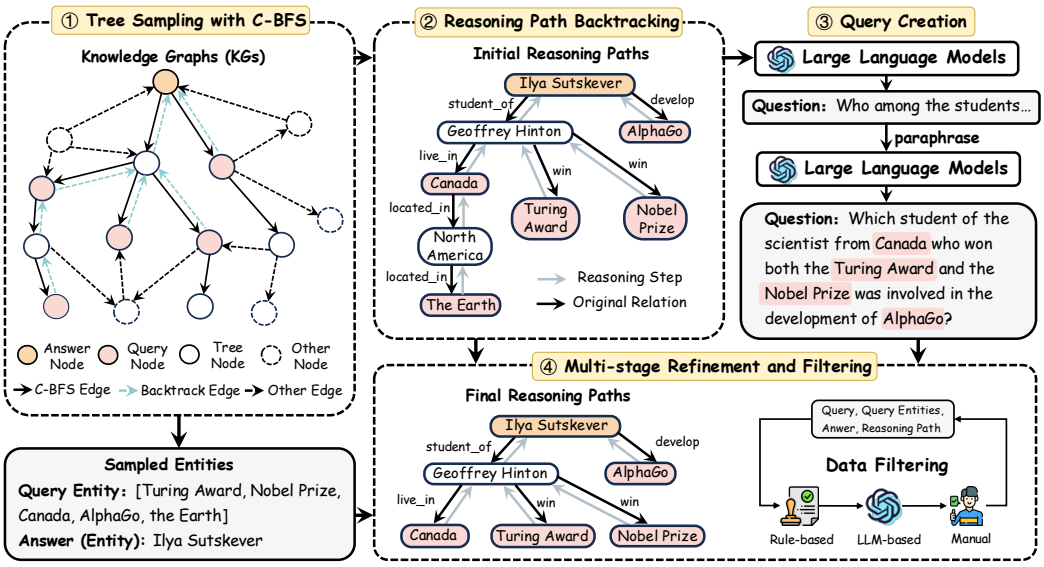


Figure 2: The overview of our four-step data construction method: tree sampling, reasoning path backtracking, query creation, and multi-stage refinement and filtering.

neighboring nodes of the answer nodes and then start the multi-source BFS from these nodes.

Aggregation Setting: Based on the set setting, the aggregation setting imposes further constraints on answer nodes. Specifically, we examine all common neighbors of answer nodes in the reverse direction of the BFS process and construct the data using two strategies. The first strategy is to identify a common relation shared by all answer nodes. Then neighboring nodes connected by this relation are included in the search tree for subsequent processes. The second strategy is to obtain the relation that is unique to one of the answer nodes. In this case, we add the edge to the search tree and treat that special answer node as the final answer.

Editing Setting: To construct data in the editing setting, we first modify the graph using the approach mentioned in Section 3 and then apply the tree sampling algorithm using the same method as the single-hop and multi-hop settings.

Answerability Setting: In contrast to the steps in the editing setting, here we first apply the tree sampling algorithm and then modify the graph. In this way, we can determine whether the query constructed based on the reasoning path can be answered using the given graph.

4.2 Reasoning Path Backtracking

After obtaining the query and answer nodes, we construct possible reasoning paths based on the search tree. Considering that the search tree may contain a lot of irrelevant information, we propose a backtracking method to prune it. Specifically, we take each query node v_q as the starting point and trace back along the search tree until we reach each answer node v_a. This process can be denoted as:

$$p_{v_q,v_a} = \text{BackTrack}(v_q, v_a, G_{\mathcal{T}_{v_a}}). \quad (2)$$

The resulting reasoning paths can be represented as $\mathcal{P} = \{p_{v_q,v_a} | v_q \in V_q, v_a \in V_a\}$. To provide an intuitive understanding, we include examples of reasoning paths in six settings in Appendix A.3.

4.3 Query Creation

In this section, we aim to create the corresponding queries according to query nodes V_q, answer nodes V_a, and reasoning paths $\mathcal{P}$. Thanks to the powerful text generation capabilities of LLMs, we can generate diverse, and fluent queries with carefully designed prompts. Besides, since the factual information, such as the reasoning path, query entity, and answer entity, is derived directly from the knowledge graph, issues such as hallucination and bias in LLMs are effectively mitigated during the question generation process. The query creation process can be formalized as:

$$q = \text{LLM}_c(V_q, V_a, \mathcal{P}). \quad (3)$$

Queries in majority settings of M^{3}GQA can be directly generated using Equation 3, with the only exception of the aggregation setting. This is because queries in this setting not only need to represent the relationship between the query and answer

nodes but also impose additional constraints to ensure that the final answer is one of the nodes with similar attributes, which is too complex for LLMs. Therefore, we employ a two-step method to generate queries in the aggregation setting. We first create queries whose answer is a node set, ignoring the additional constraints. Subsequently, we incorporate the constraints into prompts and call LLMs again to rewrite the queries in the previous step.

Additionally, to make created queries more natural and fluent, and better resemble the questions humans would ask in real-world scenarios, we add an additional LLM call for query refinement, which can be formulated as:

$$q = \text{LLM}_r(q, V_q). \tag{4}$$

We provide all prompts in Appendix F.

4.4 Multi-Stage Refinement and Filtering

Considering that LLMs may not incorporate all sampled query nodes when generating queries, we also need to refine the initial reasoning paths, aligning them with actual query nodes included in the final queries. Specifically, we first employ text matching to identify the included query nodes, and then apply the same backtracking algorithm outlined in Section 4.2 to refine the reasoning paths.

To further ensure the data quality, we implement diverse filtering methods, including rule-based filtering, LLM-based filtering, and manual filtering, which are carried out in sequence.

Rule-based Filtering. We design the following rules to remove low-quality data: (1) The number of nodes in each query must be no less than 3. (2) The queries must not contain answer nodes. (3) The query must be a standalone problem. (4) The reasoning paths must include the answer nodes. (5) To prevent data leakage, for each setting, we use random deduplication to avoid multiple entries sharing the same answer.

LLM-based Filtering. We employ LLMs to verify the correctness of the generated data. Specifically, we feed the query, reasoning paths, and the answer into LLMs, and ask LLMs to determine whether the answer is correct. We retain the data deemed correct by LLMs. The prompt used in this stage is shown in Appendix F.

Manual Filtering. We manually review and correct the data in validation and test sets, focusing on the clarity of query expressions, the correctness of answers, and the accuracy of reasoning paths.

Appendix B demonstrates more details of the filtering stage, including detailed guidelines for manual filtering and additional data statistics.

	Entry Count	Avg. Token	Avg. Entities	Avg. Hops	Avg. Triplets	Avg. Nodes
Single-hop	1,542	29.68	4.00	1.00	3.92	4.92
Multi-hop	1,430	39.93	3.43	2.66	5.23	6.23
Set	1,335	41.59	3.36	2.83	10.17	9.53
Aggregation	1,136	37.22	3.65	2.12	11.07	9.16
Editing	926	39.98	3.45	2.70	5.31	6.31
Answerability	1,997	40.40	3.35	-	-	-
Total	8,366	38.06	3.54	2.20	7.00	7.14

Table 2: Data analysis across six settings.

4.5 Data Analysis

Our M^{3}GQA dataset consists of 8,366 instances, with an average of 3.54 entities in each query. We conduct a statistical analysis of the data quantity in six settings and examine the average number of query tokens, entities, reasoning hops, as well as the average number of triplets and nodes in the reasoning paths for each setting. Relevant results are presented in Table 2. We additionally visualize the number of query entities and the inference hops of queries in the dataset, as shown in Figure 4.

5 Experiments

5.1 Experimental Settings

Benchmark Settings. We construct M^{3}GQA based on Freebase (Bollacker et al., 2008) and randomly split the data into train/validation/test sets by 6/1/3 for each setting.

Baselines. We evaluate baselines on M^{3}GQA, which can be divided into 3 classes:
LLM-only Methods: We adopt GPT-4 (model: gpt-4-turbo), ChatGPT (model: gpt-3.5-turbo), and Claude-3.5 (model: claude-3.5-sonnet) to answer queries directly.
Traditional Methods: We first retrieve all shortest paths between query nodes as a rule-based baseline (abbrev. SP). Besides, we leverage sparse retriever (BM25 (Robertson and Zaragoza, 2009)) and dense retriever (paraphrase-multilingual-MiniLM-L12-v2 (Wang et al., 2021)) to retrieve relevant triplets, and feed the triplets with similarity scores from high to low into GPT-4 to generate answers.
GraphRAG Methods: We reproduce several state-of-the-art GraphRAG methods on M^{3}GQA, includ-

Baselines	Single-hop Setting				Multi-hop Setting				Set Setting				Aggregation Setting			
	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1
LLM-Only Methods																
GPT4	33.41	34.10	35.85	33.26	11.87	12.20	12.82	11.66	21.92	24.79	44.75	28.00	25.65	26.98	26.03	24.93
ChatGPT	29.64	30.35	32.18	28.94	7.73	7.90	8.39	6.99	16.72	19.02	34.75	19.00	21.01	21.59	23.46	21.99
Claude3.5	30.71	31.78	34.77	31.32	10.14	10.76	12.12	8.86	17.24	19.40	40.25	21.50	24.75	24.05	27.57	25.51
Traditional Methods																
SP	**82.48**	**83.96**	**87.47**	**80.13**	10.68	10.83	11.19	10.26	26.49	28.22	43.00	33.00	43.19	44.21	47.21	41.64
Sparse	65.06	66.28	69.33	64.15	14.82	15.48	16.78	14.22	34.57	37.51	54.50	37.75	36.68	37.70	40.47	35.78
Dense	69.04	70.54	74.30	65.66	**25.20**	**28.26**	**35.20**	**22.61**	**43.30**	**47.16**	**70.00**	45.00	**49.01**	**50.57**	**55.13**	**44.57**
GraphRAG Methods																
ToG	71.33	71.94	74.00	68.00	19.00	19.31	20.00	20.00	23.13	25.42	44.00	30.00	28.67	28.97	30.00	26.00
RoG	57.05	58.15	61.12	56.59	16.78	17.59	19.58	15.38	27.94	31.21	50.75	34.50	35.72	36.75	39.59	33.72
RoG-SFT	68.54	69.74	73.00	67.60	17.10	18.29	20.75	16.32	31.14	34.81	58.00	37.00	34.01	34.66	36.36	32.26
G-Retriever	55.97	56.92	59.40	55.08	21.74	23.05	25.87	21.21	36.97	41.45	69.75	**47.50**	38.78	39.59	41.64	38.71

Table 3: Performance of different methods across four general settings of M^{3}GQA. We randomly sample 50 entries from each setting for the evaluation of ToG considering the time and expenses.

ing: G-Retriever (He et al., 2024), ToG (Sun et al., 2024), RoG (Luo et al., 2024). We additionally replace the retriever of RoG with Qwen2.5-7B-Instruct, which is trained with SFT on the training set of M^{3}GQA (abbrev. RoG-SFT), as a baseline.

The introduction of these methods and details of their reproduction are presented in Appendix C.1.

Evaluation Metrics. We evaluate the performance of baselines from two aspects. For graph retrieval, we utilize two metrics to measure the effectiveness, including: recall of triplets in reasoning paths (Recall) and whether all answer nodes have been retrieved (Acc). In addition, we also report the total number of triplets contained in the subgraphs retrieved (Num) by different baselines. For answer generation, we utilize Macro-F1 (abbrev. MacF1), Micro-F1 (abbrev. MicF1), Hit, and Hit@1 to measure the quality of predictions. For the answerability setting, since it is essentially a binary classification task, we use Accuracy, Precision, Recall, and F1 as evaluation metrics.

5.2 Overall Performance

We evaluate the baselines across six settings from answer prediction and graph retrieval aspects, which are shown in Table 3, 4 and 5, we summarize our findings about M^{3}GQA as follows:

Comparison between different settings Although M^{3}GQA overall presents a significant challenge, there are still noticeable differences in the difficulty of six settings. From the results, the single-hop setting appears to be the simplest, while the multi-hop setting is the toughest, possibly due to the complex reasoning involved in multi-entity multi-hop queries. Meanwhile, the set and aggregation settings also demonstrate a considerable degree of difficulty. Considering there are multiple answer entities in the set setting, we additionally introduce the Exact Match (EM) metric to reflect the model's ability to provide complete answers, which is shown in Table 10 in Appendix C.3. It appears that the current model's performance in this regard is not satisfying.

Baselines	Editing Setting				Answerability Setting			
	MacF1	MicF1	Hit	Hit@1	Acc	Precision	Recall	F1
Traditional Methods								
SP	5.90	5.98	6.12	5.76	47.25	60.59	43.70	50.78
Sparse	8.92	9.50	10.79	8.63	50.58	65.98	42.63	51.79
Dense	17.95	19.94	**24.46**	17.63	52.92	62.96	59.25	61.05
GraphRAG Methods								
ToG	**20.00**	**20.00**	20.00	**20.00**	48.00	**68.42**	39.39	50.00
RoG	8.19	9.25	11.51	7.19	46.41	61.40	37.53	46.59
RoG-SFT	8.82	10.13	12.59	8.27	48.75	63.10	42.63	50.88
G-Retriever	11.82	12.80	15.47	12.59	**54.26**	62.28	**67.29**	**64.69**

Table 4: Performance of different methods across two special settings of M^{3}GQA.

Comparison between different Baselines Since GraphRAG requires the model to select entities from the graph as answers, the performance of LLM-only methods is not ideal. Compared to the current GraphRAG approaches, traditional retrieval methods still show strong competitiveness, likely due to the more comprehensive content retrieved by these methods. In addition, current state-of-the-art GraphRAG models perform suboptimally on M^{3}GQA, indicating that our dataset is relatively complex and presents challenges to the retrieval and answering strategies of previous methods. A detailed analysis of the performance of state-of-the-art methods can be found in Appendix C.2.

Correlation between retrieval and generation By analyzing the results of graph retrieval and answer prediction simultaneously, we can see a clear

Baselines	Single-hop Setting			Multi-hop Setting			Set Setting			Aggregation Setting			Editing Setting		
	Recall	Acc	Num	Recall	Acc	Num	Recall	Acc	Num	Recall	Acc	Num	Recall	Acc	Num
Traditional Methods															
SP	92.00	95.03	51.31	61.86	10.72	34.26	40.78	15.25	43.95	45.53	55.13	52.60	60.59	8.63	32.28
Sparse	41.60	79.27	100.00	32.68	24.01	100.00	26.90	39.00	100.00	23.92	59.24	100.00	31.84	19.42	100.00
Dense	67.73	88.55	100.00	51.34	66.43	100.00	46.50	53.75	100.00	45.53	82.99	100.00	51.35	59.71	100.00
GraphRAG Methods															
ToG	41.87	74.00	4.89	32.82	16.00	5.64	12.60	4.00	5.98	8.81	38.00	5.76	34.92	24.00	5.60
RoG	38.17	62.42	58.06	13.36	24.01	41.33	12.31	30.75	77.47	12.07	56.89	76.50	14.56	23.38	41.71
RoG-SFT	43.83	70.41	19.26	12.11	25.87	48.10	13.10	34.50	124.64	9.56	48.68	58.32	11.07	31.65	45.66
G-Retriever	50.15	82.94	208.53	41.72	75.76	236.10	29.44	60.50	207.25	31.88	79.47	199.60	40.43	67.99	226.91

Table 5: Evaluation of graph retrieval in different models across five settings. Since a considerable portion of queries in the answerability setting could not retrieve subgraphs, we do not report the results in that setting.

correlation between them, indicating that improving the performance of the retrievers is crucial to the overall performance. In M^{3}GQA, we provide ground-truth reasoning paths, which are believed to be helpful in enhancing the retriever's capabilities. We also identify some special cases from the results. For example, in some settings (e.g., set setting), the retrieval accuracy (Acc) of RoG is lower than the Hit value in answer prediction. This suggests that some queries might be correctly answered relying solely on the knowledge in LLMs. Moreover, although G-retriever shows high retrieval accuracy, the answer prediction performance is relatively average. We believe this may be due to incomplete retrieval paths, as the recall score of the retriever is relatively low. Furthermore, we believe that besides recall, accuracy, and triplet number, the order in which the retrieved triplets are fed into the LLMs can also affect the model's answer prediction performance. We validate this idea through experiments in Appendix C.5.

5.3 Difficulty of M^{3}GQA

We further construct two additional settings: simple single-hop and simple multi-hop settings, by reducing the number of entities in queries. We evaluate the models' performance from both perspectives of answer prediction and graph retrieval to show the difficulty of M^{3}GQA, with results in Table 6 and 7. We do not report the results of SP because it is not applicable to queries that involve only a single entity. By comparing these results with the original single-hop and multi-hop settings in Table 3 and 5, we can observe that increasing the number of entities can significantly improve the difficulty of queries, which further highlights the challenges and importance of multi-entity queries. We also compare M^{3}GQA with classical datasets such as WebQSP and CWQ. Due to the lack of ground-truth reasoning paths in WebQSP and CWQ, we only evaluate the model's ability to predict answers, which are shown in Table 8. We can see that compared to CWQ and WebQSP, the models' performance on M^{3}GQA shows a significant decline, indicating that our dataset presents a considerable challenge to current models.

Baselines	Simple Single-hop				Simple Multi-hop			
	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1
LLM-Only Methods								
GPT4	34.83	35.65	37.98	35.19	23.84	24.61	26.65	22.98
ChatGPT	30.55	31.25	32.83	30.69	19.99	20.68	22.25	19.56
Claude3.5	30.15	31.45	34.76	31.33	20.10	20.95	23.23	20.05
Traditional Methods								
Sparse	62.44	62.95	64.16	62.88	35.72	36.49	38.14	34.47
Dense	84.50	85.30	87.34	84.12	50.95	53.25	58.19	48.17
GraphRAG Methods								
ToG	81.47	82.21	84.00	82.00	54.00	54.00	54.00	54.00
RoG	58.69	59.34	60.73	59.01	32.04	33.55	36.67	31.05
RoG-SFT	69.38	69.83	70.82	68.88	35.15	36.27	38.63	33.99
G-Retriever	67.50	68.33	70.39	68.24	43.40	44.68	47.43	41.56

Table 6: Difficulty comparison of M^{3}GQA against other two simple settings.

Baselines	Simple Single-hop			Simple Multi-hop		
	Recall	Acc	Num	Recall	Acc	Num
Traditional Methods						
Sparse	29.94	64.81	100.00	24.63	36.92	100.00
Dense	88.84	96.35	100.00	61.53	70.17	100.00
GraphRAG Methods						
ToG	83.00	90.00	2.88	45.20	36.00	4.90
RoG	33.80	48.07	20.90	23.96	14.42	20.44
RoG-SFT	43.78	56.22	5.45	24.69	14.67	14.99
G-Retriever	62.12	88.63	217.99	43.26	71.15	213.55

Table 7: Evaluation of the graph retrieval process of different models on two simple settings.

6 Conclusion

In this paper, we introduce the challenge of multi-entities queries and construct a Multi-Entity Multi-Hop Multi-Setting Graph Question Answering Benchmark, named M^{3}GQA, to evaluate the ca-

Baselines	WebQSP		CWQ		M^{3}GQA	
	MacF1	Hit	MacF1	Hit	MacF1	Hit
ToG	-	82.60	-	67.60	37.72	43.72
RoG	70.26	86.67	54.63	61.94	38.81	48.34
G-Retriever	53.41	73.46	-	-	38.73	49.68

Table 8: Difficulty comparison of M^{3}GQA against WebQSP and CWQ.

pabilities of GraphRAG methods. To ensure the quality and diversity of the data, we introduce a non-template approach for data construction and classify the data into six settings according to query types and complexity. We reproduce the state-of-the-art methods and assess their performance focusing on both graph retrieval and answer prediction. Experimental results show that M^{3}GQA is highly challenging and holds significant implications for the future development of the field.

Ethical Consideration

In this paper, we construct M^{3}GQA based on Freebase, with carefully designed methods to minimize the ethical risks. As Freebase contains publicly available knowledge, no private or sensitive individual data is included, ensuring compliance with data privacy standards. M^{3}GQA is intended strictly for academic research and educational purposes, with clear guidelines to prevent misuse in harmful applications. Furthermore, the dataset adheres to Freebase's licensing terms, with proper attribution required. In summary, M^{3}GQA poses low ethical risk while enabling advancements of both retrieval and reasoning processes in GraphRAG systems.

Limitations

In this paper, we propose the first GraphRAG benchmark named M^{3}GQA consisting of multi-entity queries. To facilitate the construction of the dataset, we utilize the knowledge graph (Freebase) as the primary source of graph-structured information. We believe that extending this approach to more general graph data, multilingual data (Zhao and Zhang, 2024), and other vertical domains (e.g., biomedical (Jiang et al., 2024b), academic (Jin et al., 2024), and financial (Arslan and Cruz, 2024)) represents a promising direction for future improvements. Besides, due to time and expense constraints, we only reproduce a subset of the state-of-the-art methods. Expanding the reproduction to include more baselines would undoubtedly contribute to further advancements in the field. Additionally, this work primarily focuses on graph question answering scenarios, which is currently the most widely applied task for GraphRAG methods. Designing benchmarks for other GraphRAG tasks (e.g., long context summary) is also a direction worth exploring.

Acknowledgments

This work is supported by the National Key Research and Development Plan of China (2023YFB4502305), and Ant Group through Ant Research Intern Program. This work is also supported in part by Ucap Cloud and the State Key Laboratory of General Artificial Intelligence.

References

Muhammad Arslan and Christophe Cruz. 2024. Business-rag: Information extraction for business insights. In *Proceedings of the 21st International Conference on Smart Business Technologies, ICSBT 2024, Dijon, France, July 9-11, 2024*, pages 88–94.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1533–1544.

Kurt D. Bollacker, Colin Evans, Praveen K. Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks.

Shulin Cao, Jiaxin Shi, Liangming Pan, Lunyiu Nie, Yutong Xiang, Lei Hou, Juanzi Li, Bin He, and Hanwang Zhang. 2022. KQA pro: A dataset with explicit compositional programs for complex question answering over knowledge base. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 6101–6119.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the*

North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186.

Junnan Dong, Qinggang Zhang, Xiao Huang, Keyu Duan, Qiaoyu Tan, and Zhimeng Jiang. 2023. Hierarchy-aware multi-hop question answering over knowledge graphs. In *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, pages 2519–2527. ACM.

Mohnish Dubey, Debayan Banerjee, Abdelrahman Abdelkawi, and Jens Lehmann. 2019. Lc-quad 2.0: A large dataset for complex question answering over wikidata and dbpedia. In *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part II*, volume 11779 of *Lecture Notes in Computer Science*, pages 69–78.

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.

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on RAG meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2024, Barcelona, Spain, August 25-29, 2024*, pages 6491–6501.

Yu Gu, Sue Kase, Michelle Vanni, Brian M. Sadler, Percy Liang, Xifeng Yan, and Yu Su. 2021. Beyond I.I.D.: three levels of generalization for question answering on knowledge bases. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 3477–3488.

Tiezheng Guo, Qingwen Yang, Chen Wang, Yanyi Liu, Pan Li, Jiawei Tang, Dapeng Li, and Yingyou Wen. 2024a. Knowledgenavigator: Leveraging large language models for enhanced reasoning over knowledge graph.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2024b. Lightrag: Simple and fast retrieval-augmented generation.

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.

Yuntong Hu, Zhihan Lei, Zheng Zhang, Bo Pan, Chen Ling, and Liang Zhao. 2024. Grag: Graph retrieval-augmented generation.

Yongfeng Huang, Yanyang Li, Yichong Xu, Lin Zhang, Ruyi Gan, Jiaxing Zhang, and Liwei Wang. 2023. Mvp-tuning: Multi-view knowledge retrieval with prompt tuning for commonsense reasoning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 13417–13432.

Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Xin Zhao, and Ji-Rong Wen. 2023a. Structgpt: A general framework for large language model to reason over structured data. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 9237–9251.

Jinhao Jiang, Kun Zhou, Wayne Xin Zhao, Yang Song, Chen Zhu, Hengshu Zhu, and Ji-Rong Wen. 2024a. Kg-agent: An efficient autonomous agent framework for complex reasoning over knowledge graph.

Jinhao Jiang, Kun Zhou, Xin Zhao, and Ji-Rong Wen. 2023b. Unikgqa: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graph. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*.

Xinke Jiang, Ruizhe Zhang, Yongxin Xu, Rihong Qiu, Yue Fang, Zhiyuan Wang, Jinyi Tang, Hongxin Ding, Xu Chu, Junfeng Zhao, and Yasha Wang. 2024b. Hykge: A hypothesis knowledge graph enhanced framework for accurate and reliable medical llms responses.

Bowen Jin, Chulin Xie, Jiawei Zhang, Kashob Kumar Roy, Yu Zhang, Zheng Li, Ruirui Li, Xianfeng Tang, Suhang Wang, Yu Meng, and Jiawei Han. 2024. Graph chain-of-thought: Augmenting large language models by reasoning on graphs. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 163–184.

Shiyang Li, Yifan Gao, Haoming Jiang, Qingyu Yin, Zheng Li, Xifeng Yan, Chao Zhang, and Bing Yin. 2023. Graph reasoning for question answering with triplet retrieval. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 3366–3375.

Zhuoqun Li, Xuanang Chen, Haiyang Yu, Hongyu Lin, Yaojie Lu, Qiaoyu Tang, Fei Huang, Xianpei Han, Le Sun, and Yongbin Li. 2024. Structrag: Boosting knowledge intensive reasoning of llms via inference-time hybrid information structurization.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2829–2839.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis,

Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. 2024. Reasoning on graphs: Faithful and interpretable large language model reasoning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 9802–9822.

Costas Mavromatis and George Karypis. 2024. Gnn-rag: Graph neural retrieval for large language model reasoning.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2381–2391.

OpenAI. 2024. Gpt-4 technical report.

Boci Peng, Yongchao Liu, Xiaohe Bo, Sheng Tian, Baokun Wang, Chuntao Hong, and Yan Zhang. 2024a. Subgraph retrieval enhanced by graph-text alignment for commonsense question answering. In *Machine Learning and Knowledge Discovery in Databases. Research Track - European Conference, ECML PKDD 2024, Vilnius, Lithuania, September 9-13, 2024, Proceedings, Part VI*, pages 39–56.

Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, and Siliang Tang. 2024b. Graph retrieval-augmented generation: A survey.

Aleksandr Perevalov, Dennis Diefenbach, Ricardo Usbeck, and Andreas Both. 2022. Qald-9-plus: A multilingual dataset for question answering over dbpedia and wikidata translated by native speakers. In *16th IEEE International Conference on Semantic Computing, ICSC 2022, Laguna Hills, CA, USA, January 26-28, 2022*, pages 229–234.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Andrey Sakhovskiy, Mikhail Salnikov, Irina Nikishina, Aida Usmanova, Angelie Kraft, Cedric Möller, Debayan Banerjee, Junbo Huang, Longquan Jiang, Rana Abdullah, Xi Yan, Dmitry Ustalov, Elena Tutubalina, Ricardo Usbeck, and Alexander Panchenko. 2024. TextGraphs 2024 shared task on text-graph representations for knowledge graph question answering. In *Proceedings of TextGraphs-17: Graph-based Methods for Natural Language Processing*, pages 116–125.

Mikhail Salnikov, Hai Le, Prateek Rajput, Irina Nikishina, Pavel Braslavski, Valentin Malykh, and Alexander Panchenko. 2023. Large language models meet knowledge graphs to answer factoid questions. In *Proceedings of the 37th Pacific Asia Conference on Language, Information and Computation, PACLIC 2023, The Hong Kong Polytechnic University, Hong Kong, SAR, China, 2-4 December 2023*, pages 635–644.

Priyanka Sen, Alham Fikri Aji, and Amir Saffari. 2022. Mintaka: A complex, natural, and multilingual dataset for end-to-end question answering. In *Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, October 12-17, 2022*, pages 1604–1619.

Haitian Sun, Tania Bedrax-Weiss, and William W. Cohen. 2019. Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2380–2390.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel M. Ni, Heung-Yeung Shum, and Jian Guo. 2024. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 641–651.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4149–4158.

Dhaval Taunk, Lakshya Khanna, Siri Venkata Pavan Kumar Kandru, Vasudeva Varma, Charu Sharma, and Makarand Tapaswi. 2023. Grapeqa: Graph augmentation and pruning to enhance question-answering. In *Companion Proceedings of the ACM Web Conference*

2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023, pages 1138–1144.

Wenhui Wang, Hangbo Bao, Shaohan Huang, Li Dong, and Furu Wei. 2021. Minilmv2: Multi-head self-attention relation distillation for compressing pretrained transformers. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, pages 2140–2151.

Yuqi Wang, Boran Jiang, Yi Luo, Dawei He, Peng Cheng, and Liangcai Gao. 2024. Reasoning on efficient knowledge paths: Knowledge graph guides large language model for domain question answering.

Yilin Wen, Zifeng Wang, and Jimeng Sun. 2024. Mindmap: Knowledge graph prompting sparks graph of thoughts in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 10370–10388.

Yike Wu, Nan Hu, Sheng Bi, Guilin Qi, Jie Ren, Anhuan Xie, and Wei Song. 2023. Retrieve-rewrite-answer: A kg-to-text enhanced llms framework for knowledge graph question answering.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 535–546.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*.

Jing Zhang, Xiaokang Zhang, Jifan Yu, Jian Tang, Jie Tang, Cuiping Li, and Hong Chen. 2022. Subgraph retrieval enhanced model for multi-hop knowledge base question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 5773–5784.

Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J. Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 6069–6076.

Jinman Zhao and Xueyan Zhang. 2024. Large language model is not a (multilingual) compositional relation reasoner. In *First Conference on Language Modeling*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2024. A survey of large language models.

A Data Analysis of M^{3}GQA

A.1 Statistics of WebQSP and CWQ

We calculate the number of multi-entity queries in WebQSP and CWQ respectively. The results are presented in Figure 3 below.

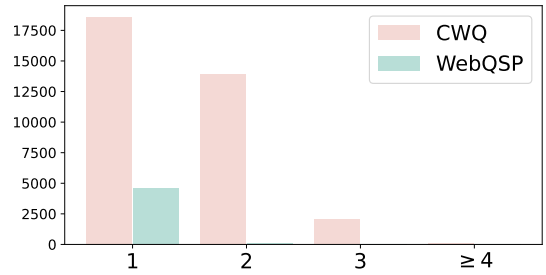


Figure 3: Statistics on the number of query entities in the WebQSP and CWQ datasets.

A.2 Visualization of the Data

We visualize the number of query entities and the number of reasoning hops for each query under six settings, and present the results in Figure 4.

A.3 Pattern of Reasoning Paths

To provide a clearer understanding of the various settings discussed in this paper, we present examples of reasoning path patterns for each setting, which are presented in Figure 5.

B Details of Filtering

B.1 Guidelines of Manual Filtering

We assign samples from the validation and test sets of M^{3}GQA randomly to five volunteers for manual inspection. Each volunteer identify any samples they consider unreasonable, and then all five volunteers judge these samples individually, with a majority vote determining whether each sample should be retained or discarded. The guidelines for determining whether a sample is reasonable are as follows:

• Clarity of the question: Check if the query includes any extraneous query entities, if the question is ambiguous, if it is likely to have a clear answer, and exclude subjective questions.

• Manual reasoning path and answer validation: For each question, we manually track the reasoning path and compare the answer with the ground-truth provided in the dataset.

B.2 Data Statistics

We show the proportions of data filtered out in each filtering stage in Table 9.

From the results, we can observe that (1) Each filtering stage is capable of eliminating a portion of low-quality data, which validates the necessity of every filtering stage. (2) The entire data construction pipeline is reliable because the proportion of manually filtered out unreasonable data is relatively small.

Furthermore, to ensure the reliability of the manual filtering process, we calculate the inter-annotator agreement using Fleiss' Kappa, which measures the level of agreement among multiple annotators. The Fleiss' Kappa score for our annotators is 0.74, indicating substantial agreement.

C Additional Experiments

C.1 Implementation Details

For traditional methods, we retrieve the top 100 triplets according to similarity scores and feed them into GPT-4 as additional inputs for question answering. For GraphRAG methods, we leverage the open-source codes and first reproduce them on the WebQSP dataset, ensuring that the reproduction results are consistent with those reported in original papers. Then we evaluate them on M^{3}GQA and adjust hyper-parameters according to the performance on the validation set. For fair comparison, we use GPT-4 as the base generator.

C.2 Introduction and Analysis of Baselines

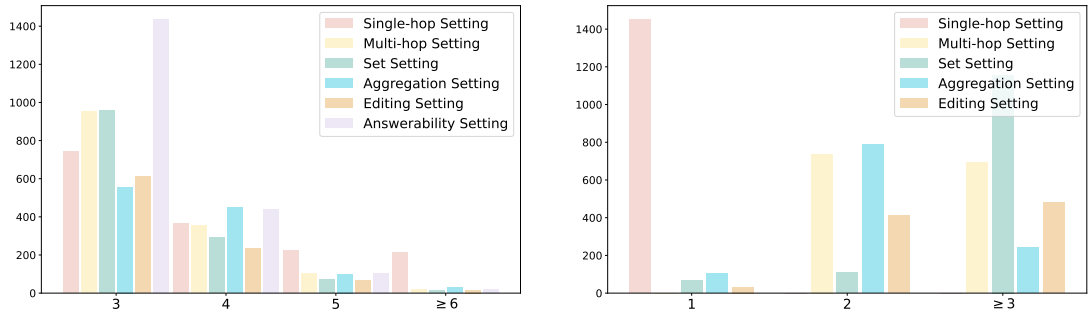
In this section, we will provide a detailed introduction to the current state-of-the-art GraphRAG methods and thoroughly analyze the potential issues that may arise when applying them to M^{3}GQA.

ToG This method[2] (Sun et al., 2024) proposes utilizing LLMs to perform graph retrieval starting from query entities, iteratively selecting edges and neighboring nodes related to the current node until the model can answer the question. Applying the method to M^{3}GQA may face the following challenges:

• Error Propagation in Iterative Search: Since the method employs a BFS-like iterative retrieval approach, any errors occurring at a particular search layer will render subsequent searches invalid.

• Complexity in Handling Nodes with High Connectivity: If a node in the search process is con-

[2] https://github.com/IDEA-FinAI/ToG



(a) Number of query entities across six settings. (b) Number of reasoning hops across five settings.

Figure 4: Visualization of query entities and reasoning hops in M^{3}GQA.

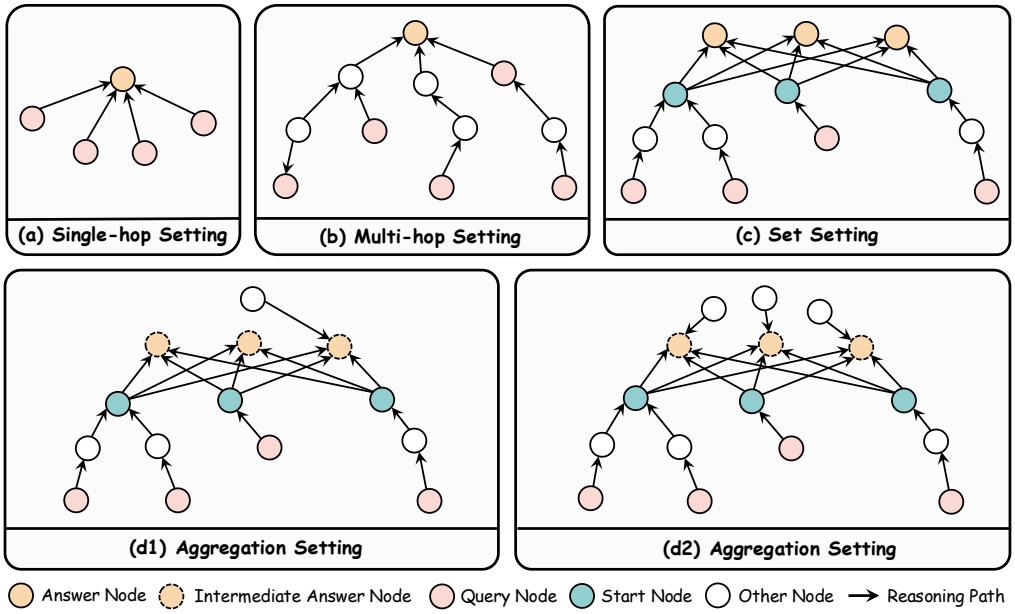


Figure 5: Pattern of reasoning paths under single-hop setting, multi-hop setting, set setting, and aggregation setting.

	Single-hop	Multi-hop	Set	Aggregation	Editing	Answerability
Rule-Based	4.75%	7.41%	8.44%	8.57%	7.96%	6.47%
LLM-Based	5.11%	15.88%	19.09%	23.40%	17.47%	9.72%
Manual (test)	2.94%	4.46%	5.44%	7.59%	3.14%	3.54%
Manual (val)	2.57%	4.19%	5.83%	7.21%	3.19%	3.41%

Table 9: Data statistics of each filtering stage.

nected to many or highly complex relationships, the model may struggle to correctly select the relevant edges and neighboring nodes.

• Over-confidence of Large Language Models: Due to the tendency of large language models to be overly confident, this approach might lead to the model answering the question prematurely before reaching the correct answer entity node, which would degrade answer prediction performance.

RoG This method[3] (Luo et al., 2024) proposes to train an LLM to generate several relation chains as reasoning plans, and then parsing out reasoning paths from the graph that match the relation chains. During the reasoning phase, since the trained language model has not seen the relations in the graph from the test set in advance, it becomes difficult to parse out reasoning paths from the graph that match the reasoning plan generated by the LLM.

[3] https://github.com/RManLuo/reasoning-on-graphs

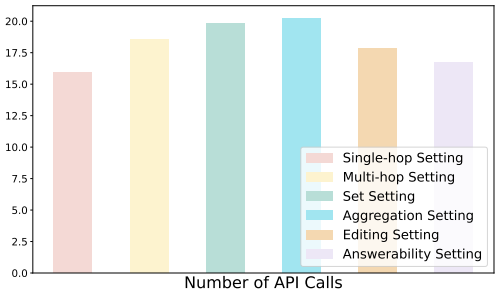


Figure 6: Number of API Calls of ToG on six settings.

G-Retriever This method[4] (He et al., 2024) proposes using the Prize-Collected Steiner Tree (PCST) algorithm to retrieve subgraphs relevant to the query. However, the method's performance on M^3GQA is not ideal, possibly due to the following reasons:

• The subgraphs retrieved using the PCST algorithm typically have low recall values, indicating that the retrieved reasoning paths may not be comprehensive enough, posing a high requirement for the answer reasoning ability of LLMs.

• The subgraphs retrieved using this algorithm are relatively large, which poses a challenge to the context comprehension ability of LLMs for subsequent answer prediction.

• Since this retrieval algorithm cannot obtain the relevance scores for the corresponding triplets, it is difficult to determine an appropriate triplet order as input for the answer generator, which could also impact the model's performance.

C.3 EM Results on Set Setting

We additionally introduce the EM metric to evaluate the models' ability to answer the queries without any omission. By analyzing the results in Table 10, we can find that the model's ability in this regard still requires improvement.

C.4 Statistics of API Calls in ToG

ToG proposes using LLMs to retrieve relevant information from the graph, which introduces significant API expenses, especially for multi-entity queries. We calculate the average number of API calls per query for ToG across six settings, and the results are shown in Figure 6.

[4] https://github.com/XiaoxinHe/G-Retriever

C.5 Retrieval Factors Affecting the Answer Performance

In the previous section, we evaluate the retrieval quality of the retriever using Recall, Acc, and Num, and briefly analyze the relationship between the retrieval quality and the final answer prediction performance. Here, we conduct an in-depth analysis of the impact of the number and order of retrieved triplets on the prediction results. Relevant results analysis are as follows.

C.5.1 Order of Retrieved Triplets

In this paper, we further investigate the impact of the concatenation order of triplets retrieved by the dense retriever on the model's answer prediction performance. Specifically, we randomly shuffle the retrieved triplets and compare them with the original Dense method. Table 11 and 12 show a comparison of the results from the two methods across six settings. Through analysis, we can observe that the order of the triplets does have an impact on the model's prediction performance.

C.5.2 Number of Retrieved Triplets

To facilitate control over the number of retrieved triplets, we use the dense retriever-based approach as the backbone model for GraphRAG. We explore the model's task performance under six different settings when the number of retrieved triplets is 50, 100, 200, and 300. We evaluate the model performance using the same metrics described in the main text, focusing both on the retrieval and answer prediction process. Corresponding results can be found in Figure 7, 8 and 9. Through the experiments, we can find a rough positive correlation between the answer prediction performance and the number of retrieved triplets.

C.6 Performance of Hybrid Baselines

We additionally incorporate a hybrid baseline that combines the top 100 results based on the dense retrieval with the shortest paths between query nodes. The results are shown in Table 13, 14, and 15.

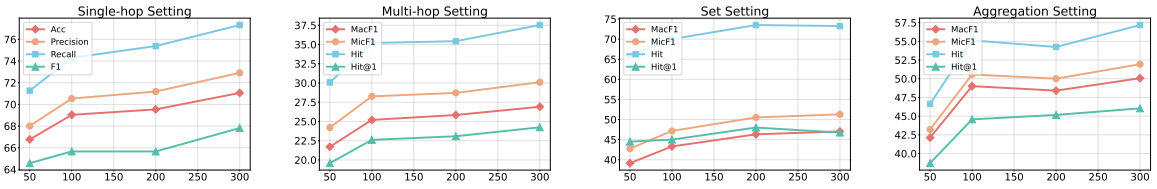
From the results above, we can observe that SP performs relatively well in simpler scenarios (such as single-hop setting), but they perform poorly in more complex scenarios (such as multi-hop setting). Furthermore, the hybrid method effectively combines the advantages of the SP method and dense retrieval, achieving good results in scenarios that involve both single-hop and multi-hop reasoning.

Set Setting	GPT4	ChatGPT	Claude3.5	MiniLM	BM25	ToG	RoG	RoG-SFT	G-Retriever
EM	6.25	4.00	3.75	22.25	17.00	6.00	9.00	9.50	10.25

Table 10: EM results of different models on the set setting.

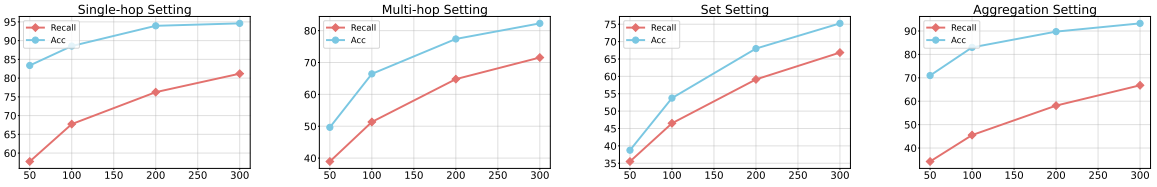
Baselines	Single-hop				Multi-hop				Set				Aggregation			
	Macro-F1	Micro-F1	Hit	Hit@1	Macro-F1	Micro-F1	Hit	Hit@1	Macro-F1	Micro-F1	Hit	Hit@1	Macro-F1	Micro-F1	Hit	Hit@1
Dense	69.04	70.54	74.30	65.66	25.20	28.26	35.20	22.61	43.30	47.16	70.00	45.00	49.01	50.57	55.13	44.57
Dense +Shuffle	69.96	71.34	74.95	67.39	23.28	25.75	31.24	22.14	43.57	47.49	71.00	48.75	45.55	46.91	50.73	42.23

Table 11: Effect of triplets order for answer prediction on four general settings.



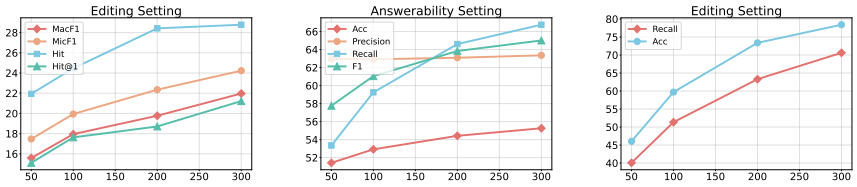
(a) Single-hop Setting. (b) Multi-hop Setting. (c) Set Setting. (d) Aggregation Setting.

Figure 7: The answer prediction performance of dense retriever-based methods varies with the number of retrieved triplets across four settings.



(a) Single-hop Setting. (b) Multi-hop Setting. (c) Set Setting. (d) Aggregation Setting.

Figure 8: The graph retrieval performance of dense retriever-based methods varies with the number of retrieved triplets across four settings.



(a) Editing Setting. (b) Answeability Setting. (c) Editing Setting.

Figure 9: The performance of dense retriever-based methods varies with the number of retrieved triplets across two special settings.

Baselines	Editing				Answerability			
	Macro-F1	Micro-F1	Hit	Hit@1	Acc	Precision	Recall	F1
Dense	17.95	19.94	24.46	17.63	52.92	62.96	59.25	61.05
Dense +Shuffle	17.81	19.92	25.18	16.19	54.76	64.01	62.47	63.23

Table 12: Effect of triplets order for answer prediction on two special settings.

Additionally, the retrieval performance further supports the correlation between retrieval quality and generation quality.

C.7 Generalization of Query Structures

To investigate the generalization of different query structures, we train RoG using training data from both single-hop and multi-hop settings and evaluate it across all settings (abbrev. RoG-subset). Furthermore, for a fair comparison, we randomly sample an equivalent amount of training data as the former from all settings (abbrev. RoG-general). We show results in Table 16, 17, and 18.

From the results, we observe that RoG-subset

Baselines	Single-hop Setting				Multi-hop Setting				Set Setting				Aggregation Setting			
	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1
SP	**82.48**	**83.96**	**87.47**	**80.13**	10.68	10.83	11.19	10.26	26.49	28.22	43.00	33.00	43.19	44.21	47.21	41.64
Dense	69.04	70.54	74.30	65.66	**25.20**	**28.26**	**35.20**	**22.61**	43.30	47.16	70.00	45.00	49.01	50.57	55.13	44.57
Hybrid	79.07	80.67	84.67	77.11	24.81	27.43	33.10	**22.61**	**45.85**	**50.04**	**71.50**	**47.75**	**52.39**	**53.89**	**58.06**	**49.85**

Table 13: Performance of the hybrid baseline across four general settings.

Baselines	Editing Setting				Answerability Setting			
	MacF1	MicF1	Hit	Hit@1	Acc	Precision	Recall	F1
SP	5.90	5.98	6.12	5.76	47.25	60.59	43.70	50.78
Dense	**17.95**	**19.94**	**24.46**	**17.63**	52.92	**62.96**	59.25	61.05
Hybrid	17.20	19.26	23.38	16.19	**55.26**	**62.90**	**68.63**	**65.64**

Table 14: Performance of the hybrid baseline across two special settings.

and RoG-general outperform vanilla RoG in the vast majority of scenarios. This is attributed to the fact that vanilla RoG is trained on WebQSP and CWQ, which have a significantly different data distribution from that of M^3GQA. Furthermore, we find that RoG-subset surpasses RoG-general in the single-hop and multi-hop settings; however, the opposite is true for other settings. This indicates that there are substantial differences in query structures across various settings.

C.8 Ablation Studies

We further conduct ablation studies on reasoning path lengths, query entities count, and reasoning path construction.

Reasoning Path Lengths We first conduct an ablation study on reasoning path lengths, selecting Dense, Sparse, and G-Retriever as baselines for experiments across three general settings (the single-hop setting, having only one hop by definition, is excluded from this analysis). The path lengths are categorized into four groups (1-4 hops), with results as shown in Table 19, Table 20, and Table 21. We can observe that: (1) In the vast majority of cases, the longer the reasoning path, the more difficult the query becomes, and the lower the performance of the baselines. The set setting is an exception when the reasoning paths reach 4, as the relatively small number of test samples leads to significant fluctuations in the results. (2) Compared to the semantic matching-based retrieval methods of Dense and Sparse, G-Retriever performs relatively poorly in single-hop scenarios but demonstrates better performance in multi-hop queries. This highlights G-Retriever's advantage in retrieving subgraphs for capturing higher-order information.

Query Entities Count We conduct an ablation study on the number of entities in the query, dividing them into three groups: 3, 4, and ≥ 5. The results are as shown in Table 22, Table 23, and Table 24. We can observe that: Except for the single-hop setting, the model's performance decreases as the number of entities increases in all other settings. This indicates that a higher number of entities leads to greater problem difficulty. In the single-hop setting, the results across the three groups are comparable. A possible reason is that the model tends to retrieve one-hop neighbors, preventing a decline in performance. However, it is worth noting that this observation does not contradict the conclusions in Section 5.3, because when the number of entities is 1 or 2, the task only involves examining neighbors of a single entity or paths of length 2 between two entities. Thus, when the number of entities ≥ 3, the difficulty increases significantly.

Reasoning Path Construction We implement an approach in the multi-hop setting where all shortest paths between the answer entities and query entities are used as reasoning paths. We find that only 66.43% of the samples had identical reasoning paths between the tree sampling method and the shortest-path method. We then compute the average recall of the shortest-path method (treating tree sampling reasoning paths as ground truth), which is 83.56%. This indicates that the shortest-path method fails to retrieve all reasoning paths—meaning some reasoning paths are not the shortest paths between the two sets of entities. Additionally, we measure the average precision, which is 32.54%, suggesting that the reasoning paths obtained via the shortest-path method contain substantial redundant information.

D Algorithm and Formalization

D.1 Algorithm

We show the algorithm pseudo code of the data construction pipeline in Algorithm 1.

Baselines	Single-hop Setting			Multi-hop Setting			Set Setting			Aggregation Setting			Editing Setting		
	Recall	Acc	Num	Recall	Acc	Num	Recall	Acc	Num	Recall	Acc	Num	Recall	Acc	Num
SP	92.00	95.03	51.31	61.86	10.72	34.26	40.78	15.25	43.95	45.53	55.13	52.60	60.59	8.63	32.28
Dense	67.73	88.55	100.00	51.34	66.43	100.00	46.50	53.75	100.00	45.53	82.99	100.00	51.35	59.71	100.00
Hybrid	97.93	99.78	132.94	79.00	69.00	124.17	66.78	60.25	130.80	67.18	90.62	137.89	77.87	61.51	121.49

Table 15: Graph retrieval performance of the hybrid baseline in different models across five settings.

Baselines	Single-hop Setting				Multi-hop Setting				Set Setting				Aggregation Setting			
	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1
RoG	57.05	58.15	61.12	56.59	16.78	17.59	19.58	15.38	27.94	31.21	50.75	34.50	35.72	36.75	39.59	33.72
RoG-SFT	68.54	69.74	73.00	67.60	17.10	18.29	20.75	16.32	31.14	34.81	58.00	37.00	34.01	34.66	36.36	32.26
RoG-subset	58.74	60.05	62.63	57.67	16.45	17.35	19.11	15.62	28.02	31.05	51.25	34.75	33.50	34.85	35.19	31.67
RoG-general	57.34	58.10	61.99	56.80	15.97	16.81	18.64	15.15	29.12	32.85	55.75	35.78	33.92	34.96	36.07	31.96

Table 16: Performance of different RoG versions across four general settings.

Baselines	Editing Setting				Answerability Setting			
	MacF1	MicF1	Hit	Hit@1	Acc	Precision	Recall	F1
RoG	8.19	9.25	11.51	7.19	46.41	61.40	37.53	46.59
RoG-SFT	8.82	10.13	12.59	8.27	48.75	63.10	42.63	50.88
RoG-subset	8.32	9.53	11.87	7.91	46.74	62.26	41.23	49.61
RoG-general	8.67	9.75	12.23	7.91	47.25	62.90	42.76	50.91

Table 17: Performance of different RoG versions across two special settings.

Baselines	Single-hop Setting			Multi-hop Setting			Set Setting			Aggregation Setting			Editing Setting		
	Recall	Acc	Num	Recall	Acc	Num	Recall	Acc	Num	Recall	Acc	Num	Recall	Acc	Num
RoG	38.17	62.42	58.06	13.36	24.01	41.33	12.31	30.75	77.47	12.07	56.89	76.50	14.56	23.38	41.71
RoG-SFT	43.83	70.41	19.26	12.11	25.87	48.10	13.10	34.50	124.64	9.56	48.68	58.32	11.07	31.65	45.66
RoG-subset	42.05	65.87	20.90	12.47	24.24	43.96	12.27	32.75	101.94	9.23	47.50	59.94	10.45	26.25	42.37
RoG-general	41.21	64.57	20.15	11.88	22.84	46.15	13.08	33.50	111.40	9.41	48.09	63.21	10.88	26.98	48.74

Table 18: Evaluation of graph retrieval in different RoG versions across five settings.

Path Lengths	1				2				3				4			
	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1
Multi-hop	-	-	-	-	28.78	32.38	41.41	24.67	23.47	26.09	30.83	22.50	17.23	19.35	24.05	16.46
Set	79.96	82.39	92.00	92.00	38.95	42.43	69.70	42.42	41.04	44.88	67.96	41.32	40.90	45.78	87.50	62.50
Aggregation	69.61	70.47	73.53	64.71	47.68	49.42	54.47	42.28	42.90	44.02	47.54	42.62	-	-	-	-

Table 19: Ablation study of dense retriever on reasoning path lengths.

Path Lengths	1				2				3				4			
	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1
Multi-hop	-	-	-	-	15.07	15.81	17.18	14.54	15.45	16.17	17.50	15.00	12.45	12.78	13.92	11.39
Set	80.13	81.61	88.00	84.00	33.60	37.13	57.58	36.36	30.92	33.77	51.20	34.43	48.53	53.94	75.00	37.50
Aggregation	66.18	67.99	73.53	61.76	33.57	34.63	37.40	33.33	32.79	33.15	34.43	31.15	-	-	-	-

Table 20: Ablation study of sparse retriever on reasoning path lengths.

Path Lengths	1				2				3				4			
	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1
Multi-hop	-	-	-	-	22.08	23.46	26.43	21.59	25.16	26.51	29.17	25.00	15.14	16.15	18.99	13.92
Set	79.43	82.11	96.00	96.00	32.04	38.08	66.67	27.27	34.85	39.19	68.86	46.41	13.11	16.76	37.50	25.00
Aggregation	57.84	58.68	61.76	55.88	39.60	40.50	42.68	40.24	24.86	25.24	26.23	22.95	-	-	-	-

Table 21: Ablation study of G-Retriever on reasoning path lengths.

Entities Count	3				4				≥ 5			
	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1
Single-hop	70.38	71.96	75.66	68.14	61.95	63.10	66.36	57.94	72.53	74.18	78.46	67.69
Multi-hop	26.15	29.52	37.81	23.32	23.56	26.31	31.53	21.62	22.72	23.66	25.71	20.00
Set	44.91	48.84	70.95	46.96	41.93	45.99	70.24	44.05	25.22	27.06	55.00	20.00
Aggregation	49.26	50.80	55.23	44.77	52.45	54.19	59.09	49.24	35.59	36.59	40.54	27.03

Table 22: Ablation study of dense retriever on query entities count.

Entities Count	3				4				≥ 5			
	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1
Single-hop	62.67	63.76	66.37	62.39	68.13	69.36	72.90	64.49	66.68	68.11	71.54	66.92
Multi-hop	15.91	16.59	18.02	15.19	14.30	15.06	16.21	14.41	7.62	7.79	8.57	5.71
Set	35.68	38.71	57.09	39.19	34.69	36.76	50.00	35.71	17.67	22.72	35.00	25.00
Aggregation	37.08	38.45	42.44	35.47	37.87	38.53	40.15	38.64	30.63	31.02	32.43	27.03

Table 23: Ablation study of sparse retriever on query entities count.

Entities Count	3				4				≥ 5			
	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1	MacF1	MicF1	Hit	Hit@1
Single-hop	54.17	55.16	57.52	54.42	56.04	57.04	59.81	53.27	59.93	59.88	62.31	57.69
Multi-hop	22.19	23.51	26.50	21.20	21.01	22.50	25.23	22.52	20.48	21.06	22.86	17.14
Set	38.66	43.26	72.30	47.39	33.82	37.44	64.29	51.19	25.09	30.87	55.00	35.00
Aggregation	37.54	38.41	40.70	37.21	42.19	43.10	45.45	42.42	32.43	32.43	32.43	32.43

Table 24: Ablation study of G-Retriever on query entities count.

D.2 Formalization of Tree Sampling under Different Settings

In order to facilitate the understanding of tree sampling algorithms under different settings, we provide formalization in this section.

Single-hop Setting We first perform C-BFS starting from node $v_a \in V$, setting $D_{\text{max}} = 1$:

$$G_{\mathcal{T}_{v_a}} = \text{C-BFS}(v_a, G, D_{\text{max}} = 1, W_{\text{max}}). \quad (5)$$

Then we randomly select n non-root nodes as query nodes, that is $V_q = \{v_{q_1}, v_{q_2}, \ldots, v_{q_n}\} \subseteq V_{\mathcal{T}_{v_a}} \backslash \{v_a\}$. To ensure the uniqueness of the answer, we check all common neighbors of the query nodes to ensure no other node shares the same relation. Formally, let $\mathcal{R}(u, v)$ represent the relation between nodes u and v in the graph G. To ensure that no other node shares the same relation with the query nodes, we define the condition as follows:

$$\forall v \in \mathcal{N}_q, \exists v_{q_i} \in V_q, \mathcal{R}(v, v_{q_i}) \neq \mathcal{R}(v_a, v_{q_i}), \quad (6)$$

where $\mathcal{N}_q = \cap_{i=1}^{n} \mathcal{N}_{v_{q_i}}$ denotes the common neighbors of query nodes.

Multi-hop Setting We first perform C-BFS starting from node $v_a \in V$, setting $D_{\text{max}} \geq 2$:

$$G_{\mathcal{T}_{v_a}} = \text{C-BFS}(v_a, G, D_{\text{max}} \geq 2, W_{\text{max}}). \quad (7)$$

Then we randomly select n non-root nodes as query nodes, that is $V_q = \{v_{q_1}, v_{q_2}, \ldots, v_{q_n}\} \subseteq V_{\mathcal{T}_{v_a}} \backslash \{v_a\}$. We check the children nodes (level 1 nodes) of the answer node to ensure its uniqueness. Specifically, let v_a be the answer node, and $\mathcal{C}(v)$ denote the set of children nodes (level 1 nodes) of node v. Specifically, the parent nodes of the answer node v_a are those nodes that are directly connected to v_a at depth 1 in the tree $G_{\mathcal{T}_{v_a}}$. We define the condition as follows:

$$\forall v \in \mathcal{N}_{c_v}, \exists v_c \in \mathcal{C}(v), \mathcal{R}(v, v_c) \neq \mathcal{R}(v_a, v_c), \quad (8)$$

where $\mathcal{N}_{c_v} = \cap_{v_c \in C(v)} \mathcal{N}_{v_c}$ denotes the common neighbors of the children nodes.

Set Setting We first adopt the same searching approach as the single-hop setting:

$$G_{\mathcal{T}_v} = \text{C-BFS}(v, G, D_{\text{max}} = 1, W_{\text{max}}). \quad (9)$$

To obtain the answer node set, we set relation consistency constraints to ensure the answer nodes share some similar properties:

$$\begin{aligned} V_a = \{v_{a_1}, v_{a_2}, \ldots, v_{a_n}\} \subseteq V_{\mathcal{T}_v} \backslash \{v\}, \\ \forall v_{a_i}, v_{a_j} \in V_a, \mathcal{R}(v_{a_i}, v) = R(v_{a_j}, v). \end{aligned} \quad (10)$$

Algorithm 1 The pipeline of data construction

Input: Knowledge graph $G = (V, E)$, construction iterations T
Output: Constructed data $D = \{(q, V_q, v_a, \mathcal{P})\}$, including query q, query nodes V_q, answer nodes V_a, and reasoning paths $\mathcal{P}$

$D \leftarrow \{\}$
for $i = 1$ **to** T **do**
  $v_a \leftarrow \text{Sample}(V, 1)$ # Sample one node as the answer
  $G_{\mathcal{T}_{v_a}} \leftarrow \text{C-BFS}(v_a, G, D_{\max}, W_{\max})$, where $G_{\mathcal{T}_{v_a}} = (V_{\mathcal{T}_{v_a}}, E_{\mathcal{T}_{v_a}})$
  $V_q \leftarrow \text{Sample}(V_{\mathcal{T}_{v_a}} \backslash \{v_a\}, k)$ # Sample query nodes
  $\mathcal{P} \leftarrow \{\}$
  for $v_q \in V_q$ **do**
    $p_{v_q, v_a} \leftarrow \text{BackTrack}(v_q, v_a, G_{\mathcal{T}_{v_a}})$
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{p_{v_q, v_a}\}$
  end for
  $q \leftarrow \text{LLM}_\text{c}(V_q, v_a, \mathcal{P})$ # Query creation
  $q \leftarrow \text{LLM}_\text{r}(q, V_q)$ # Query refinement
  $V_q \leftarrow \text{EntityMatching}(q, V_q)$
  $\mathcal{P} \leftarrow \{\}$
  for $v_q \in V_q$ **do**
    $p_{v_q, v_a} \leftarrow \text{BackTrack}(v_q, v_a, G_{\mathcal{T}_{v_a}})$
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{p_{v_q, v_a}\}$
  end for
  $D \leftarrow D \cup \{(q, V_q, v_a, \mathcal{P})\}$
end for
$D \leftarrow \text{Filtering}(D)$
return D

Finally, we use the common neighbors of the answer nodes as the starting points for multi-source BFS to ensure the connectivity of the search tree:

$$G_{\mathcal{T}_A} = \text{Multi-source BFS}(\mathcal{S}, G, D_{\max}, W_{\max}), \tag{11}$$

where $\mathcal{S} = \bigcap_{v_a \in V_a} \mathcal{N}_{v_a}$ denotes the common neighbors.

Aggregation Setting We first obtain the node set $V_a = \{v_{a_1}, v_{a_2}, \ldots, v_{a_n}\}$ and search tree $G_{\mathcal{T}_A} = (V_{\mathcal{T}_A}, E_{\mathcal{T}_A})$ using the same strategy as the set setting. Then we construct the final answer node using two strategies:

For the first strategy, we identify common relations shared by all answer nodes and find neighboring nodes connected.

$$\exists r, s.t. \forall v_a \in V_a, \exists v \in \mathcal{N}_{v_a}, \mathcal{R}(v_a, v) = r. \tag{12}$$

Then we add the newly retrieved nodes and edges to original search tree $G_{\mathcal{T}_A}$ and obtain $G'_{\mathcal{T}_A}$:

$$\begin{aligned} E'_{\mathcal{T}_A} &= E_{\mathcal{T}_A} \cup \{(v, r, v_a) | v_a \in V_a, v \in \mathcal{N}_{v_a}\}, \\ V'_{\mathcal{T}_A} &= V_{\mathcal{T}_A} \cup \{v | v \in \mathcal{N}_{v_a}, v_a \in V_a, \mathcal{R}(v_a, v) = r\}, \\ G'_{\mathcal{T}_A} &= (V'_{\mathcal{T}_A}, E'_{\mathcal{T}_A}). \end{aligned} \tag{13}$$

For the second strategy, we construct the edge set E by taking all the edges adjacent to v_a:

$$E = \{(v_a, r, v) \mid v \in \mathcal{N}_{v_a}\}. \tag{14}$$

For each edge (v_a, r, v) in E, iterate over all other nodes $v' \in V_a$ (i.e., $v' \neq v_a$) and check whether the edge (v', r, v) exists. If such an edge exists, remove the edge (v_a, r, v) from E.This condition can be formally expressed as:

$$E' = \{(v_a, r, v) \in E \mid \nexists v' \in V_a,\ v' \neq v_a\}. \tag{15}$$

Finally, we randomly select one edge (v_a, r, v) from the remaining edge set E' and add the edge to the search tree $G_{\mathcal{T}_A} = (V_{\mathcal{T}_A}, E_{\mathcal{T}_A})$.

Editing Setting First, we modify the graph G according to the specified editing method:

$$G' = \text{edit}(G). \tag{16}$$

Next, we construct the tree based on the tree sampling method used in the single-hop or multi-hop settings.

Answerability Setting We first construct the reasoning tree, query and nodes using the same strategy as the single-hop or multi-hop settings. Then we sample some of the data and modify the graph structure G:

$$\begin{aligned} V' &= V \backslash \{v_a\}, \\ E' &= \{(u, r, v) \in E \mid u \neq v_a, v \neq v_a\}, \\ G' &= (V', E'). \end{aligned} \tag{17}$$

E Examples

Examples in six settings of our constructed dataset M^{3}GQA can be found in Table 25 and 26.

F Prompts

Prompts used in our paper are listed below.

Table 25: Cases of data in four general settings.

	Question	Answer	Reasoning Paths
Single-hop	Which film production company produced the movies {The Pink Panther}, {A League of Their Own}, and {Awakenings}?	Columbia Pictures	[A League of Their Own, film.film.production_companies, Columbia Pictures], [The Pink Panther, film.film.production_companies, Columbia Pictures], [Awakenings, film.film.production_companies, Columbia Pictures]
Multi-hop	Which river, flowing through {Myanmar} and near {Phou Khe}, was partially affected by {Typhoon Mirinae} in {Asia}?	Mekong	["Laos", "location.location.partially_contains", "Mekong"],[Vietnam, location.location.partially_contains, Mekong],[Phou Khe, location.location.partially_containedby, Laos],[Typhoon Mirinae, meteorology.tropical_cyclone.affected_areas, Vietnam],[Myanmar, location.location.partially_contains, Mekong],[Asia, base.locations.continents.countries_within, Vietnam]
Set	What languages are spoken in a country with a {Presidential system}, located in the same region as where {Juba Arabic} and {Algerian Arabic} are spoken?	Swahili Language, English Language, Arabic Language	[Tanzania, location.country.languages_spoken, English Language],[Africa, base.locations.continents.countries_within, Tanzania],[Tanzania, location.country.languages_spoken, Swahili Language],[Tanzania, location.country.languages_spoken, Arabic Language],[Algerian Arabic, language.human_language.region, Africa],[Presidential system, government.form_of_government.countries, Tanzania],[Juba Arabic, language.human_language.region, Africa]
Aggregation	Which female literary figure influenced both {Sherwood Anderson} and {Ralph Ellison}, and subsequently impacted the works of {Ray Bradbury}, {William Faulkner}, {John Steinbeck}, and {Thomas Wolfe}?	Mark Twain, Gertrude Stein, Walt Whitman	[Sherwood Anderson, influence.influence_node.influenced_by, Gertrude Stein],[Sherwood Anderson, influence.influence_node.influenced_by, Walt Whitman],[Ralph Ellison, influence.influence_node.influenced_by, Mark Twain],[Thomas Wolfe, influence.influence_node.influenced_by, Sherwood Anderson],[Walt Whitman, people.person.gender, Male],[Gertrude Stein, people.person.gender, Female],[Ralph Ellison, influence.influence_node.influenced_by, Gertrude Stein],[William Faulkner, influence.influence_node.influenced_by, Sherwood Anderson],[Ralph Ellison, influence.influence_node.influenced_by, Walt Whitman],[John Steinbeck, influence.influence_node.influenced_by, Sherwood Anderson],[Sherwood Anderson, influence.influence_node.influenced_by, Mark Twain],[Ray Bradbury, influence.influence_node.influenced_by, Sherwood Anderson],[Mark Twain, people.person.gender, Male]

Table 26: Cases of data in two special settings.

	Question	Answer	Reasoning Paths
Editing	Who is a notable person born in the region that includes {Alameda County}, served by the {San Francisco Chronicle}, and covered by {Cairn Marketing}?	Spike Lee	[San Francisco Chronicle, book.newspaper.circulation_areas, San Francisco Bay Area],[San Francisco Bay Area, location.location.containedby, Northern California],[Alameda County, location.location.containedby, San Francisco Bay Area],[Northern California, location.location.people_born_here, 'Spike Lee],[Cairn Marketing, organization.organization.geographic_scope, San Francisco Bay Area]
Answerability	Which {Taylor Swift} concert tour featured performances of {"The Lucky One"}, {"Shake It Off"}, and {"If This Was a Movie"}?	No	None

The prompt for query generation.

Please help me design a multi-entity (multi-hop) reasoning question. The question must include at least 3 of the following entities: {entities}

And the answer of this question must be: {answer entity}

You can create the question according to the following knowledge graph. The question must be natural and close to reality

The question needs to include the relationships in the knowledge graph. Just need to output the question, no need to output additional explanatory statements

Knowledge graph: {reasoning paths}

Question:

The prompt for query paraphrase.

Please polish the following question to make it more natural and more close to reality.
Just need to output the question, no need to output additional explanatory statements.

Question:

The prompt for the first strategy of aggregation setting.

I will give you a question and its answer. I am now providing you with an additional edge related to the answer on the knowledge graph. Please help me modify the original question into a question involving further reasoning according to the additional edge, that is, further reasoning on the answer set.

For example, the original question is, which scientists in the field of deep learning have won the Turing Award? The answer if Hinton, Bengio, Lecun. We can modify it into a question for further reasoning about the answer: Who among the Turing Award winning scientists in the field of deep learning graduated from the University of Cambridge?

You should only output the modified question, no need to output an additional statement. If rewriting is not possible, output "No"

Original question: {original question}

The answer of the original question: {answer entities}

The additional edge related to the answer: {constrained edges}

The answer to the modified question must be: {final answer}

Modified question:

The prompt for the first strategy of aggregation setting.

I will give you a question and its answer. I am now providing you with the answer set of all neighboring nodes on the knowledge graph. Please help me modify it into a question involving further reasoning, that is, further reasoning on the answer set. Additionally, you also need to provide the answer of modified question to me.

The output format is a binary (question, answer), where the first item represents the modified question and the second item represents the answer to the question. If the answer is a set, please write the answer part in the form of a list, such as (question, [a1, a2, a3, a4]).

If rewriting is not possible, output "No"

For example, the original question is, which famous rivers pass through China? The answer is Yangtze River, Yellow River, the Yarlung Zangbo River, etc. We can transform it into a question for further reasoning about the answer: ("Among all the famous rivers that pass through China, which one is the longest?", "Yangtze River")
For example, the original question is, which scientists in the field of deep learning have won the Turing Award? The answer if Hinton, Bengio, Lecun. We can modify it into a question for further reasoning about the answer: ("Who among the Turing Award winning scientists in the field of deep learning graduated from the University of Cambridge?", "Hinton")

The answer to the modified question can also be a collection. For example: ("which deep learning scientists who have won the Turing Award are Canadian?", "[Hinton, Bengio]")

Please strictly follow the form: (question, answer)! No need to output additional explanatory statements.

Original question: {original question}

The answer of the original question: {answer entities}

Edges and neighbors related to the answer: {constrained edges}

Modified question:

The prompt for replacing the answer node in editing setting.

Give you an entity: {original answer entity}, please provide another entity that belongs to the same category as it.

You should only output the entity, no need to output additional explanatory statements.

New entity:

The prompt for LLM-based filtering.

Given a question and an answer, please use the given knowledge graph to determine if the answer is correct.

If the answer is correct, please output 'yes', otherwise, output 'no'. No need to output additional statement.

Question: {question}

Answer: {answer}

Knowledge graph: {reasoning paths}

Judgement:

The prompt for answer generation of LLM-only methods.

Given a question, please use one or more entities to answer it.

Please directly output the entity (entities) that meet(s) the requirements, without the need to output additional statements.

If the answer is multiple entities, please separate them with commas.

Question: {question}

Answer:

The prompt for answer generation of other baselines.

Given a question and a relevant knowledge graph, please use one or more entities to answer the question.

Please directly output the entity (entities) that meet(s) the requirements, without the need to output additional statements.

If the answer is multiple entities, please separate them with commas.

Question: {question}

Knowledge graph: {knowledge graph}

Answer:

The prompt for answer generation of answerability setting.

Given a question and a relevant knowledge graph, please check whether the question is answerable according to the knowledge graph.

If the question is answerable, please output 'yes', otherwise output 'no', without the need to output additional statements.

Question: {question}

Knowledge graph: {knowledge graph}

Answer: