Recursive Short-to-Long Generalization for Multi-hop Reasoning

Mayi Xu
School of Computer Science, Wuhan University
Wuhan, China
xumayi@whu.edu.cn

Ke Sun
School of Computer Science and Technology, Wuhan University of Science and Technology
Wuhan, China
sunke@wust.edu.cn

Jianhao Chen
School of Computer Science, Wuhan University
Wuhan, China
Zhongguancun Academy
Beijing, China
chenjianhao@whu.edu.cn

Qiankun Pi
School of Computer Science, Wuhan University
Wuhan, China
piqiankun@whu.edu.cn

Guixin Su
School of Computer Science, Wuhan University
Wuhan, China
cometsue@whu.edu.cn

Yunfeng Ning
School of Computer Science, Wuhan University
Wuhan, China
ningyunfeng@whu.edu.cn

Yongqi Li
School of Computer Science, Wuhan University
Wuhan, China
liyongqi@whu.edu.cn

Yuanyuan Zhu
School of Computer Science, Wuhan University
Wuhan, China
yyzhu@whu.edu.cn

Ming Zhong
School of Computer Science, Wuhan University
Wuhan, China
clock@whu.edu.cn

Jiawei Jiang
School of Computer Science, Wuhan University
Wuhan, China
blue.jwjiang@gmail.com

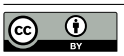
Tieyun Qian*
School of Computer Science, Wuhan University
Wuhan, China
Zhongguancun Academy
Beijing, China
qty@whu.edu.cn

Abstract

Reasoning is fundamental to human intelligence and critical for problem-solving. In practice, answering complex questions often requires abundant reasoning steps involving long reasoning paths. Existing multi-hop reasoning methods mainly focus on short-hop questions with only short reasoning paths, whose distribution is different from that of long paths. While training on long-hop data can improve corresponding performance, collecting it at scale is difficult and expensive. Hence, we explore the *short-to-long generalization scenario* for the first time, which aims to enhance the model's capability to handle long-hop questions using easily collected short-hop reasoning data. In this scenario, the reasoning model can only learn the features of short-hop questions due to the lack of long-hop data, and thus two challenges emerge. (1) *Struggling to determine where to go next*: as the number of steps increases, the model needs to reason about the next step based on a long context composed of more evidences. This reasoning process will confuse the reasoning model, which can only learn the pattern from a short context to the next step. (2) *Hard to determine when to stop reasoning*: too many reasoning hops may introduce excessive noise, while too few may fail to gather sufficient supporting information. The path lengths of long-hop questions vary significantly. Thus, a fixed stopping threshold, commonly used for short-hop questions with similar path lengths, cannot handle them effectively.

Inspired by the classic recursive algorithm, we propose a novel *Recursion-based Short-to-Long Generalization* (RSLG) reasoning framework, which recursively decomposes long-hop questions into multiple short-hop questions that can be handled by a short-hop reasoning model. In this way, the above two challenges are transformed into *how to decompose* and *when to stop decomposition*. For *how to decompose*, we introduce the parallel and sequential recursion patterns to guide the recursive decomposition processes. A local-to-global recursive demonstration construction strategy is proposed to obtain the corresponding recursive demonstrations from the perspective of certainty, complexity, and diversity, respectively. For *when to stop decomposition*, we propose three recursive termination criteria in view of decomposability, redundancy, and relevance. Extensive experiments on six short-to-long settings across diverse domains demonstrate RSLG's superior performance, robustness, and efficiency in this challenging scenario.[1]

*Corresponding author.



SIGIR '26, Melbourne, VIC, Australia

ACM ISBN 979-8-4007-2599-9/2026/07
https://doi.org/10.1145/3805712.3809697

[1]The code and data link: https://github.com/NLPGM/RSLG

CCS Concepts

• **Computing methodologies** → **Natural language processing**;
• **Information systems** → **Question answering**.

Keywords

Question answering, Multi-hop reasoning, Short-to-long generalization, Recursive decomposition.

ACM Reference Format:
Mayi Xu, Ke Sun, Jianhao Chen, Qiankun Pi, Guixin Su, Yunfeng Ning, Yongqi Li, Yuanyuan Zhu, Ming Zhong, Jiawei Jiang, and Tieyun Qian. 2026. Recursive Short-to-Long Generalization for Multi-hop Reasoning. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '26), July 20–24, 2026, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3805712.3809697

1 Introduction

Reasoning refers to drawing new conclusions from existing knowledge [1]. Reasoning capability is essential for solving complex tasks, such as problem-solving, decision-making, and critical thinking, which is fundamental to human intelligence. The multi-hop reasoning process involves using multiple evidences to draw conclusions or make judgments [2]. Robust multi-hop reasoning capability is essential for various applications like clinical diagnosis [3–5], education [6–8], legal service [9, 10], and financial analysis [11, 12].

In practice, there are many complex multi-hop questions [13], and answering them often requires abundant reasoning steps involving long reasoning paths. As shown in Fig. 1 (I), existing multi-hop reasoning methods [14–17] mainly focus on **short**-path multi-**hop** (**short-hop**) reasoning where a question only requires several reasoning steps, typically concentrating on 2-hop. To enhance the model's capability to handle **long**-path multi-**hop** (**long-hop**) questions, i.e., more than 2-hop, we can collect some specialized long-hop reasoning data to train a long-hop reasoning model as illustrated in Fig. 1 (II). However, collecting and annotating such long-hop reasoning data at scale is challenging and costly.

Inspired by the easy-to-hard generalization capability of humans, we explore the *short-to-long generalization scenario* for the first time. As shown in Fig. 1 (III), this scenario explores enhancing the model's capability to handle long-hop questions using easily collected short-hop data, just as humans acquire the ability to solve complex problems through the study of simpler ones. Generally, humans can solve problems that are more challenging than those they have previously learned about, which can be attributed to their excellent easy-to-hard generalization skill [18]. In contrast, current artificial intelligence systems are typically limited to solving problems with the same difficulty as the training data. Hence, our proposed *short-to-long generalization scenario* is worthy of exploration for the complex long-hop reasoning problem.

In the *short-to-long generalization scenario*, the reasoning model can only learn the features of short-hop questions due to the lack of long-hop data. Hence, there are two challenges when reasoning for long-hop questions. (1) *Struggling to determine where to go next*: as the number of reasoning steps increases, the model needs to determine the next step based on a long context composed of more

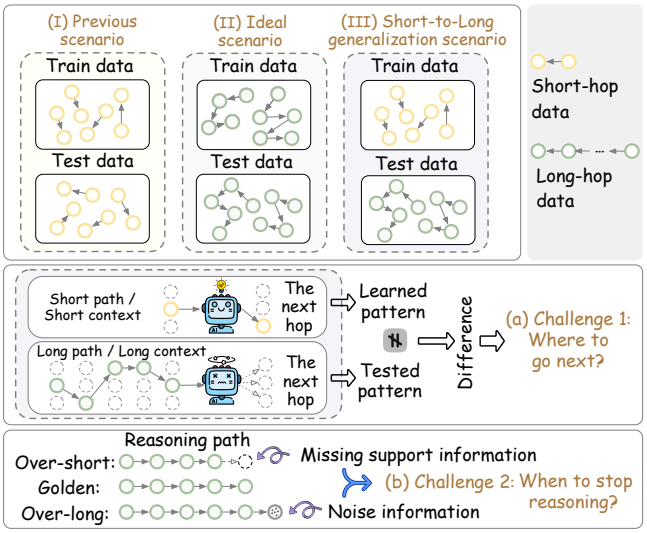


Figure 1: The multi-hop reasoning scenarios and challenges.

evidences. As shown in Fig. 1 (a), this process will confuse the reasoning model, which only learns the pattern from a short context to the next step. (2) *Hard to determine when to stop reasoning*: precisely stopping reasoning is crucial for accurate reasoning. As shown in Fig. 1 (b), too many reasoning hops may introduce excessive noisy information, whereas too few may fail to gather sufficient supporting information. The long-hop questions and short-hop questions differ significantly in terms of stopping points. Since the path length of the short-hop questions is primarily concentrated on two hops [19–21], a fixed and small threshold is generally set to control the stopping point [15–17, 22, 23]. However, the path length distribution in long-hop questions varies significantly. For example, in the long-hop reasoning dataset FanoutQA [13], the path lengths range from 5 to 46 hops. Therefore, the fixed threshold may fail.

In cognitive science, extensive works [24–26] have demonstrated that humans achieve easy-to-hard generalization capabilities by decomposing complex problems into simpler ones. This skill also contributes to addressing the aforementioned challenges, as we can decompose a long-hop question into multiple short-hop questions, allowing the short-hop reasoning model to easily determine *where to go* and *when to stop*. Generally, such a decomposition process can be achieved by Large Language Models (LLMs) through in-context learning. Existing LLM-based question decomposition methods [18, 27] rely heavily on detailed decomposition demonstrations, which share close decomposition steps with the target questions. However, in the *short-to-long generalization scenario*, the distribution of decomposition steps for target questions is diverse and hard to predetermine. Furthermore, short-hop data cannot be directly used to create decomposition demonstrations tailored to questions that require different decomposition steps. Therefore, directly decomposing long-hop questions is challenging.

To tackle these problems, we propose a **R**ecursion-based **S**hort-to-**L**ong **G**eneralization (**RSLG**) reasoning framework to recursively decompose the long-hop questions into multiple short-hop questions. Drawing inspiration from classic recursive algorithms, RSLG

develops recursive functions and recursive termination criteria to guide the decomposition processes. The recursive functions are used to control *how to decompose* the long-hop questions, while the termination criteria determine *when to stop decomposition*. By imitating the recursive algorithm, the aforementioned two challenges are collectively transformed into *how to decompose* and *when to stop decomposition*.

To address the challenge of *how to decompose*, we propose two recursive functions corresponding to the parallel and sequential recursion patterns, as each decomposition step of a question follows one of the two patterns. By constructing decomposition demonstrations based on these patterns, we do not need to create separate demonstrations for questions involving varying numbers of decomposition steps. To obtain high-quality recursive decomposition demonstrations, we propose a local-to-global recursive demonstration construction strategy, which optimizes demonstrations from the perspectives of certainty, complexity, and diversity, respectively.

To address the challenge of *when to stop decomposition*, we propose three recursive termination criteria to collaboratively avoid under-decomposition and over-decomposition. These criteria assess decomposability, redundancy, and relevance to ensure an appropriate stopping point. The recursive decomposition processes terminate when any of the criteria is met. By integrating these termination criteria with two recursion functions, we can effectively decompose a long-hop question into multiple short-hop questions, which can be handled by a short-hop reasoning model.

In summary, our main contributions are as follows.

- We make the first attempt to explore the *short-to-long generalization scenario*, which aims to enhance the model's capability to handle long-hop questions using easily collected short-hop reasoning data.
- We propose a novel Recursion-based Short-to-Long Generalization reasoning framework, which recursively decomposes long-hop questions into multiple short-hop questions that can be easily handled by a short-hop reasoning model.
- Extensive experiments on six short-to-long settings across different domains demonstrate the superior performance, robustness, and efficiency of our proposed RSLG.

2 Related Works

Multi-hop reasoning tasks mainly include Knowledge Graph Question Answering (KGQA) [28], Multi-hop Question Answering (MHQA) [16], and Multi-document Question Answering (MDQA) [17].

KGQA methods aim to reason about the answer entity from knowledge graphs (KGs) [29]. Their multi-hop reasoning paths are composed of entities and relations in the KGs. Existing KGQA methods [22, 30–34] primarily focus on questions that only require short-hop entity-relation paths within KGs. MHQA methods focus on reasoning across multiple passages to answer knowledge-intensive questions. Existing MHQA methods [14–16, 35] are mainly designed to perform short-hop reasoning across several passages.

MDQA is an emerging task that aims to reason and retrieve the supporting information across multiple entire documents. MDQA requires a comprehensive understanding of the logical relationships between the contents and structures of entire documents, which is more challenging than KGQA and MHQA. Recently, KGP [17] is first proposed for this task. It first splits the documents into multiple paragraphs and then creates a graph structure to connect these paragraphs. Then, it designs an LLM-based graph traversal agent that navigates across nodes and gathers supporting paragraphs. Based on KGP, CuriousLLM [36] tries to merge the semantic gap between current support information and the next one by generating an intermediate question. However, it requires a large amount of additional customized data (50K) for model training. Note that these MDQA methods are tailored to facilitate short-hop reasoning.

In summary, existing multi-hop reasoning methods primarily focus on short-hop reasoning, which requires only a few reasoning steps. In practice, many complex questions require many reasoning steps involving long-hop reasoning paths. In this paper, we explore the *short-to-long generalization scenario*, aiming to improve long-hop reasoning capability using easily collected short-hop data.

3 Methods

This section introduces our Recursion-based Short-to-Long Generalization (RSLG) reasoning framework. Compared to the refined fact triplets in KGQA and the concise passages in MHQA, entire documents in MDQA are substantially longer and contain significantly more distracting information. The setting of MDQA more closely reflects real-world applications and poses greater challenges for multi-hop reasoning. Therefore, we select MDQA as a representative task to investigate the *short-to-long generalization scenario*.

3.1 Problem Definition

The definition of MDQA can be formulated as follows: given a question q and a collection of n documents $D = \{d_1, d_2, \ldots, d_n\}$, where the j-th document d_j consists of k_j paragraphs: $\{p_j^1, p_j^2, \ldots, p_j^{k_j}\}$. Through multi-hop reasoning, MDQA methods aim to reason and retrieve a set of supporting paragraphs that provide evidences for answering questions. Formally, let $P = \bigcup_{j=1}^{n} \{p_j^1, p_j^2, \ldots, p_j^{k_j}\}$ be the set of all paragraphs, where the number of supporting paragraphs is significantly fewer than that of irrelevant paragraphs. Let $S(q, D) \subseteq P$ represent the retrieved supporting paragraphs, the final answer a is then generated by an LLM $\mathcal{P}$ as follows:

$$p(a \mid q, S(q, D)) = \prod_{i=1}^{|a|} \mathcal{P}\left(a^i \mid q,\ S(q, D),\ a^{<i}\right), \tag{1}$$

where a^i and $a^{<i}$ denote the i-th token of answer a, and the $i-1$ tokens before a^i, respectively. Our method aims to accurately identify supporting paragraphs, thereby maximizing the probability of generating an accurate answer.

3.2 Method Overview

Our method focuses on the *short-to-long generalization scenario*, which involves two key challenges: hard to determine *where to go next* and *when to stop*. To address them, we propose a novel Recursion-based Short-to-Long Generalization (RSLG) framework. It consists of graph construction, local-to-global recursive demonstration construction, and recursion-based decomposition modules. As shown in Fig. 2, our main contributions lie in the latter two

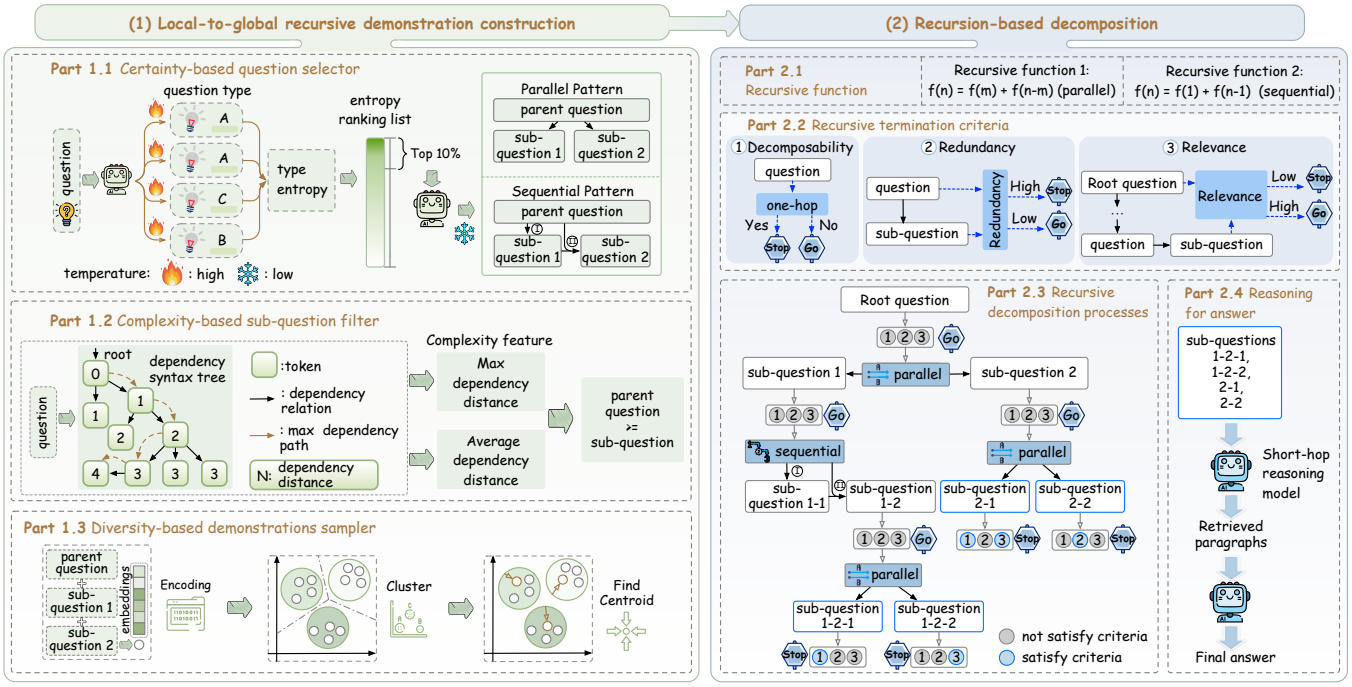


Figure 2: Our proposed Recursion-based Short-to-Long Generalization (RSLG) reasoning framework.

modules, which emulate recursive algorithms by integrating recursion functions and termination criteria to effectively resolve these challenges.

3.3 Graph Construction

In the MDQA setting, a collection consisting of many entire documents serves as an external knowledge base. Since the supporting information is distributed across different parts of multiple documents, it is common practice to construct a graph to support multi-hop reasoning. To ensure a fair comparison focusing on reasoning stage, we adopt the same document collection process and graph construction procedures as previous MDQA methods [17, 36].

Specifically, documents containing supporting information relevant to the question are gold documents. For each question, there are d distractor documents sampled from Wikipedia. When constructing the graph, each document is segmented into multiple paragraphs with a maximum length of l_{max}. For each paragraph, there exists a corresponding node v_i in the graph $\mathcal{G}$, with an attribute X_i that represents the text of the paragraph. Next, TagMe [37] is adopted to extract entities from each paragraph and establish edges between nodes that share common entities.

3.4 Local-to-global Recursive Demonstration Construction

In this section, we first analyze the underlying recursive decomposition patterns of multi-hop questions to imitate the recursive function. Then, we propose a local-to-global recursive demonstration construction strategy to obtain high-quality demonstrations for these patterns, as illustrated in part (1) of Fig. 2.

From a dependency perspective, the relationships between two questions are either independent or interdependent. For independent sub-questions, as illustrated in Fig. 3 (a), their parent question can be decomposed directly. This forms a parallel recursion pattern, and we refer to such a parent question as a parallel question. In contrast, for interdependent sub-questions, as shown in Fig. 3 (b), decomposing their parent question requires answering the first sub-question before generating the second. This results in a sequential recursion pattern, and we refer to such a parent question as a sequential question.

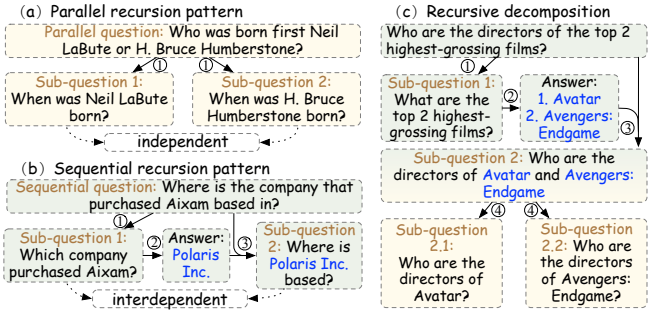


Figure 3: Cases of recursive decomposition patterns.

For a long-hop question, as depicted in Fig. 3 (c), recursive decomposition can involve both parallel and sequential patterns at different steps. The dynamic combination of the two foundational patterns in a recursive way allows extensive flexibility for solving complex problems. For instance, it yields up to $2^9 = 512$ recursion

combinations for a 10-hop question. Therefore, to guide the recursive decomposition process, both parallel and sequential recursion patterns should be incorporated to serve as recursive functions.

Ideally, the two patterns can be achieved through the powerful in-context learning capabilities of LLMs without requiring a specialized model. We randomly select 2-hop questions from short-hop training set and prompt LLMs to annotate four parallel demonstrations M_{par}, four sequential demonstrations M_{seq}, and four one-hop demonstrations M_{one}. Each parallel and sequential demonstration contains a parent question along with its decomposition process, i.e., its sub-questions. One-hop questions cannot be further decomposed and can be answered directly. Note that four is not a special choice but a widely-used demonstration number [18, 38, 39]. This annotation is achieved by LLMs using simple instructions grounded in the aforementioned definitions of parallel and sequential questions. However, ensuring the quality of demonstrations is challenging, as it requires both well-designed questions and precise decomposition processes to support high and robust performance.

Hence, to ensure effectiveness and robustness, we propose a local-to-global recursive demonstration construction strategy, which automatically constructs high-quality demonstrations by optimizing both the local and global parts of the demonstration. This strategy consists of a certainty-based question selector, a complexity-based sub-question filter, and a diversity-based demonstration sampler. Each global demonstration has two local parts, including a parent question and its sub-questions. For the two local components of a demonstration, the question selector is employed to identify high-quality questions with an unambiguous type, and the sub-question filter is used to eliminate unreasonable sub-questions whose complexity exceeds that of their parent questions. For the global demonstrations, the demonstration sampler is designed to sample demonstrations that emphasize diversity, making them more broadly applicable to a wide range of target questions.

Certainty-based question selector (Part 1.1 in Fig. 2). The types of questions in demonstrations must be clear, as the demonstrations guide the type identification and question decomposition of specific types. Hence, the certainty-based question selector aims to identify a set of high-quality questions, whose types are unambiguous among parallel, sequential, and undecomposable one-hop.

Generally, if LLMs repeatedly predict a question as the same type, it indicates they place greater certainty and confidence in the correctness of this type. Hence, to select questions with unambiguous types, we quantify type certainty by calculating the information entropy across their predicted types. The information entropy falls within $[0, 1]$, where lower entropy indicates higher certainty. Questions with high type certainty are subsequently reserved.

To guide the type prediction, we first merge M_{par}, M_{seq}, and M_{one} into a single set. All demonstrations in this set are randomly shuffled to create new classification demonstrations M_{cls}. These demonstrations guide the LLMs to classify questions into parallel, sequential, or one-hop ones in the training set. To quantify the type uncertainty of a question, we compute the normalized information entropy over the predicted types obtained from m runs under a high temperature 1. Using higher temperatures (e.g., 1) encourages more diverse results [40]. This increased variability is beneficial for assessing the certainty of question types, as it can make the content generated by LLMs more diverse. The entropy is calculated as:

$$\mathbb{H}(x^{(i)}|\{\hat{y}_j\}_{j=1}^{m}) = -\frac{\sum_{t=1}^{3} \tilde{\mathrm{p}}(\hat{y}_t) \log \tilde{\mathrm{p}}(\hat{y}_t)}{\log 3}, \tag{2}$$

where t is used to index over the unique question types $\{\hat{y}_1$: parallel, $\hat{y}_2$: sequential, $\hat{y}_3$: one-hop$\}$, $\tilde{\mathrm{p}}(\hat{y}_t)$ is the empirical frequency of unique type $\hat{y}_t$ in all m answers, and the number of prediction times m is set to 10. We select the top 10% of questions with the lowest entropy and filter out the rest. For the selected high-quality questions, we regenerate their decomposition processes using LLMs at a low temperature to reduce the instability caused by variability in the generation processes. Each decomposition process contains two sub-questions of their short-hop parent question.

Complexity-based sub-question filter (Part 1.2 in Fig. 2). A complete decomposition demonstration consists of a parent question and its sub-questions. Hence, after obtaining high-quality questions through the certainty-based question selector, acquiring their corresponding high-quality sub-questions is crucial to the decomposition demonstration. Generally, the complexity of sub-questions should be lower than that of their parent questions. For example, as shown in Fig. 4 (a), the LLMs decompose the parent question "Are Bocconia and Bellevalia both flowering plants?" into "What is the classification of Bocconia?" and "What is the classification of Bellevalia?". The two sub-questions expand the scope of the parent question, making them more complex than the parent question. As a result, they do not serve as effective demonstrations of question decomposition. In contrast, as shown in Fig. 4 (b), if the parent question is decomposed into "Are Bocconia flowering plants?" and "Are Bellevalia flowering plants?", the two sub-questions are simpler than the parent question and can effectively facilitate its resolution.

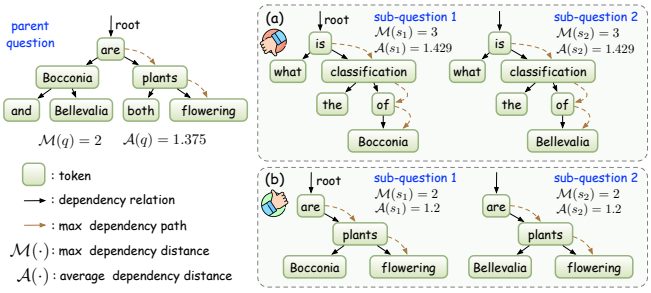


Figure 4: The maximum and average dependency distances.

We propose to assess the complexity of a question by analyzing its syntax features. Generally, the greater the complexity of a question is, the more complex the information it attempts to convey, and its syntax structure is typically more complex. In linguistics, a dependency syntax tree is a graphical representation used to illustrate the syntax structure of a sentence. In this tree, tokens in a sentence are connected by labeled edges that indicate dependency relationships between them. We employ both the maximum and average dependency distances derived from the dependency syntax tree as quantitative indicators of syntax complexity.

Specifically, we use the widely-used tool SpaCy[2] to obtain the dependency syntax tree of the question q. Let T_q denote the set

[2]Available at: https://spacy.io

of all tokens in q, and for each token $w \in T_q$, define $d_q(w)$ as the distance from a token w to the root node in the dependency tree of q. Then, the maximum dependency distance $\mathcal{M}(q)$ and average dependency distance $\mathcal{A}(q)$ can be formally written as:

$$\mathcal{M}(q) = \max_{w \in T_q} d_q(w), \tag{3}$$

$$\mathcal{A}(q) = \frac{1}{|T_q|} \sum_{w \in T_q} d_q(w). \tag{4}$$

If either the maximum or average dependency distances of sub-questions s_1 and s_2 exceed those of their parent question q, the sub-questions and their parent questions will be filtered out by the complexity-based sub-question filter $\mathcal{CF}(\cdot)$. The process can be defined as:

$$\mathcal{CF}(q) = \left\{ (q, s_1, s_2) \;\middle|\; \forall s \in \{s_1, s_2\} : \begin{array}{l} \mathcal{M}(q) \geq \mathcal{M}(s) \\ \wedge\, \mathcal{A}(q) \geq \mathcal{A}(s) \end{array} \right\}. \tag{5}$$

Diversity-based demonstration sampler (Part 1.3 in Fig. 2). Although there are only two decomposition patterns for multi-hop questions, the characteristics of questions sharing the same decomposition pattern may also differ significantly. Therefore, after obtaining high-quality questions and their corresponding sub-questions, it is essential to sample representative demonstrations. These demonstrations are used to construct recursive decomposition demonstrations with diversity, ensuring they can be more broadly applicable to different target questions.

Specifically, for each demonstration, we first use the widely-used ALL-MPNET-BASE-V2 model[3] in Sentence-Transformer to obtain its embedding. Second, for each type of demonstration, we use the K-means algorithm [41] to divide them into K clusters, respectively. The K is set to four, which is a widely-used demonstration number in previous prompting-based methods [18, 38, 39]. Third, for each cluster, we select the demonstration exhibiting the highest semantic similarity to its centroid as the representative demonstration, where the semantic similarity is measured by the cosine similarity.

Finally, we obtain high-quality parallel decomposition demonstrations D_{par}, sequential decomposition demonstrations D_{seq}, and one-hop demonstrations D_{one}. By merging and randomly shuffling these demonstrations, we derive demonstrations D_{cls}, which is used to classify questions as parallel, sequential, or one-hop.

3.5 Recursion-based Decomposition

As demonstrated in part (2) of Fig. 2, we propose a recursion-based decomposition strategy to construct a comprehensive recursive system in this section.

Given a long-hop question, this strategy recursively decomposes it into multiple short-hop sub-questions. The decomposition is guided by two recursive functions, corresponding to parallel and sequential recursion patterns. Then, we define three recursive termination criteria from the perspectives of decomposability, redundancy, and relevance, respectively. The recursive decomposition processes terminate when any of the criteria is met. Furthermore, a short-hop reasoning model is trained using short-hop reasoning data to address these short-hop sub-questions effectively.

Recursive functions (Part 2.1 in Fig. 2). As illustrated in Section 3.4, we automatically construct demonstrations based on short-hop reasoning data for the parallel and sequential decomposition patterns. These two patterns correspond to two recursive functions in the recursive algorithm, which can be formalized as:

$$f(n) = \begin{cases} f(m) + f(n-m), & \text{(Parallel)} \\ f(1) + f(n-1), & \text{(Sequential)} \end{cases} \tag{6}$$

where $f(n)$ represents the n-hop question. In the parallel decomposition pattern, $f(n)$ is decomposed into an m-hop sub-question $f(m)$ and an (n-m)-hop sub-question $f(n-m)$. In the sequential decomposition pattern, $f(n)$ is first decomposed into a one-hop sub-question $f(1)$. After obtaining the answer to $f(1)$, $f(n)$ is further decomposed into another (n-1)-hop sub-question $f(n-1)$.

Recursive termination criteria (Part 2.2 in Fig. 2). In addition to recursive functions, defining termination criteria is also essential for recursive algorithms. To effectively determine an appropriate stopping point, we propose three recursive termination criteria from the perspectives of decomposability, redundancy, and relevance. These criteria collaboratively avoid under-decomposition and over-decomposition, which are detailed below:

- **Decomposability termination criterion.** The decomposition processes should terminate when the LLM identifies the question as a one-hop question, indicating that it is adequately decomposed and cannot be further decomposed properly.
- **Redundancy termination criterion.** If the semantic similarity between a sub-question and its parent question is too high, i.e., higher than a redundancy *hyperparameter*, the sub-question is regarded as redundant. Thus, decomposition should be halted to prevent introducing noisy information.
- **Relevance termination criterion.** If the semantic similarity between a sub-question and the root question is too low, i.e., lower than a relevance *hyperparameter*, the sub-question is likely to drift off-topic from the root question, making it irrelevant. In such cases, decomposition should also be terminated to avoid introducing distracting information.

Following previous multi-hop reasoning works [16, 17, 36], these two hyperparameters are determined on short-hop validation set.

Recursive decomposition and reasoning (Part 2.3 & 2.4 in Fig. 2). Based on the recursive functions and recursive termination criteria, we decompose a long-hop question into multiple short-hop questions that can be handled by a short-hop reasoning model. Finally, we sequentially perform reasoning and retrieval on the decomposed sub-questions, and then input both the questions and the retrieved paragraphs into the LLMs to generate the final answer.

Short-hop reasoning model. To ensure a fair comparison, we adopt the same base short-hop reasoning model with previous method [17]. The short-hop data contains two evidences x_1 and x_2. As the input to the short-hop model, the first evidence x_1 is prefixed by the question and the instruction "What evidence do we need to answer the question given the current evidence?". The output is the next evidence x_2. Utilizing the input-output pairs described above, the short-hop model is fine-tuned via LoRA [42].

During inference, the short-hop model first identifies the seeding node v_1, whose paragraph $\mathcal{X}_1$ has the highest cosine similarity to question q under TF-IDF embeddings. Then, inputting question q and previously retrieved j paragraphs $\{\mathcal{X}_i\}_{i=1}^{j}$, it reasons about the next evidence x_{j+1} and retrieves corresponding paragraph $\mathcal{X}_{j+1}$ as

[3]Available at: https://huggingface.co/sentence-transformers/all-mpnet-base-v2

Table 1: Main experimental results. Bold: the best score. Underline: the second best score.

Method \ Dataset	*Short → Long* HotpotQA FanoutQA		*Short → Long* HotpotQA LonghopQA		*Short → Long* 2WikiMQA FanoutQA		*Short → Long* 2WikiMQA LonghopQA		*Short → Long* MuSiQue FanoutQA		*Short → Long* MuSiQue LonghopQA	
	Acc.	*F1*	*Acc.*	*F1*	*Acc.*	*F1*	*Acc.*	*F1*	*Acc.*	*F1*	*Acc.*	*F1*
LLAMA-3.1-8B-INSTRUCT												
PURE-LLM	9.35	18.38	1.61	4.18	9.35	18.38	1.61	4.18	9.35	18.38	1.61	4.18
DPR	9.35	17.48	14.84	13.12	9.03	16.24	18.06	14.68	9.03	16.94	17.10	15.61
MDR	11.29	22.36	14.19	16.85	9.68	17.96	17.10	15.24	10.00	19.48	20.65	18.48
CURIOUSLLM	10.32	21.79	20.65	20.18	10.00	19.28	15.16	17.43	9.67	20.38	25.48	19.95
IRCOT	10.65	23.20	21.29	19.48	9.68	24.60	16.13	16.11	10.00	23.02	26.45	22.96
KGP	10.97	19.07	30.65	24.03	12.26	21.70	27.74	23.12	**15.16**	**25.89**	28.39	23.10
RSLG(OURS)	**16.77**	**24.38**	**35.48**	**27.14**	**16.13**	**24.78**	**30.97**	**25.31**	**15.16**	24.32	**29.03**	**23.59**
QWEN2.5-7B-INSTRUCT												
PURE-LLM	4.84	16.63	2.90	7.85	4.84	16.63	2.90	7.85	4.84	16.63	2.90	7.85
DPR	4.52	13.88	22.58	20.55	5.16	14.88	24.52	21.56	5.16	14.85	24.52	20.45
MDR	7.74	17.64	18.06	18.97	5.48	14.81	23.23	25.08	4.84	15.98	18.71	20.60
CURIOUSLLM	7.10	16.31	29.03	26.52	5.81	15.31	22.58	24.17	6.45	16.25	27.42	22.13
IRCOT	5.81	15.09	32.26	29.67	5.81	16.72	24.84	25.19	5.48	17.10	29.03	23.62
KGP	8.39	17.91	32.26	29.34	6.45	**18.39**	32.26	29.45	7.10	**19.78**	33.55	28.61
RSLG(OURS)	**10.00**	**18.89**	**38.39**	**35.93**	**10.97**	16.98	**34.19**	**33.12**	**8.06**	19.64	**37.10**	**36.35**
MISTRAL-7B-INSTRUCT-V0.3												
PURE-LLM	7.42	19.30	3.55	5.28	7.42	19.30	3.55	5.28	7.42	19.30	3.55	5.28
DPR	6.45	19.56	15.16	7.72	5.48	18.50	16.13	8.11	6.13	18.54	17.74	9.57
MDR	5.48	21.96	10.00	7.34	5.16	17.79	9.68	7.45	4.84	19.41	8.71	7.97
CURIOUSLLM	5.81	19.68	14.52	9.10	6.12	20.32	11.29	7.36	7.74	21.76	15.81	10.23
IRCOT	6.45	26.07	16.77	11.90	6.77	21.31	8.06	7.41	8.39	23.96	17.10	10.39
KGP	7.10	22.07	13.55	9.41	7.10	23.43	10.65	7.97	5.48	20.92	16.45	9.41
RSLG(OURS)	**10.32**	**26.60**	**24.52**	**12.38**	**10.32**	**26.54**	**20.00**	**10.46**	**9.35**	**25.96**	**18.71**	**11.03**

follows:

$$\mathcal{X}_{j+1} = \arg\max_{v \in \mathcal{N}_j} \phi\big(g(\mathcal{X}_v),\ g(\mathcal{F}(q,\ ||_{i=1}^{j}\mathcal{X}_i))\big), \tag{7}$$

where $||_{i=1}^{j}\mathcal{X}_i$ denotes the concatenation of previously retrieved paragraphs in visited nodes, $\mathcal{F}(\cdot)$ is the short-hop reasoning model for generating the next evidence x_i, $\mathcal{N}_j$ denotes the adjacent nodes of node v_j in the Graph $\mathcal{G}$, $\mathcal{X}_v$ is the paragraph of node v, $g(\cdot)$ is the TF-IDF encoder, and $\phi(\cdot)$ is the cosine similarity measuring the text similarity. Finally, we obtain a reasoning path $\{\mathcal{X}_1, \cdots, \mathcal{X}_n\}$. Following previous works [16, 17], we sample k reasoning paths that exhibit the top-k similarity scores with the evidence sequences.

4 Experiments

4.1 Datasets, Baselines, and Experimental Setup

Datasets. Following the *short-to-long generalization scenario*, models are trained on 2-hop data from HotpotQA [19], MuSiQue [21], 2WikiMQA [20] and subsequently tested on the long-hop datasets FanoutQA (mainly 5 to 13-hop) [13] and LonghopQA (mainly 3 to 4-hop). LonghopQA is derived from MuSiQue, retaining data that is longer than 2-hop. For each training set, there is a validation set of 500 samples drawn from the same distribution.

Baselines. Since MDQA is an emerging task, only KGP [17] and CURIOUSLLM [36] are specifically designed for it currently. For more comprehensive comparison, we transfer some strong MHQA baselines, including DPR [14], MDR [15], and IRCOT [16], to MDQA. Experiments are conducted on three representative LLMs: LLAMA-3.1-8B-INSTRUCT, QWEN2.5-7B-INSTRUCT, and MISTRAL-7B-INSTRUCT-V0.3.

Experimental Setup. We set all common hyperparameters to be consistent with baselines [16, 17, 36] without tuning. Specifically, the sample number of distractor documents d for each question is set to 10, the max paragraph length l_{max} is set to 250, the learning rate is set to $3e-4$, the fine-tuning epoch is set to 5, the batch size is set to 1024, the number of sample paths k is set to 30. The primary metric is Accuracy. Following previous methods [17, 36], a powerful LLM, GPT-4o, is employed to assess whether the generated answer and labeled answer express identical meanings. This approach has been proven to achieve similar or even higher levels than human evaluators [43, 44]. We also report $F1$ scores as an auxiliary metric. It measures the overlap between tokens in the generated and labeled answers. Note that $F1$ is a less reliable metric for this task, as predicted and labeled answers may share the same meaning despite having low token-level overlap.

4.2 Main Experimental Results

Table 1 shows the main experimental results. Key findings include:

High performance. Regardless of the specific LLMs and datasets employed, our method consistently achieves the state-of-the-art performance on the primary metric, *accuracy*. Although the $F1$ is a less reliable metric than *accuracy* in this task (as detailed in experimental setup), our method still performs best in most cases. These results underscore the effectiveness of our RSLG framework.

Table 2: Ablation results. "w/o": removing the corresponding strategy. Bold: the best score.

Method \ Dataset	*Short → Long* HotpotQA FanoutQA		*Short → Long* HotpotQA LonghopQA		*Short → Long* 2WikiMQA FanoutQA		*Short → Long* 2WikiMQA LonghopQA		*Short → Long* MuSiQue FanoutQA		*Short → Long* MuSiQue LonghopQA	
	Acc.	*F1*	*Acc.*	*F1*	*Acc.*	*F1*	*Acc.*	*F1*	*Acc.*	*F1*	*Acc.*	*F1*
RSLG	**16.77**	**24.38**	**35.48**	**27.14**	**16.13**	**24.78**	**30.97**	**25.31**	**15.16**	**24.32**	**29.03**	**23.59**
w/o question selector	15.16	24.06	33.55	26.95	14.52	24.69	28.39	23.10	14.19	23.43	27.42	22.47
w/o sub-question filter	15.48	22.75	32.26	26.16	15.16	22.86	29.68	23.91	13.87	23.50	**29.03**	22.22
w/o demonstration sampler	14.19	22.70	33.87	26.46	13.23	23.99	27.10	22.15	13.23	20.89	27.42	21.39
w/o parallel pattern	11.61	20.51	28.39	23.10	10.65	22.54	26.45	21.86	12.26	19.77	25.48	20.85
w/o sequential pattern	12.90	22.03	29.03	23.62	10.32	21.64	26.77	21.93	12.90	22.38	26.45	21.55
w/o decomposability criterion	13.87	23.50	29.68	24.34	12.90	21.66	27.42	22.93	13.23	20.65	26.77	21.85
w/o redundancy criterion	14.19	21.59	33.23	26.02	13.87	23.09	30.32	24.49	14.19	22.70	28.39	23.10
w/o relevance criterion	15.48	22.97	32.26	25.95	14.52	23.36	29.68	23.73	13.87	19.66	28.71	22.93
w/o all strategies	10.32	21.64	28.06	24.24	10.00	19.97	25.81	20.58	11.61	20.70	24.84	20.16

Strong Domain Generalization. For training and test data from different domains, our method demonstrates strong out-of-domain generalization capability. For example, the training set 2WikiMQA primarily covers personal information, films, and albums, while the test data FanoutQA focuses on geography, history, and sports. Compared with previous SOTA methods, RSLG achieves an average absolute *accuracy* improvement of 3.76% across three LLMs, showing a strong out-of-domain generalization capability.

Necessity of Short-to-Long Generalization. Answering long-hop questions remains highly challenging for LLMs without multi-hop reasoning on external knowledge (Pure-LLM). The internal knowledge of LLMs is limited and may be outdated, whereas long-hop questions require reasoning across multiple evidences. Since long-hop data is difficult and expensive to collect at scale for model training, improving the short-to-long generalization capability is critical for effectively resolving long-hop questions.

4.3 Ablation Study

We conduct ablation study on Llama-3.1-8B-Instruct to assess the contribution of each component. The results are shown in Table 2.

First, removing any component leads to a significant performance drop, demonstrating the effectiveness of each strategy. In particular, removing either the parallel or sequential pattern causes a larger performance drop than removing others, proving that these patterns provide crucial guidance for the decomposition process.

Second, within the local-to-global demonstration construction strategy, removing the demonstration sampler generally produces the biggest performance loss, highlighting the importance of diversity in recursive demonstrations, as it enhances their applicability to a broader range of target questions. Among the recursive termination criteria, the decomposability criterion matters most to the performance, as it is crucial to avoid under-decomposition and over-decomposition simultaneously.

Third, removing all components generally causes a larger performance drop than removing any single one. This suggests that these components complement each other across different aspects and collaboratively achieve excellent performance.

4.4 Reasoning Effectiveness Analysis

In this section, we verify whether RSLG is effective in determining *where to go next* and *when to stop reasoning* during reasoning stage.

The effectiveness of reasoning can be reflected by the *recall* and *precision* metrics computed between the retrieved supporting paragraphs and golden paragraphs. Generally, if the method can effectively determine *where to go next* and *when to stop reasoning* during reasoning stage, it is expected to achieve higher *recall* and *precision* in retrieving the relevant paragraphs. As FanoutQA does not provide annotated golden paragraphs, we conduct experiments on LonghopQA, with models trained on HotpotQA. The sample paths refer to the retrieved supporting paragraph paths that achieve the highest top-k similarity scores with the reasoning paths composed of generated evidences. The results are shown in Fig. 5.

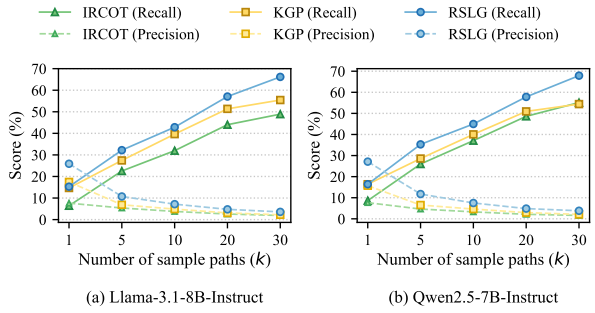


Figure 5: The *recall* and *precision* of retrieved paragraphs.

Overall, our method consistently outperforms IRCOT and KGP in both *precision* and *recall*. Notably, with a small number of sample paths, RSLG achieves substantially higher *precision* than baselines, indicating that it is more effective at precisely identifying the next steps and stopping at the appropriate point to avoid under-decomposition and over-decomposition. For instance, when sampling just one path, RSLG attains absolute gains of 8.44% and 11.42% in *precision* over the previous best results on the two LLMs, respectively. As the number of sample paths increases, improvements in *recall* become more pronounced, reflecting our method's enhanced reasoning ability to find the relevant evidences as fully as possible without stopping at an inappropriate point. For example, when sampling 30 paths, it achieves absolute gains of 10.72% and 12.77% in *recall* over the prior best results on the two LLMs, respectively. In summary, these findings demonstrate that our method excels at effectively determining *where to go next* and *when to stop reasoning*.

4.5 Decomposition Effectiveness Analysis

In this section, we evaluate our recursion-based decomposition strategy by comparing it with other high-performance decomposition strategies, including PS+ [45], SELF-ASK [46], and LEAST-TO-MOST [18]. Since these strategies cannot work independently, we integrate them into the RSLG framework. By replacing only decomposition component while keeping all other modules unchanged, we ensure a fair comparison focused solely on the decomposition.

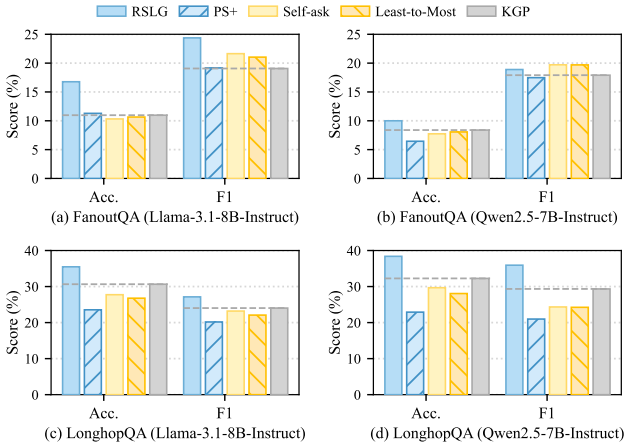


Figure 6: The performance of decomposition strategies.

As shown in Fig. 6, our decomposition strategy significantly outperforms others. The primary reason is that our approach learns to decompose long-hop questions through two recursive patterns using only short-path data. Conversely, other methods depend on demonstrations similar to the target complexity or rely entirely on the LLMs themselves, making them less effective for short-to-long generalization. Furthermore, their performance often falls below that of the short-hop method KGP. This highlights that without long-path data, these decomposition strategies struggle to handle the complexities of long-path reasoning.

4.6 The Effectiveness on Different Long Hops

In this section, we investigate the advantages of our method over the previous SOTA method (KGP) across questions with varying hops. The LLAMA-3.1-8B-INSTRUCT is adopted as the underlying LLM for this and the following sections. We group data from the LonghopQA and FanoutQA datasets based on reasoning hops and separately evaluate performance on different hops. As shown in Fig. 7, as the number of required reasoning hops increases, our method achieves increasingly significant performance gains compared to KGP. This trend indicates that our method possesses stronger adaptability when handling complex long-hop questions.

4.7 The Effectiveness of Short-hop Reasoning

In this section, we evaluate on short-hop data as previous multi-hop methods. We compare our method against state-of-the-art baselines on three 2-hop datasets. Other settings are consistent with the main experiments. The experimental results, presented in

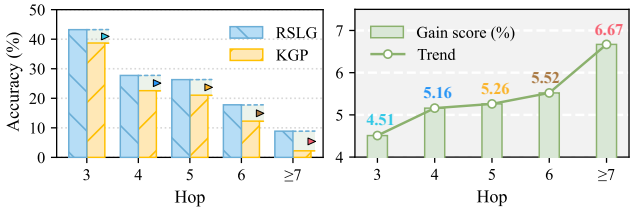


Figure 7: The performance gains on different long hops.

Table 3, demonstrate that our RSLG method also performs well on short-hop reasoning. This observation proves that our method is widely-adaptive on both short-hop and long-hop reasoning data.

Table 3: The comparison of short-hop reasoning.

Dataset	HotpotQA		2WikiMQA		MuSiQue	
	Acc.	*F1*	*Acc.*	*F1*	*Acc.*	*F1*
PURE-LLM	38.60	27.84	36.60	21.17	30.60	13.49
IRCOT	68.40	43.97	51.80	31.64	49.20	26.38
KGP	70.60	45.78	54.80	34.06	50.60	27.77
RSLG(OURS)	**74.80**	**47.62**	**57.40**	**35.89**	**54.00**	**28.93**

4.8 Robustness Analysis

This section evaluates the RSLG's robustness across different seed demonstrations, redundancy and relevance hyperparameters.

As shown in Fig. 8, on three groups of random seed demonstrations, RSLG exhibits great stability, as the standard deviation of performance stays below 1.00% across different datasets. This high robustness is primarily attributed to the rigorous selector–filter–sampler pipeline in the demonstration construction strategy, which effectively avoids the instability caused by seed demonstrations and ensures the high quality of final demonstrations.

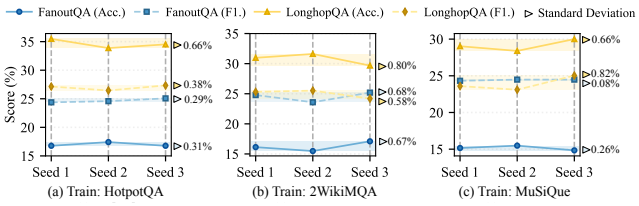


Figure 8: The robustness on different seeds.

As illustrated in Fig. 9, RSLG also demonstrates robust stability across a wide range of values around the hyperparameter determined on the short-hop HotpotQA validation set. The standard deviation of performance remains below 0.50% across datasets from different domains. This indicates that RSLG is not sensitive to the two hyperparameters and generalizes well to different domains.

Overall, the above results demonstrate the high robustness of our method, confirming its suitability for real-world deployment.

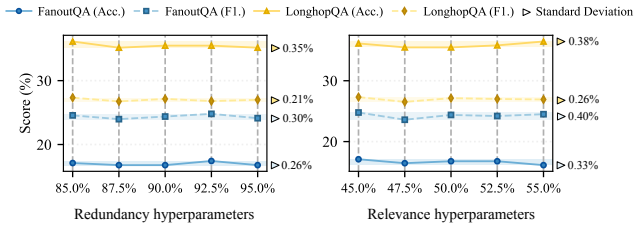


Figure 9: The robustness on different hyperparameters.

4.9 Efficiency Analysis

We evaluate the efficiency on NVIDIA A800 by comparing with KGP and IRCOT, two representative baselines known for their high efficiency and competitive performance. The comparison results on FanoutQA dataset are shown in Fig. 10.

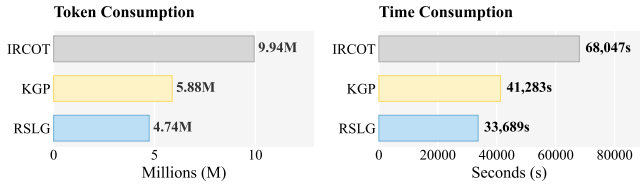


Figure 10: The comparison of token and time consumption

Our method is more efficient than KGP and IRCOT in both token and time consumption. KGP and IRCOT repeatedly include all previously retrieved information at each reasoning step, leading to a rapid increase in token usage and inference time as reasoning depth grows. In contrast, our method decomposes long-hop questions into independent short-hop sub-questions. Consequently, it achieves lower token growth and latency, demonstrating superior practicality for complex and large-scale applications.

4.10 Case Study

To intuitively demonstrate the decomposition effectiveness, we analyze a representative case. As shown in Fig. 11 (a), RSLG recursively decomposes a long-hop (i.e., 8) question based on both parallel and sequential patterns, and the decomposition terminates when any of the termination criteria is met. Finally, the sub-questions located at the leaf nodes (blue dashed boxes) require only short-hop reasoning, which can be easily handled by a short-hop reasoning model.

In addition, we select a SOTA decomposition method and depict its decomposition processes in Fig. 11 (b). Our observations reveal that even when utilizing the advanced GPT-4o, the generated sub-questions remain suboptimal. First, the decomposition granularity is coarse-grained, with several essential short-hop reasoning sub-questions omitted. The reason is that questions with a sequential pattern cannot be decomposed directly. For instance, "What is the box office value of The Phantom Menace (1999)?" cannot be generated directly when "The Phantom Menace (1999)" is not explicitly available. Second, sub-questions 5 and 6 do not contribute to resolving the root question. Furthermore, sub-question 7 diverges from the root question and is likely to mislead the answer.

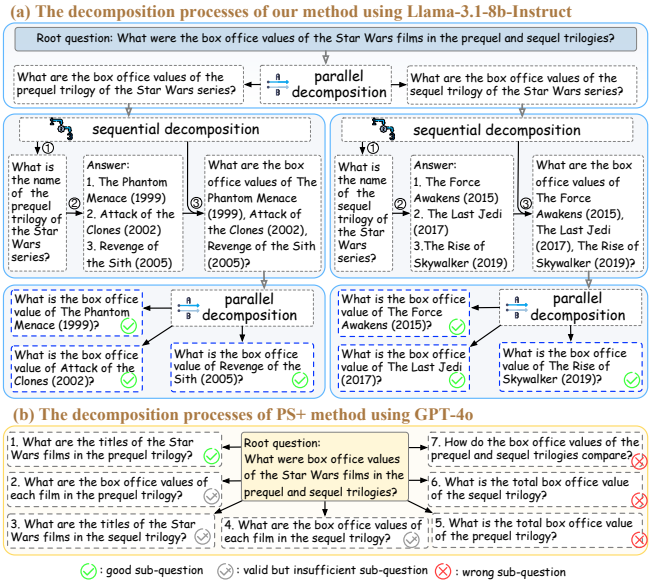


Figure 11: Case study.

5 Conclusion

In this paper, we explore the *short-to-long generalization scenario* for the first time to enhance the model's capability to handle long-hop questions using easily collected short-hop reasoning data. To address the challenges in determining *where to go next* and *when to stop*, we propose the RSLG framework inspired by recursive algorithms. It recursively decomposes long-hop questions into multiple short-hop questions using both parallel and sequential recursion patterns. Furthermore, three termination criteria are proposed from the perspectives of decomposability, redundancy, and relevance to ensure an appropriate stopping point. Extensive experiments demonstrate the superiority of RSLG with excellent performance, robustness, and efficiency.

Acknowledgments

This work was supported by the grants from the National Natural Science Foundation of China (NSFC) project (Grant No. 62276193, 62576256) and the Fundamental Research Funds for the Central Universities,China (Grant No. 2042022dx0001).

References

[1] Fei Yu, Hongbo Zhang, Prayag Tiwari, and Benyou Wang. Natural language reasoning, a survey. *ACM Computing Surveys*, 56(12):1–39, 2024.
[2] Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. In *Findings of the association for computational linguistics: ACL 2023*, pages 1049–1065, 2023.
[3] Tassallah Abdullahi, Ioanna Gemou, Nihal V Nayak, Ghulam Murtaza, Stephen H Bach, Carsten Eickhoff, and Ritambhara Singh. K-paths: Reasoning over graph paths for drug repurposing and drug interaction prediction. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 5–16, 2025.
[4] Travis Zack, Gurpreet Dhaliwal, Rabih Geha, Mary Margaretten, Sara Murray, et al. A clinical reasoning-encoded case library developed through natural language processing. *Journal of General Internal Medicine*, 38(1):5–11, 2023.
[5] Mariano Barone, Antonio Romano, Giuseppe Riccio, Marco Postiglione, and Vincenzo Moscato. Combining evidence and reasoning for biomedical fact-checking. In *Proceedings of the 48th International ACM SIGIR Conference on*

Research and Development in Information Retrieval, pages 1087–1097, 2025.

[6] Behrooz Mansouri and Reihaneh Maarefdoust. Using large language models for math information retrieval. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2693–2697, 2024.

[7] Changyu Chen, Xiting Wang, Ting-En Lin, Ang Lv, Yuchuan Wu, Xin Gao, Ji-Rong Wen, Rui Yan, and Yongbin Li. Masked thought: Simply masking partial reasoning steps can improve mathematical reasoning learning of language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5872–5900, 2024.

[8] Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research*, 2023, 2023.

[9] Kepu Zhang, Weijie Yu, Zhongxiang Sun, and Jun Xu. Syler: A framework for explicit syllogistic legal reasoning in large language models. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 4117–4127, 2025.

[10] Albert Sadowski and Jaroslaw A Chudziak. On verifiable legal reasoning: A multi-agent framework with formalized knowledge representations. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 2535–2545, 2025.

[11] Guijin Son, Hanearl Jung, Moonjeong Hahm, Keonju Na, and Sol Jin. Beyond classification: Financial reasoning in state-of-the-art language models. In *Proceedings of the Fifth Workshop on Financial Technology and Natural Language Processing and the Second Multimodal AI For Financial Forecasting*, pages 34–44, 2023.

[12] Yilun Zhao, Yitao Long, Hongjun Liu, Ryo Kamoi, Linyong Nan, Lyuhao Chen, Yixin Liu, Xiangru Tang, Rui Zhang, and Arman Cohan. Docmath-eval: Evaluating math reasoning capabilities of llms in understanding long and specialized documents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16103–16120, 2024.

[13] Andrew Zhu, Alyssa Hwang, Liam Dugan, and Chris Callison-Burch. Fanoutqa: A multi-hop, multi-document question answering benchmark for large language models. In *ACL*, pages 18–37, 2024.

[14] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pages 6769–6781, 2020.

[15] Wenhan Xiong, Xiang Lorraine Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Scott Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oguz. Answering complex open-domain questions with multi-hop dense retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021.

[16] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 10014–10037, 2023.

[17] Yu Wang, Nedim Lipka, Ryan A Rossi, Alexa Siu, Ruiyi Zhang, and Tyler Derr. Knowledge graph prompting for multi-document question answering. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 19206–19214, 2024.

[18] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V. Le, and Ed H. Chi. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*, 2023.

[19] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2369–2380, 2018.

[20] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, 2020.

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

[22] Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel M. Ni, Heung-Yeung Shum, and Jian Guo. 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*, 2024.

[23] Zhangyue Yin, Yuxin Wang, Xiannian Hu, Yiguang Wu, Hang Yan, Xinyu Zhang, Zhao Cao, Xuanjing Huang, and Xipeng Qiu. Rethinking label smoothing on multi-hop question answering. In *China National Conference on Chinese Computational Linguistics*, pages 72–87. Springer, 2023.

[24] Ronald B Dekker, Fabian Otto, and Christopher Summerfield. Curriculum learning for human compositional generalization. *Proceedings of the National Academy of Sciences*, 119(41):e2205582119, 2022.

[25] Yingxu Wang and Vincent Chiew. On the cognitive process of human problem solving. *Cognitive systems research*, 11(1):81–92, 2010.

[26] Andreas Fischer, Samuel Greiff, and Joachim Funke. The process of solving complex problems. *The Journal of Problem Solving*, 4(1):3, 2012.

[27] Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*, 2023.

[28] Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. Complex knowledge base question answering: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 35(11):11196–11215, 2023.

[29] Ke Liang, Lingyuan Meng, Meng Liu, Yue Liu, Wenxuan Tu, Siwei Wang, Sihang Zhou, Xinwang Liu, et al. A survey of knowledge graph reasoning on graph types: Static, dynamic, and multi-modal. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):9456–9478, 2024.

[30] Trond Linjordet and Krisztian Balog. Would you ask it that way? measuring and improving question naturalness for knowledge graph question answering. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3090–3098, 2022.

[31] Wentao Ding, Jinmao Li, Liangchuan Luo, and Yuzhong Qu. Enhancing complex question answering over knowledge graphs through evidence pattern retrieval. In *Proceedings of the ACM Web Conference 2024*, pages 2106–2115, 2024.

[32] Yike Wu, Yi Huang, Nan Hu, Yuncheng Hua, Guilin Qi, Jiaoyan Chen, and Jeff Z Pan. Cotkr: Chain-of-thought enhanced knowledge rewriting for complex knowledge graph question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3501–3520, 2024.

[33] Armin Toroghi, Willis Guo, Mohammad Mahdi Abdollah Pour, and Scott Sanner. Right for right reasons: Large language models for verifiable commonsense knowledge graph question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6601–6633, 2024.

[34] Tiesunlong Shen, Rui Mao, Jin Wang, Xuejie Zhang, and Erik Cambria. Flow-guided direct preference optimization for knowledge graph reasoning with trees. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1165–1175, 2025.

[35] Zhouyu Jiang, Mengshu Sun, Lei Liang, and Zhiqiang Zhang. Retrieve, summarize, plan: Advancing multi-hop question answering with an iterative approach. In *Companion Proceedings of the ACM on Web Conference*, pages 1677–1686, 2025.

[36] Zukang Yang, Zixuan Zhu, and Jennifer Zhu. Curiousllm: Elevating multi-document question answering with llm-enhanced knowledge graph reasoning. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 3: Industry Track)*, pages 274–286, 2025.

[37] Paolo Ferragina and Ugo Scaiella. Tagme: on-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1625–1628, 2010.

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

[39] Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. Complexity-based prompting for multi-step reasoning. In *The Eleventh International Conference on Learning Representations, ICLR 2023*, 2023.

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

[41] James MacQueen. Some methods for classification and analysis of multivariate observations. In *the Berkeley Symposium on Mathematical Statistics and Probability*, volume 5, pages 281–298, 1967.

[42] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*, 2022.

[43] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *NeurIPS*, 36:46595–46623, 2023.

[44] Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. Chatgpt outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30):e2305016120, 2023.

[45] Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 2609–2634, 2023.

[46] Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711, 2023.