Beyond the Answer: Advancing Multi-Hop QA with Fine-Grained Graph Reasoning and Evaluation

Qichuan Liu[1], **Chentao Zhang**[1], **Chenfeng Zheng**[1], **Guosheng Hu**[2], **Xiaodong Li**[1], **Zhihong Zhang**[1†]
[1]School of Informatics, Xiamen University, China
[2]School of Engineering Mathematics and Technology, University of Bristol, UK
{qcliu,zhangchentao,zhengchenfeng}@stu.xmu.edu.cn
{xdli,zhihong}@xmu.edu.cn

Abstract

Recent advancements in large language models (LLMs) have significantly improved the performance of multi-hop question answering (MHQA) systems. Despite the success of MHQA systems, the evaluation of MHQA is not deeply investigated. Existing evaluations mainly focus on comparing the final answers of the reasoning method and given ground-truths. We argue that the reasoning process should also be evaluated because wrong reasoning process can also lead to the correct final answers. Motivated by this, we propose a "**P**lanner-**E**xecutor-**R**easoner" (PER) architecture, which forms the core of the Plan-anchored Data Preprocessing (PER-DP) and the Plan-guided Multi-Hop QA (PER-QA). The former provides the ground-truth of intermediate reasoning steps and final answers, and the latter offers them of a reasoning method. Moreover, we design a fine-grained evaluation metric called **P**lan-aligned **S**tepwise **E**valuation (PSE), which evaluates the intermediate reasoning steps from two aspects: planning and solving. Extensive experiments on ten types of questions demonstrate competitive reasoning performance, improved explainability of the MHQA system, and uncover issues such as "fortuitous reasoning continuance" and "latent reasoning suspension" in RAG-based MHQA systems. Besides, we also demonstrate the potential of our approach in data contamination scenarios. Our data and code have been released at https://github.com/GenIRAG/PER-PSE.

1 Introduction

Recent advancements in large language models (LLMs) have enhanced the performance of multi-hop question answering (MHQA) systems (Gao et al., 2023a; Ma et al., 2023; Press et al., 2023; Trivedi et al., 2023), enabling them to address complex reasoning tasks. Unlike single-hop QA systems, which rely on direct retrieval from a single document, MHQA requires models to perform multiple reasoning steps across different documents to find the correct answer. For example, consider the question in Figure 1: *"How many campuses are there in the institution that Desert Studies Center is a part of?"*. The model should first identify that "Desert Studies Center" belongs to "California State University" and then determine that "California State University" has "12 Campuses", ultimately reasoning that the final answer is *"12"*.

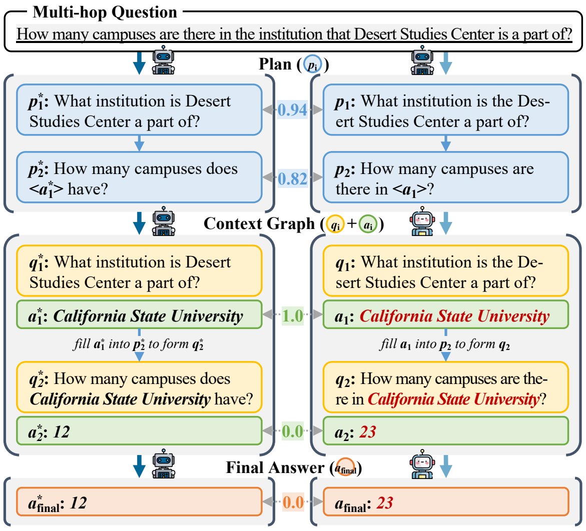


Figure 1: The core idea of our approach is to explicitly represent *the planning and solving of intermediate steps* via graph structures. "*": ground-truth. In this 2-hop bridge question, the first sub-question is answered correctly, but the second one is incorrect, resulting in an inaccurate final answer. Evaluating only the final answer fails to capture the true performance of MHQA.

Existing works seek to improve the performance of MHQA system by stating the intermediate steps to conduct multi-hop reasoning (Press et al., 2023; Trivedi et al., 2023; Zhou et al., 2024). However, accurately evaluating the correctness and completeness of reasoning remains difficult due to several key issues: lack of annotation for intermediate reasoning steps, different reasoning methods make

†Corresponding author

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 23433–23456
July 27 - August 1, 2025

intermediate steps hard to trace uniformly, and evaluation metrics such as Exact Match (EM) and F1, which overlook intermediate reasoning steps.

Thus, fine-grained evaluation requires a gold standard that provides intermediate reasoning steps, and reasoning methods should enable the explicit generation of these steps for alignment and analysis. This necessitates a coordinated arrangement of "Data-Reasoning-Evaluation" to ensure effective and accurate assessment. Therefore, we propose a "**P**lanner-**E**xecutor-**R**easoner" (PER) architecture, which serves as the core for the Plan-anchored Data Preprocessing pipeline (PER-DP) and the Plan-guided Multi-Hop QA pipeline (PER-QA). The *planner* generates a graph-structured plan based on the given question, the *executor* executes the plan in topological order to obtain sub-questions and step-answers, and the *reasoner* turns them into context for reasoning. Using PER-DP, we regularize and expand the data from HotpotQA (Yang et al., 2018), 2WikiMultihopQA (Ho et al., 2020) and MuSiQue (Trivedi et al., 2022), generating 8,394 samples with gold intermediate steps. Then, we sample on these data and use PER-QA for reasoning with two settings: PER-QA$_{\text{vanilla}}$ and PER-QA$_{\text{rag}}$. The former relies on the parametric knowledge of the LLM, while the latter relies on retrieval-augmented generation (RAG). Beyond PER-DP and PER-QA, we design a fine-grained evaluation metric, called **P**lan-aligned **S**tepwise **E**valuation (PSE). As shown in Figure 1, unlike conventional metrics that focus on final answers, PSE focuses more on the intermediate reasoning steps.

We conduct extensive experiments on ten types of questions, achieve competitive QA performance, improve the explainability of the MHQA system, identify the *fortuitous reasoning continuance* and *latent reasoning suspension* in the RAG-based MHQA system, and show potential in data contamination scenarios, offering a solid base for future research in LLM-based knowledge-intensive tasks. In summary, our contributions include:

- Existing MHQA evaluations mainly focus on evaluating the final answers, potentially causing bias. In this work, we propose a new evaluation framework which can evaluate both reasoning process and final answers.

- We introduce PER, a task-oriented agent architecture, and develop PER-DP for data preprocessing and PER-QA for MHQA, supporting the evaluation of both intermediate reasoning steps and final answers.

- To conduct our evaluations, we present PSE, a new graph-based fine-grained evaluation metric, to better adapt to relevant scenarios of the underlying explainability.

- With our new reasoning and evaluation framework, we clearly find that the RAG-based MHQA system experiences "fortuitous reasoning continuance" and "latent reasoning suspension". Moreover, we show the potential of our approach for data contamination.

2 Related Works

Multi-hop QA Reasoning. MHQA presents notable challenges for LLMs due to its demand for systematic integration of distributed knowledge and multi-step reasoning (Yu et al., 2024). The core principle for tackling such tasks is to break them down and solve them incrementally (Patel et al., 2022). Several studies have explored using prompt engineering to solve questions step-by-step (Dua et al., 2022; Zhou et al., 2023; Khot et al., 2023; Wei et al., 2022), though these approaches often struggle to effectively address the sub-questions, primarily due to the lack of external knowledge.

Retrieval-augmented generation (RAG) improves performance by integrating external knowledge through various strategies. (Gao et al., 2023b; Zhang et al., 2025). One effective strategy for MHQA is iterative retrieval and reasoning (Press et al., 2023; Trivedi et al., 2023; Shi et al., 2024; Zhuang et al., 2024). For example, Self-Ask (Press et al., 2023) progressively decomposes the question through self-generated queries, while IRCoT (Trivedi et al., 2023) employs iterative Chain-of-Thought (CoT) to support stepwise retrieval and reasoning. However, these approaches may lack a comprehensive view of the overall reasoning process. In contrast, recent studies have shifted towards explicit planning (Verma et al., 2025; Xin et al., 2024), where systems develop a reasoning plan before QA, organizing these plans into structured formats like trees or graphs to make the reasoning process more transparent and explainable.

Multi-hop QA Evaluation. Traditional evaluation metrics for MHQA systems, such as Exact Match (EM) and F1, primarily assess the correctness of the final answer (Mavi et al., 2024). Similarly, LLM-based evaluation methods offer a

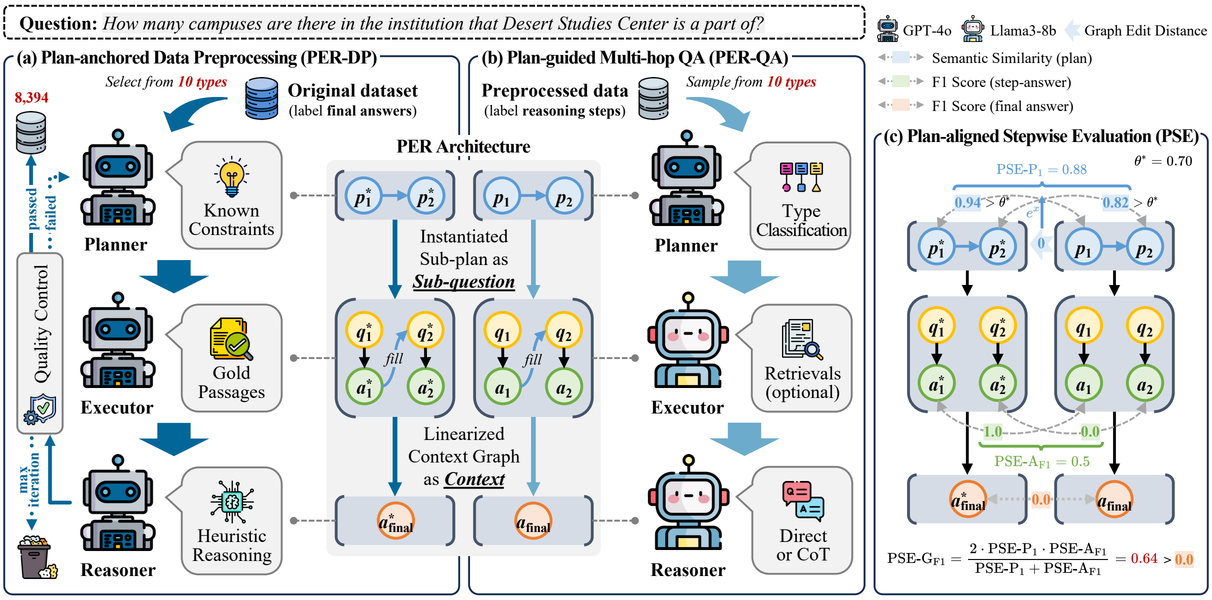


Figure 2: The overview of the "Data-Reasoning-Evaluation". We propose the "**P**lanner-**E**xecutor-**R**easoner" (PER) architecture, which represents intermediate steps based on a graph structure. Based on PER, we design Plan-anchored Data Preprocessing pipeline (PER-DP) and Plan-guided Multi-hop QA pipeline (PER-QA). The former is used to obtain the gold annotations of the intermediate steps, while the latter conducts reasoning through explicit intermediate steps. In addition, to better evaluate the steps, we design **P**lan-aligned **S**tepwise **E**valuation (PSE), which reveals the true performance of MHQA by evaluating the planning and solving of the intermediate steps.

more flexible approach (Wang et al., 2023, 2024). However, they often tend to focus mainly on evaluating the final answer, and pay relatively less attention to the underlying reasoning process involved in MHQA (Tang et al., 2021). While earlier benchmarks like HotpotQA (Yang et al., 2018), 2WikiMultihopQA (Ho et al., 2020), and MuSiQue (Trivedi et al., 2022) primarily evaluate answer accuracy and alignment with ground truth, recent benchmarks such as MultiHop-RAG (Tang and Yang, 2024), FANOutQA (Zhu et al., 2024), and MintQA (He et al., 2024) offer a more nuanced and comprehensive evaluation of multi-hop reasoning capabilities. These newer benchmarks emphasize multi-document comprehension and the complexity of reasoning steps. While works like TWOHOPFACT (Yang et al., 2024a), SOCRATES (Yang et al., 2024b), and CofCA (Wu et al., 2025) have made valuable progress in assessing multi-hop reasoning, they emphasize evaluating LLM capabilities rather than focusing on MHQA, and do not assess the dynamic planning and solution process required in open-ended scenarios. Therefore, a more comprehensive evaluation is needed, one that considers both the structure of reasoning plans and their suitability for different question types, to better assess the full scope of MHQA system.

3 Fine-grained Reasoning and Evaluation

We propose a novel reasoning and evaluation framework via graph structure that focuses on the intermediate reasoning steps of multi-hop QA systems. As shown in Figure 2, it consists of three components: Based on the PER architecture, we design (i) the **Plan-anchored Data Preprocessing** pipeline (PER-DP) to supplement intermediate steps for the MHQA dataset and (ii) the **Plan-guided Multi-Hop QA** pipeline (PER-QA) to achieve multi-hop reasoning with explicit intermediate reasoning steps. In addition, we design (iii) the **Plan-aligned Stepwise Evaluation** (PSE) to evaluate the planning and solving of intermediate reasoning steps.

3.1 PER Architecture

The "**P**lanner-**E**xecutor-**R**easoner" (PER) architecture consists of three task-oriented agents. Given a multi-hop question, the *planner* first formulate a plan. A plan can be represented as a graph $\mathcal{G}_{\mathcal{P}} = \{\mathcal{P}, \mathcal{E}_{\mathcal{P}}\}$. Each node $p_i \in \mathcal{P}$ represents a sub-plan, each edge $(p_i, p_j) \in \mathcal{E}_{\mathcal{P}}$ explicitly defines the execution order between sub-plans. Then, the *executor* answers each sub-question q_i after instantiating each sub-plan p_i as a sub-question q_i according to the execution order. After the entire plan $\mathcal{P}$ is executed, all intermediate steps will form a context graph $\mathcal{G}_{\mathcal{C}} = \{\mathcal{C}, \mathcal{E}_{\mathcal{C}}\}$. Each context $c_i = (q_i, a_i) \in \mathcal{C}$ contains a sub-question and its answer, each edge $(c_i, c_j) \in \mathcal{E}_{\mathcal{C}}$ explicitly defines the logical relationship between sub-questions. Finally, the *reasoner* will use the linearized context $q_1 \circ a_1 \circ q_2 \circ a_2 \circ \cdots \circ q_{|\mathcal{P}|} \circ a_{|\mathcal{P}|}$ to perform multi-hop reasoning and obtain the final answer a_{final}.

3.1.1 Plan-anchored Data Preprocessing

Considering that there is no complete annotation of intermediate steps in the current MHQA datasets, this poses challenges to the fine-grained evaluation of multi-hop reasoning. Therefore, we propose a Plan-anchored Data Preprocessing pipeline (PER-DP) based on PER for regularizing and expanding data from the open-source MHQA datasets. The overview of PER-DP is shown in Figure 2 (a).

Planner. We identify ten types of multi-hop questions in HotpotQA (Yang et al., 2018), 2WikiMultihopQA (Ho et al., 2020) and MuSiQue (Trivedi et al., 2022), such as *2-hop bridge* (Bri.^{2H}), *2-hop comparison* (Comp.^{2H}), *2-hop inference* (Infer.^{2H}), and *4-hop bridge-comparison* (B.C.^{4H}). Later, we design ten personalized prompts for ten types to guide the *planner* to generate plans for each multi-hop question. We use graph isomorphism matching for strict structural verification. More detailed information can be found in Appendix A.

Executor. To collect the ground-truth of intermediate reasoning steps, we provide the *executor* with golden passages paired with sub-questions in the original dataset. To explore all possibilities of intermediate reasoning steps, we allow the *executor* to answer one or more answers for each step.

Reasoner. To avoid hallucination in multi-hop reasoning by LLMs, we design heuristic multi-hop reasoning according to different question types to obtain the final answer. Heuristic multi-hop reasoning includes *symbolic reasoning*, *LLM reasoning*, and *mathematical reasoning*. Specifically, for all bridge type (Bri.^{2H}, Bri.^{3H}, and Bri.^{4H}) and inference type (Infer.^{2H}), we adopt symbolic reasoning and take the answer to the last sub-question as the final answer; for 2-hop comparison type (Comp.^{2H}) and bridge-comparison type (B.C.^{4H}), we provide intermediate reasoning steps to the LLM and require it to answer "yes/no" or "span answer" as the final answer; for 4-hop comparison type (Comp.^{4H}), we employ mathematical reasoning, where a program parses dates and calculates the "lifespan", yielding a "span answer" as the final answer[1].

Type	HotpotQA	2WikiMQA	MuSiQue
Bri.^{2H}	1,166	1,036	1,018
Bri.^{3H1}	-	-	478
Bri.^{3H2}	-	-	163
Bri.^{4H1}	-	-	230
Bri.^{4H2}	-	-	31
Bri.^{4H3}	-	-	52
Infer.^{2H}	-	1,013	-
Comp.^{2H}	1,052	1,011	-
Comp.^{4H}	-	125	-
B.C.^{4H}	-	1,019	-
Total **(8,394)**	2,218	4,204	1,972

Table 1: The statistics of the preprocessed samples. Some types have fewer than 1,000 due to their limited quantity in the original dataset. For more details about PER-DP and preprocessed samples, see Appendix C.

Quality Control and Final Dataset. To overcome possible data leakage (Chen and Durrett, 2019; Min et al., 2019) and overly strict ground-truth in MHQA datasets, we conduct quality control through *automatic evaluation*, *leakage detection*, and *multiple iteration*. Given that powerful LLMs have been proven to have near-human-level performance as judges (Wang et al., 2023), we use powerful LLMs to evaluate the final answers and supplement the dataset with the final answers evaluated as correct. In addition, we conduct leakage detection for bridge-type questions. We do not allow the final answer to appear before the last reasoning step. If a sample fails the automatic evaluation and leakage detection, it will re-enter the PER-DP. This process continues until the maximum number of iterations is reached. Finally, we obtained a total of 8,394 preprocessed samples, with contributions of 2,218 from HotpotQA, 4,204 from 2WikiMultihopQA, and 1,972 from MuSiQue. The statistics of the preprocessed samples are shown in Table 1.

3.1.2 Plan-guided Multi-Hop QA

Based on PER, we propose a Plan-guided Multi-Hop QA pipeline (PER-QA) for multi-hop reasoning that explicitly specifies intermediate reasoning steps. According to whether the *executor* uses external knowledge (i.e., whether to retrieve), we provide two settings: PER-QA$_{\text{vanilla}}$ and PER-QA$_{\text{rag}}$. The overview of PER-QA is shown in Figure 2 (b).

Unlike PER-DP which provides personalized

[1]The 4-hop comparison questions are structured as follows: "*Who lives longer,* ***A*** *or* ***B****?*". To answer this, one must first gather the birth and death dates of both **A** and **B**.

multi-hop reasoning for each question type, we provide a more universal setting for PER-QA. Specifically, in PER-QA, the *planner* first independently determines the question type and makes a plan. Then, the *executor* can choose to use internal knowledge or external knowledge to determine an answer for sub-question. Finally, the *reasoner* linearizes all intermediate reasoning steps of sub-questions and their answers into the context and completes multi-hop reasoning through In-Context Learning (ICL). The ICL strategy can be flexibly changed, we provide two strategies: direct reasoning and Chain-of-Thought (CoT) reasoning. We choose direct reasoning in the experiment to demonstrate the basic performance of PER-QA.

3.2 Plan-aligned Stepwise Evaluation

With the support of data and reasoning methods, we propose a fine-grained evaluation metric, called **P**lan-aligned **S**tepwise **E**valuation (PSE), to accurately reflect the true multi-hop reasoning capabilities of LLMs and provide support for perceiving the intermediate steps of the MHQA system. The overview of PSE is shown in Figure 2 (c).

Plan Evaluation. The plan are mostly masked by placeholders, shifting the focus to abstract semantics. Given a multi-hop question, we jointly consider structural matching and semantic similarity (the similarity θ of the sub-plan must exceed threshold θ^*) to find a mapping $\mathcal{M}_{\mathcal{P}} : \mathcal{P}'_{\text{pred}} \rightarrow \mathcal{P}'_{\text{gold}}$ (Algorithm see Appendix E), where $\mathcal{P}'_{\text{pred}}$ is a subset of the predicted plan $\mathcal{P}_{\text{pred}}$, $\mathcal{P}'_{\text{gold}}$ is a subset of the gold plan $\mathcal{P}_{\text{gold}}$. Each node $p_{\text{pred}} \in \mathcal{P}'_{\text{pred}}$ corresponds to exactly one node $p_{\text{gold}} \in \mathcal{P}'_{\text{gold}}$, and vice versa. For each $(p_{\text{gold}}, p_{\text{pred}}) \in \mathcal{M}_{\mathcal{P}}$, we use `all-MiniLM-L6-v2`[2] to obtain embeddings and define the semantic score of the predicted plan, i.e.,

$$s_{\textbf{sem}} = \frac{\sum_{i=1}^{|\mathcal{M}_{\mathcal{P}}|} \cos\left(\boldsymbol{p}_{\text{pred}}^{(i)}, \boldsymbol{p}_{\text{gold}}^{(i)}\right)}{|\mathcal{P}_{\text{gold}}|}. \tag{1}$$

However, $s_{\textbf{sem}}$ does not take into account structural differences, which may provide a perspective for explaining the reasons behind performance. Therefore, we use Graph Edit Distance (GED) as the structural score. The GED is defined as the minimal number of operations required to transform predicted plan $\mathcal{G}_{\mathcal{P}_{\text{pred}}}$ into gold plan $\mathcal{G}_{\mathcal{P}_{\text{gold}}}$, i.e.,

$$d(\mathcal{G}_{\mathcal{P}_{\text{pred}}}, \mathcal{G}_{\mathcal{P}_{\text{gold}}}) = |\mathcal{N}_+| + |\mathcal{N}_-| + |\mathcal{E}_+| + |\mathcal{E}_-|, \tag{2}$$

where $|\mathcal{N}_+|$, $|\mathcal{N}_-|$, $|\mathcal{E}_+|$, and $|\mathcal{E}_-|$ respectively represent the numbers of node and edge additions or deletions. We apply an exponential function to map $d \in [0, +\infty)$ to the range $[0, 1]$ and define the structural score of the predicted plan, i.e.,

$$s_{\textbf{struc}} = \exp\left(-\beta \cdot d(\mathcal{G}_{\mathcal{P}_{\text{pred}}}, \mathcal{G}_{\mathcal{P}_{\text{gold}}})\right), \tag{3}$$

where $\beta \in (0, +\infty)$ can adjust the magnitude of structural changes. Finally, we introduce $\alpha \in [0, 1]$ to balance the semantic and structural scores, i.e.,

$$\text{PSE-P}_{\alpha} = \alpha \cdot s_{\textbf{sem}} + (1 - \alpha) \cdot s_{\textbf{struc}}. \tag{4}$$

Step-answer Evaluation. Unlike plan evaluation, which focuses on abstract semantics, step-answer evaluation concentrates on concrete semantics. Given $\mathcal{M}_{\mathcal{P}}$, we can easily get the paired predicted step-answer and gold step-answer $(a_{\text{pred}}, a_{\text{gold}})$ on an intermediate step. We use the average F1 score of all mapped step-answers as the score of the predicted step-answers, i.e.,

$$\text{PSE-A}_{\phi} = \frac{\sum_{i=1}^{|\mathcal{M}_{\mathcal{P}}|} \phi\left(a_{\text{pred}}^{(i)}, a_{\text{gold}}^{(i)}\right)}{|\mathcal{P}_{\text{gold}}|}, \tag{5}$$

where ϕ represents EM or F1. For fine-grained evaluation, we set $\phi = \text{F1}$ in this paper.

Global Evaluation. The evaluation of intermediate steps needs to comprehensively consider planning (abstract-level) and solving (concrete-level). By incorporating both PSE-P_1 and PSE-A_{ϕ}, we can measure whether the reasoning remains focused without drifting into irrelevant details. We set $\alpha = 1$ to truly reflect the reasoning ability of the MHQA system, rather than its planning ability. Therefore, we use the harmonic mean to measure the global performance, i.e.,

$$\text{PSE-G}_{\phi} = \frac{2 \cdot \text{PSE-P}_1 \cdot \text{PSE-A}_{\phi}}{\text{PSE-P}_1 + \text{PSE-A}_{\phi}}. \tag{6}$$

4 Experiment

4.1 Datasets and Evaluation Metrics

We randomly sample 500 questions of each type from the PER-DP preprocessed data (Table 1) as the test set for evaluation. For MuSiQue Bri.$^{3\text{H}1}$ and Bri.$^{3\text{H}2}$, we sample 400 and 100 of two types respectively. For MuSiQue Bri.$^{4\text{H}1}$, Bri.$^{4\text{H}2}$, and Bri.$^{4\text{H}3}$, we put all types together directly.

We adopt the F1 and PSE-G_{F1} as our evaluation metrics. Considering the applicability of PSE-G_{F1} in different types of questions, we set $\theta^* = 0.7$.

[2] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

	Methods	HotpotQA		2WikiMultihopQA					MuSiQue		
		Bri.^{2H}	Comp.^{2H}	Bri.^{2H}	Comp.^{2H}	Infer.^{2H}	Comp.^{4H}	B.C.^{4H}	Bri.^{2H}	Bri.^{3H}	Bri.^{4H}
Vanilla	Llama-2_{7B}	20.0 | –	54.0 | –	14.1 | –	36.4 | –	23.5 | –	48.8 | –	46.2 | –	9.5 | –	11.7 | –	7.6 | –
	Llama-2_{13B}	24.7 | –	54.3 | –	13.8 | –	47.7 | –	23.2 | –	50.1 | –	52.2 | –	13.6 | –	10.7 | –	8.7 | –
	Llama-3.1_{8B}	25.3 | –	63.1 | –	13.5 | –	55.7 | –	26.9 | –	46.4 | –	50.5 | –	13.6 | –	13.0 | –	9.4 | –
	GPT-4o$_{mini}$	36.3 | –	72.3 | –	21.6 | –	59.2 | –	32.9 | –	50.2 | –	52.0 | –	18.0 | –	17.8 | –	**15.1** | –
RAG	Naïve RAG	35.0 | –	65.7 | –	18.5 | –	46.0 | –	35.6 | –	51.0 | –	31.3 | –	14.1 | –	7.8 | –	10.2 | –
	Self-Ask	**52.7** | –	72.2 | –	**43.9** | –	62.2 | –	49.4 | –	52.4 | –	48.1 | –	**31.1** | –	17.3 | –	13.5 | –
	IRCoT QA	51.2 | –	69.9 | –	39.6 | –	48.1 | –	43.3 | –	44.9 | –	28.5 | –	26.8 | –	14.2 | –	10.3 | –
Ours	PER-QA$_{vanilla}$	23.8 | 23.1	72.3 | 47.5	16.7 | 23.3	64.4 | 27.5	25.1 | 35.8	50.4 | 11.9	49.8 | 17.2	16.8 | 28.8	13.7 | 22.4	10.2 | 17.9
	PER-QA$_{rag}$	47.7 | 44.1	**78.1** | 64.5	42.9 | 58.2	**70.4** | 55.4	**52.2** | 65.2	**56.8** | 76.1	**52.4** | 53.2	30.6 | 48.0	**23.8** | 38.7	15.0 | 26.6

Table 2: Comparison of different methods on ten types of questions (**Left**: F1, **Right**: PSE-G$_{F1}$). "–": score is unavailable. The backbone model of `RAG` methods is Llama3.1-8B-instruct. In PER-QA, only the *planner* uses `gpt-4o-2024-11-20`, and other modules use Llama3.1-8B-instruct. Note that Self-Ask and IRCoT need to predefine the maximum number of iterations. To adapt to all types of questions, we set it to 15.

Methods	HotpotQA	2WikiMQA	MuSiQue
`Vanilla`†	44.2	38.0	12.4
PER-QA$_{vanilla}$	**48.1** (+3.9)	**39.7** (+1.7)	**14.0** (+1.6)
`RAG`	62.5	51.0	21.7
PER-QA$_{rag}$	**62.9** (+0.4)	**54.6** (+3.6)	**24.3** (+2.6)

Table 3: The overall F1 score (%). The best result of baseline is used. "†": only Llama models are included.

4.2 Baselines

We conduct comparisons separately under `Vanilla` and `RAG` settings. (i) `Vanilla`: we prompt LLMs to answer questions directly using few-shot prompting (Brown et al., 2020). We compare Llama2-7B-chat, Llama2-13B-chat, Llama3.1-8B-instruct, and GPT-4o-mini (`gpt-4o-mini-2024-07-18`). GPT-4o-mini has about 8 billion parameters (Abacha et al., 2024). (ii) `RAG`: we reproduce the representative methods of step-by-step reasoning, Self-Ask (Press et al., 2023) and IRCoT QA (Trivedi et al., 2023), on Llama3.1-8B-instruct to demonstrate their performance under low resources. In addition, we also compare Naïve RAG (Gao et al., 2023b) using Llama3.1-8B-instruct. Note that IRCoT QA only uses the iterative CoT for retrieval expansion and the QA is realized via few-shot prompting.

4.3 Main Results

PER-QA achieves competitive MHQA performance. Overall, as shown in Table 3, compared with `Vanilla` (except GPT-4o-mini), PER-QA$_{vanilla}$ gain 3.9%, 1.7%, and 1.6% F1 improvement respectively. Compared with `RAG`, PER-QA$_{rag}$ gain 0.4%, 3.6%, and 2.6% F1 improvement respectively. As shown in Table 2, we can observe that:

(i) In the `Vanilla` setting, PER-QA$_{vanilla}$ outperforms the Llama models in 7 of 10 data types. However, when compared to GPT-4o-mini, it only improves performance in 3 of 10 data types. In cases where there is no performance gain, we attribute this to the fact that this models rely on memory for responses rather than engaging in true reasoning. This becomes more evident as the models' capabilities grow. In contrast, PER-QA$_{vanilla}$ explicitly performs reasoning, which highlights challenges related to both leveraging parametric knowledge and handling multi-hop reasoning.

(ii) In the `RAG` setting, PER-QA$_{rag}$ achieves state-of-the-art F1 performance in 7 of 10 data types, demonstrating its capability to effectively integrate multi-source external information through explicit graph structures for enhanced reasoning. However, we observe F1 performance gaps of 5.0%, 1.0%, and 0.5% on three Bri.^{2H} datasets compared to Self-Ask. This difference is mainly because we set a larger iteration limit for Self-Ask to adapt to various question types, which enables it to perform more detailed decomposition and retrieval, while PER-QA relies on one-time decomposition.

PER-QA and PSE provide stronger MHQA explainability. We can use PSE-G$_{F1}$ to analyze the intermediate step on different types of questions.

(i) As shown in Figure 3 (a), the distribution patterns reveal critical differences: PER-QA$_{rag}$ results mainly cluster in the $y > x$ region and are generally close to the upper-left, while half of PER-QA$_{vanilla}$ results are in the $y < x$ region and are generally close to the lower-right. This indicates that PER-QA$_{rag}$ enhances the accuracy and attribution of intermediate steps by retrieving external

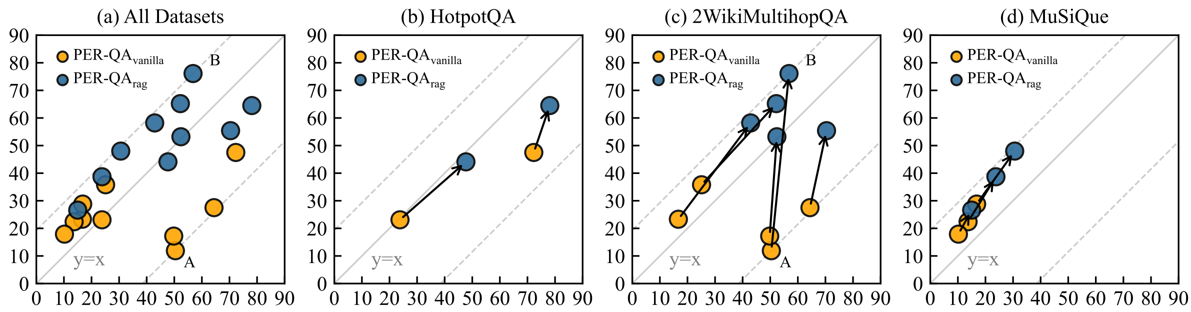


Figure 3: The distribution pattern of PER-QA$_{vanilla}$ results and PER-QA$_{rag}$ results (x-axis: F1, y-axis: PSE-G$_{F1}$). The arrow connects the results on the same dataset (from PER-QA$_{vanilla}$ to PER-QA$_{rag}$). The coordinates of each point are from Table 2. Proximity to the line $y = x$ indicates a better balance between the final answer and intermediate steps. "A" and "B" are the farthest points from $y = x$, and both of them are results on 2WikiMultihopQA Comp.^{4H}. The dashed line is the line passing through points "A" and "B" parallel to $y = x$.

knowledge, but may face incomplete retrieval or wrong reasoning. In contrast, PER-QA$_{vanilla}$, which relies solely on parametric knowledge, may lead to error propagation in the intermediate steps, but could also benefit from memory or guess.

(ii) As shown in Figure 3 (b), (c), and (d), for the same dataset, transitioning from PER-QA$_{vanilla}$ to PER-QA$_{rag}$ results in a shift toward the upper-right corner. This suggests that PER-QA enhances reasoning in intermediate steps through retrieval, which in turn strengthens the reasoning for the final answer. We also observe some extreme cases, such as ΔF1 is 6.4% between points "A" and "B" in Figure 3 , which is much smaller than the change in ΔPSE-G$_{F1}$ is 64.2%. We believe this is related to the type of question. For comparison questions, where the answer is either "yes/no" or "span answer", even if the model provides incorrect or inaccurate intermediate reasoning steps, it still has a high probability of hitting the correct final answer. In contrast, for bridge questions, ΔF1 and ΔPSE-G$_{F1}$ are more consistent. For example, in Table 2, ΔF1 and ΔPSE-G$_{F1}$ for HotpotQA Bri.^{2H} are 23.9% and 21.0%, respectively.

4.4 Sub-questions Analysis

The performance of PER-QA on the first sub-question matches gold performance, but subsequent ones are vulnerable to misinformation transmission. As can be seen from the Figure 4, on q_1, the performance of PER-QA$_{rag}$ is close to Gold-QA$_{rag}$ (↓7.3%, ↑1.3%, ↓1.1%). However, on q_2, there is a certain gap between the performance of PER-QA$_{rag}$ and Gold-QA$_{rag}$ (↓26.0%, ↓11.7%, ↓18.4%). This is because in open-domain setting, MHQA often has no independently existing gold

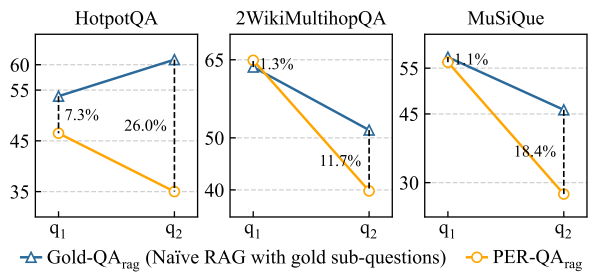


Figure 4: The sub-question answering F1 performance of Bri.^{2H} on three datasets. Gold-QA$_{rag}$ independently executes RAG for each sub-question. PER-QA$_{rag}$ executes RAG sequentially (the formation of the second sub-question depends on the previous reasoning step).

sub-questions and it needs to rely on the answer of the previous step to produce the next sub-question.

4.5 Human Evaluation

PER-DP provides high-quality preprocessed data to support fine-grained reasoning and evaluation. We evaluate the quality of the preprocessed data through human or programmatic evaluation, referring to the question decomposition strategies or supporting passages of the original dataset. As shown in Table 4 (a), we achieve a plan matching of over 90% and a step-answer matching close to 90% on all three datasets. On 2WikiMultihopQA, these two matching are as high as 99.2% and 97.1% respectively. This indicates that the preprocessed data can well preserve reasonable intermediate reasoning steps.

PSE is a more reasonable evaluation method except for human evaluation. We evaluate the intermediate steps on three datasets through manual and automatic evaluation to confirm the rational-

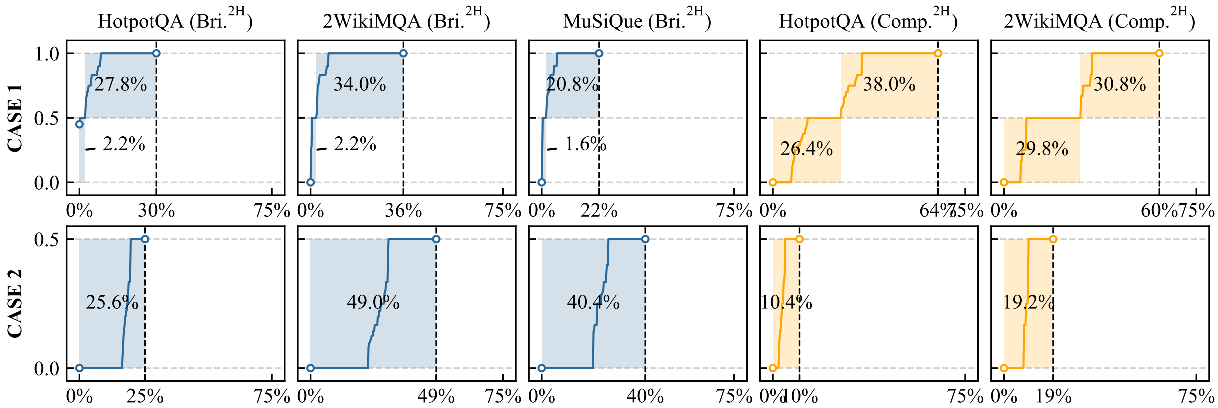


Figure 5: PER-QA reasoning results with complete intermediate reasoning steps (x-axis: the proportion of the sample in the entire test set, y-axis: PSE-A_{F1}). **CASE 1** (the first row): the final answer is completely correct. **CASE 2** (the second row): the final answer is completely wrong. We only consider samples with completely correct plans. In our analysis, we regard $0.5 < \text{PSE-A}_{F1} \leq 1$ as all intermediate reasoning steps being correct, and $0 \leq \text{PSE-A}_{F1} \leq 0.5$ as an error occurring in the intermediate reasoning steps.

Setting	HotpotQA	2WikiMQA	MuSiQue
(a) Human Eval. for PER-DP			
Plan (HE)	92.5	99.2	94.0
Step. (HE)	89.8	97.1†	88.6†
(b) Human Eval. & PSE on PER-QA$_{rag}$ (Plan)			
Plan (HE)	90.3	97.7	89.7
Plan (PSE)	87.3 (↓3.0%)	95.3 (↓2.4%)	78.3 (↓11.4%)
(c) Human Eval. & PSE on PER-QA$_{rag}$ (Step-answer)			
Step. (HE)	68.8	54.3	33.8
Step. (PSE)	55.8 (↓13.0%)	53.3 (↓1.0%)	31.1 (↓2.7%)

Table 4: Human evaluation on the preprocessed data (PER-DP) and the intermediate reasoning steps (PER-QA$_{rag}$). "†": the original dataset contains ground-truth and is verified by programs.

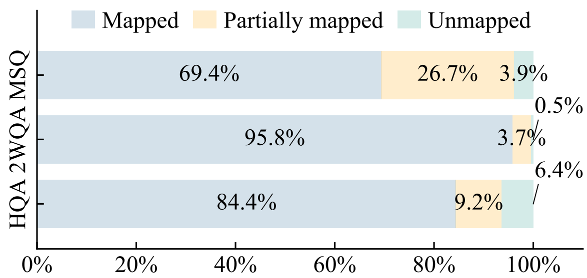


Figure 6: The mapping of plans on three datasets.

ity of PSE. For the plan evaluation, as shown in Table 4 (b), on HotpotQA and 2WikiMultihopQA, PSE is close to the human evaluation performance, with a gap of only 3.0% and 2.4%, respectively. On MuSiQue, the evaluation performance gap has increased to 11.4%. We believe this is due to the complexity of the 3-hop and 4-hop questions in MuSiQue, which leads to the failure of planning mapping. For the step-answer evaluation, as shown in Table 4 (c), on 2WikiMultihopQA (↓1.0%) and MuSiQue (↓2.7%), PSE is closer to human evaluation than on HotpotQA (↓13.0%). We believe this gap is due to the complexity of natural language and the influence of overly strict ground-truth. For example, for *"McComb, Mississippi"* (ground-truth) and *"McComb"* (prediction), human evaluation will count a missed answer as completely correct, while F1 will count it as partially wrong. For *"Berthold, Margrave of Baden"* (ground-truth) and *"William, Margrave of Baden"* (prediction), human evaluation will count a wrong answer as completely wrong, while F1 will count it as partially correct.

4.6 Error Analysis

Despite planning mapping failures that may hinder PSE's evaluation of intermediate steps, it effectively evaluates over 93.6% of samples. Figure 6 quantitatively analyzes the planning mapping on three datasets. We find that only 6.4%, 0.5%, and 3.9% of the samples on the three datasets cannot be evaluated. This may be because the *planner* of PER-QA is more generalizable and does not restrict the question decomposition strategy, while the structural and semantic limitations of the plan mapping algorithm prevent PSE from capturing such plans. In addition, PSE can avoid the problem of completely ignoring the evaluation of valuable intermediate reasoning steps due to partial structural or semantic differences. Case studies of qualitative analysis can be found in Appendix E.2.

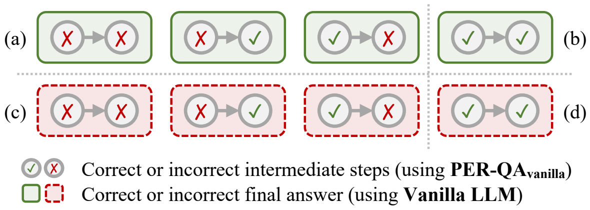


Figure 7: All possible reasoning situations regarding the 2-hop bridge question. Among them, only area (a) represents the potentially contaminated data.

4.7 Further Analysis

The RAG-based MHQA system is experiencing "fortuitous reasoning continuance" and "latent reasoning suspension". As shown in Figure 5 **CASE 1**, we find that 26.4% and 29.8% of comparison questions are answered correctly even when the intermediate reasoning step is erroneous, while this proportion is very small for bridge questions, only 2.2%, 2.2%, and 1.6%. We call this phenomenon *"fortuitous reasoning continuance"*, which means that even if there are wrong intermediate steps in the reasoning process, it still does not affect the final answer of multi-hop reasoning. This phenomenon often occurs in comparison questions. We believe this is because comparison questions are highly inclusive of factual deviations. As long as the two objects being compared maintain consistency in the direction of comparison regarding correct or incorrect facts, the reasoning can be continued and the correct final answer can be obtained.

Additionally, as shown in Figure 5 **CASE 2**, we find that a considerable proportion of the bridge questions, as high as 49.0%, has complete intermediate reasoning steps even though these steps are incorrect. We named this phenomenon *"latent reasoning suspension"*, which means that even if there are incorrect intermediate reasoning steps in the reasoning process, multi-hop reasoning does not terminate. This phenomenon often occurs in bridge questions. This may be due to the faithfulness of the RAG-based LLM to external knowledge, causing semantically similar but irrelevant passages retrieved to be used by the LLM to answer sub-questions (Wu et al., 2024). This can affect the retrieval and reasoning of subsequent sub-questions, making them meaningless, as the correct reasoning direction has been deviated from.

PER-QA and PSE also have application potential in other tasks, such as Data Contamination. Data contamination occurs when LLMs directly access parametric answers ("memory") of existing test sets (Li et al., 2024; Samuel et al., 2025; Sainz et al., 2024). The goal of contamination analysis is to classify the samples as "clean" or "dirty", thereby evaluating the extent to which the datasets has been contaminated (Li et al., 2024). We define "contaminated data" as follows: Under the `Vanilla` setting, if the final answer obtained by directly answering a multi-hop question without decomposition is correct, but the intermediate steps of answering through decomposition are incorrect, then this question is contaminated (Figure 7). Table 5 shows that: for the same dataset, the contamination on GPT-4o-mini is higher than that on Llama models; for the same model, as the difficulty of the dataset increases, the contamination decreases in turn. For more details and discussions, see Appendix F.4.

Models	HotpotQA	2WikiMQA	MuSiQue
Llama-2$_{7B}$	11.3%	7.3%	2.3%
Llama-2$_{13B}$	13.1%	7.0%	3.7%
Llama-3.1$_{8B}$	13.5%	6.5%	4.8%
GPT-4o$_{mini}$	16.7%	13.5%	5.0%

Table 5: The proportion of contaminated data on different datasets and models (only consider Bri.^{2H}).

5 Conclusion

In this paper, we focus on the intermediate reasoning rather than the final answer. We propose the "Planner-Executor-Reasoner" (PER) architecture and construct PER-DP and PER-QA to provide data and reasoning methods with explicit reasoning graphs, supporting fine-grained evaluation. Central to our approach is PSE, which rigorously quantifies the intermediate reasoning. Experiments show that PER-QA and PSE not only provide competitive performance and stronger explainability, but also reveal "fortuitous reasoning continuance" and "latent reasoning suspension" in the RAG-based MHQA system. In addition, we demonstrate potential in data contamination scenarios.

Limitations

The evaluation of existing LLM-based MHQA systems mainly focuses on the final answer rather than the intermediate reasoning. In this work, our PSE evaluates the intermediate reasoning and final answer of MHQA from a fine-grained perspective. However, our PSE aligns ground-truth and prediction based on an explicit graph structure, which may be limited to other MHQA systems (e.g., IRCoT). Considering that our method can provide clear intermediate reasoning evaluation and provides a data preprocessing method (PER-DP) and a reasoning method (PER-QA) as supplements, we believe this limitation is acceptable.

Acknowledgments

This work is supported by Xiamen Major Science and Technology Project (No. 3502Z20241028). We also thank Dr. Qinggang Zhang from the Hong Kong Polytechnic University. His valuable guidance and continuous support have been essential to this research.

References

Asma Ben Abacha, Wen-wai Yim, Yujuan Fu, Zhaoyi Sun, Meliha Yetisgen, Fei Xia, and Thomas Lin. 2024. MEDEC: A benchmark for medical error detection and correction in clinical notes. *CoRR*, abs/2412.19260.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Jifan Chen and Greg Durrett. 2019. Understanding dataset design choices for multi-hop reasoning. 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 4026–4032, Minneapolis, Minnesota. Association for Computational Linguistics.

Dheeru Dua, Shivanshu Gupta, Sameer Singh, and Matt Gardner. 2022. Successive prompting for decomposing complex questions. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1251–1265, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023a. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1762–1777, Toronto, Canada. Association for Computational Linguistics.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. 2023b. Retrieval-augmented generation for large language models: A survey. *CoRR*, abs/2312.10997.

Jie He, Nan Hu, Wanqiu Long, Jiaoyan Chen, and Jeff Z. Pan. 2024. MINTQA: A multi-hop question answering benchmark for evaluating llms on new and tail knowledge. *CoRR*, abs/2412.17032.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. 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, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jiajie Jin, Yutao Zhu, Xinyu Yang, Chenghao Zhang, and Zhicheng Dou. 2024. FlashRAG: A modular toolkit for efficient retrieval-augmented generation research. *CoRR*, abs/2405.13576.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2023. Decomposed prompting: A modular approach for solving complex tasks. In *The Eleventh International Conference on Learning Representations*. OpenReview.net.

Yucheng Li, Yunhao Guo, Frank Guerin, and Chenghua Lin. 2024. An open-source data contamination report for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 528–541, Miami, Florida, USA. Association for Computational Linguistics.

Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5303–5315, Singapore. Association for Computational Linguistics.

Vaibhav Mavi, Anubhav Jangra, Adam Jatowt, et al. 2024. Multi-hop question answering. *Foundations and Trends® in Information Retrieval*, 17(5):457–586.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. Compositional questions do not necessitate multi-hop reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4249–4257, Florence, Italy. Association for Computational Linguistics.

Pruthvi Patel, Swaroop Mishra, Mihir Parmar, and Chitta Baral. 2022. Is a question decomposition unit all we need? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4553–4569, Abu Dhabi, United

Arab Emirates. Association for Computational Linguistics.

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

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Oscar Sainz, Iker García-Ferrero, Alon Jacovi, Jon Ander Campos, Yanai Elazar, Eneko Agirre, Yoav Goldberg, Wei-Lin Chen, Jenny Chim, Leshem Choshen, et al. 2024. Data contamination report from the 2024 CONDA shared task. In *Proceedings of the 1st Workshop on Data Contamination (CONDA)*, pages 41–56, Bangkok, Thailand. Association for Computational Linguistics.

Vinay Samuel, Yue Zhou, and Henry Peng Zou. 2025. Towards data contamination detection for modern large language models: Limitations, inconsistencies, and oracle challenges. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 5058–5070, Abu Dhabi, UAE. Association for Computational Linguistics.

Zhengliang Shi, Shuo Zhang, Weiwei Sun, Shen Gao, Pengjie Ren, Zhumin Chen, and Zhaochun Ren. 2024. Generate-then-ground in retrieval-augmented generation for multi-hop question answering. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7339–7353, Bangkok, Thailand. Association for Computational Linguistics.

Yixuan Tang, Hwee Tou Ng, and Anthony Tung. 2021. Do multi-hop question answering systems know how to answer the single-hop sub-questions? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3244–3249, Online. Association for Computational Linguistics.

Yixuan Tang and Yi Yang. 2024. Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries. *CoRR*, abs/2401.15391.

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

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. 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, Toronto, Canada. Association for Computational Linguistics.

Prakhar Verma, Sukruta Prakash Midigeshi, Gaurav Sinha, Arno Solin, Nagarajan Natarajan, and Amit Sharma. 2025. Plan*rag: Efficient test-time planning for retrieval augmented generation. *CoRR*, abs/2410.20753.

Cunxiang Wang, Sirui Cheng, Qipeng Guo, Yuanhao Yue, Bowen Ding, Zhikun Xu, Yidong Wang, Xiangkun Hu, Zheng Zhang, and Yue Zhang. 2023. Evaluating open-qa evaluation. In *Advances in Neural Information Processing Systems*, volume 36, pages 77013–77042. Curran Associates, Inc.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *CoRR*, abs/2212.03533.

Minzheng Wang, Longze Chen, Fu Cheng, Shengyi Liao, Xinghua Zhang, Bingli Wu, Haiyang Yu, Nan Xu, Lei Zhang, Run Luo, Yunshui Li, Min Yang, Fei Huang, and Yongbin Li. 2024. Leave no document behind: Benchmarking long-context LLMs with extended multi-doc QA. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 5627–5646, Miami, Florida, USA. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Jian Wu, Linyi Yang, Zhen Wang, Manabu Okumura, and Yue Zhang. 2025. CofCA: A STEP-WISE counterfactual multi-hop QA benchmark. In *The Thirteenth International Conference on Learning Representations*.

Siye Wu, Jian Xie, Jiangjie Chen, Tinghui Zhu, Kai Zhang, and Yanghua Xiao. 2024. How easily do irrelevant inputs skew the responses of large language models? In *First Conference on Language Modeling*.

Amy Xin, Jinxin Liu, Zijun Yao, Zhicheng Li, Shulin Cao, Lei Hou, and Juanzi Li. 2024. Atomr: Atomic operator-empowered large language models for heterogeneous knowledge reasoning. *CoRR*, abs/2411.16495.

Sohee Yang, Elena Gribovskaya, Nora Kassner, Mor Geva, and Sebastian Riedel. 2024a. Do large language models latently perform multi-hop reasoning? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10210–10229, Bangkok, Thailand. Association for Computational Linguistics.

Sohee Yang, Nora Kassner, Elena Gribovskaya, Sebastian Riedel, and Mor Geva. 2024b. Do large language models perform latent multi-hop reasoning without exploiting shortcuts? *CoRR*, abs/2411.16679.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. 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, Brussels, Belgium. Association for Computational Linguistics.

Fei Yu, Hongbo Zhang, Prayag Tiwari, and Benyou Wang. 2024. Natural language reasoning, A survey. *ACM Computing Surveys*, 56(12):1–39.

Qinggang Zhang, Shengyuan Chen, Yuanchen Bei, Zheng Yuan, Huachi Zhou, Zijin Hong, Junnan Dong, Hao Chen, Yi Chang, and Xiao Huang. 2025. A survey of graph retrieval-augmented generation for customized large language models. *arXiv preprint arXiv:2501.13958*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, et al. 2023. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*. OpenReview.net.

Yujia Zhou, Zheng Liu, Jiajie Jin, Jian-Yun Nie, and Zhicheng Dou. 2024. Metacognitive retrieval-augmented large language models. In *Proceedings of the ACM on Web Conference 2024, WWW 2024, Singapore, May 13-17, 2024*, pages 1453–1463. ACM.

Andrew Zhu, Alyssa Hwang, Liam Dugan, and Chris Callison-Burch. 2024. FanOutQA: A multi-hop, multi-document question answering benchmark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 18–37, Bangkok, Thailand. Association for Computational Linguistics.

Ziyuan Zhuang, Zhiyang Zhang, Sitao Cheng, Fangkai Yang, Jia Liu, Shujian Huang, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Qi Zhang. 2024. EfficientRAG: Efficient retriever for multi-hop question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3392–3411, Miami, Florida, USA. Association for Computational Linguistics.

A Dataset Information

We consider ten types of multi-hop questions from HotpotQA (Yang et al., 2018), 2WikiMultihopQA (Ho et al., 2020), and MuSiQue (Trivedi et al., 2022). The dataset we use in the experiment is processed by PER-DP. We design the prompts for the three agents in PER-DP according to the **Known Constraints** of the original dataset. Known constraints include structure constraint, evidence constraint and passage constraint. Examples of ten question types are shown in Table 19, and the prompts of the *planner* are shown in Table 20.

Structure Constraint. The structure of multi-hop questions is explicitly stated or implied in the original dataset. The structure constraint limits the types of multi-hop questions and the number of sub-questions. The types include **Bridge** (Bri.), **Comparison** (Comp.), **Inference** (Infer.), and **Bridge-Comparison** (B.C.), and the number of sub-questions ranges from **2 to 4**. Notably, 2WikiMultihopQA introduces inference-type, which decomposes questions based on logical rules (e.g., “father-in-law”=“spouse”+“father”). This type is actually a kind of bridge-type.

Evidence Constraint. 2WikiMultihopQA and MuSiQue provide explicit question decomposition (including sub-questions and answers), while HotpotQA does not. We refer to explicit question decomposition as evidence, which specifies the unique decomposition format of the question. The forms of evidence for 2WikiMultihopQA and MuSiQue are shown in Table 6.

Passage Constraint. The MHQA datasets specifiy the gold passages (i.e., the document fragments containing the answers to sub-questions) required to answer the multi-hop question. Since HotpotQA does not provide evidence, we use gold passages as hints for the *planner*.

An example in 2WikiMultihopQA.
Question: When did John V, Prince Of Anhalt-Zerbst's father die? *(2-hop bridge question)* **Question Decomposition:** (("John V of Anhalt-Zerbst", "father", "Ernest I, Prince of Anhalt-Dessau"), ("Ernest I, Prince of Anhalt-Dessau", "date of death", "12 June 1516"))
An example in MuSiQue.
Question: What league does the team that plays in Stadio Ciro Vigorito play for? *(2-hop bridge question)* **Question Decomposition:** (("Stadio Ciro Vigorito » occupant", "Benevento Calcio"), ("1 » league", "Lega Pro Prima Divisione"))

Table 6: Examples of 2-hop bridge question decomposition in 2WikiMultihopQA and MuSiQue.

B Environment Settings

BM25 (Robertson et al., 2009) is used as the retriever and $E5_{base}$ (Wang et al., 2022) is employed as the reranker, and Wikipedia dump from December 2018 as the corpus. We retrieve the top 100 and select the top 5 after reranking. All methods

	Question Type	Input Volume	Output Volume	Discard Rate (%)
HQA	Bri.^{2H}	1,483	1,166	21.38
	Comp.^{2H}	1,237	1,052	14.96
2WikiMQA	Bri.^{2H}	1,210	1,036	14.38
	Comp.^{2H}	1,041	1,011	2.88
	Infer.^{2H}	1,051	1,013	3.62
	Comp.^{4H}	126	125	0.79
	B.C.^{4H}	1,027	1,019	0.78
MuSiQue	Bri.^{2H}	1,132	1,018	10.07
	Bri.^{3H1}	568	478	15.85†
	Bri.^{3H2}	192	163	15.10†
	Bri.^{4H1}	246	230	6.50†
	Bri.^{4H2}	64	31	51.56†
	Bri.^{4H3}	95	52	45.26†

Table 7: Overview of data preprocessing. PER discards some samples that fail to pass quality control while preprocessing data. "†": the discard rate in 15th iteration.

are implemented by FlashRAG (Jin et al., 2024). For PSE, we set $\theta^* = 0.7$. All experiments are conducted on 8 NVIDIA RTX 3090 24GB GPUs.

C More Details about PER-DP

C.1 Implementation Details

The core model for all agents in PER-DP is GPT-4o (`gpt-4o-2024-11-20`). We set the maximum number of iterations for each sample to be 5 or 15. For the 3-hop and 4-hop questions of MuSiQue, due to their excessive complexity, humans may also find it difficult to answer correctly. Therefore, we set the maximum number of iterations for such types to be 15, and the maximum number of iterations for other types is 5. In addition, in the first iteration, we set the temperature to 0, and from the second iteration, we set the temperature to 1. See Table 20, 22, and 23 for prompts.

The original data files we use are: (i) HotpotQA[3]: `hotpot_dev_distractor_v1.json`. (ii) 2WikiMultihopQA[4]: `dev.json` and `id_aliases.json`. (iii) MuSiQue[5]: `musique_ans_v1.0_dev.json`.

C.2 Preprocessed Data Statistics

A total of 8,394 of data that have been preprocessed by PER-DP and successfully retained. More detailed data statistics are shown in Table 7.

As shown in Table 7, the discard rates of HotpotQA's Bri.^{2H} and MuSiQue's Bri.^{4H2} and Bri.^{4H3} exceed 20%. The former may be caused by potential data leakage. Even if it is defined as a multi-hop question, it can still be solved by a single-hop question (Chen and Durrett, 2019; Min et al., 2019). The latter, due to the strong logic relied on when constructing the original dataset (Trivedi et al., 2022), further amplifies the complexity of question decomposition and solution, thus resulting in an excessively high discard rate.

[3]https://hotpotqa.github.io
[4]https://github.com/Alab-NII/2wikimultihop
[5]https://github.com/stonybrooknlp/musique

D More Details about PER-QA

D.1 Implementation Details

The core model of *planner* in PER-QA is GPT-4o (`gpt-4o-2024-11-20`). The core models of *executor* and *reasoner* adopt Llama3.1-8B-instruct. The *planner* will first conduct coarse-grained type classification (Bridge or Comparison) and select different prompts according to the types. Under the `RAG` setting, we set that the *executor* can retrieve 5 passages each time. The *reasoner* by default adopts the few-shot prompting and directly answers by taking the linearized context graph as the context. See Table 21, 22, and 23 for prompts. However, since our method decouples retrieval and reasoning, we can conveniently adopt stronger prompting methods such as CoT prompting (Wei et al., 2022).

D.2 CoT Prompting for Reasoner

We replace direct answering with CoT prompting in a question type (2WikiMultihopQA Comp.^{4H}) that requires mathematical reasoning (an example is shown in Table 8). As shown in Table 9, using CoT improves the F1 performance by 13.1%.

Question: Who lived longer, Catherine Isabella Osborne or David Bierens De Haan? *(4-hop Comparison)*

Question Decomposition:
1. When was Catherine Isabella Osborne born?
2. When did Catherine Isabella Osborne die?
The lifespan of Catherine Isabella Osborne is:
`Date(21,June,1880) - Date(30,June,1818)`
3. When was David Bierens de Haan born?
4. When did David Bierens de Haan die?
The lifespan of David Bierens de Haan is:
`Date(12,August,1895) - Date(3,May,1822)`

Table 8: An example of 4-hop comparison question decomposition in 2WikiMultihopQA.

D.3 Different Backbone of PER-QA

In PER-QA, only the *planner* uses GPT-4o to seek better planning ability. Other modules adopt Llama3.1-8B-instruct. In fact, all modules can

Methods	F1	PSE-G_{F1}
PER-QA_{rag}	56.8	76.1
+ CHAIN-OF-THOUGHT	69.9	76.1

Table 9: Performance of using direct answering and CoT prompting on 2WikiMultihopQA Comp.[4H].

be replaced with any open-source or commercial model (e.g., Qwen and Deepseek). Table 10 shows the F1 and PSE-G_{F1} performance of three representative types under different models.

As can be seen from the Table 10, PER-QA can be well adapted to open-source models. Using Llama3.1-8B-instruct as the *planner*, PER-QA_{rag} can still maintain competitive performance compared to existing RAG methods. The performance can be further improved by tuning prompts.

E More Details about PSE

E.1 Implementation Details

We use Sentence-Transformer (`all-MiniLM-L6-v2`) as an encoder and adopt the cosine similarity of vectors as semantic similarity. We set the semantic similarity threshold θ^* to 0.7 so as to be able to adapt to various types of questions.

Note that we calculate PSE-G_ϕ based on PSE-P_1. Here, $\alpha = 1$ because when conducting global evaluation, we only want to consider the mapped sub-plans to reflect the most accurate reasoning ability rather than the planning ability.

E.2 Plan Mapping Algorithm

We achieve the plan mapping through Algorithm 1 to facilitate subsequent plan evaluation and step-answer evaluation. Essentially, we achieve the alignment of two graphs with semantics on nodes.

The planning mapping algorithm may fail to capture the mapping relationship between the gold plan and the predicted plan in some cases, such as alternative planning decomposition strategies (the case is shown in Table 11) or overly strict semantic constraints (discussion about θ^* in Appendix E.3).

E.3 Hyperparameter Analysis

The impact of β on s_{struc}. As illustrated in Figure 8, β influences the rate of decay in Equation 3. The smaller the β, the higher the tolerance for structural inconsistency. While the larger the β is, the structural score with greater inconsistency will approach 0 and it will be difficult to distinguish. We set $\beta = 0.1$ to ensure differentiation.

Algorithm 1: Plan Mapping.

Input: Predicted Plan $\mathcal{G}_{\mathcal{P}_{\text{pred}}} = \{\mathcal{P}_{\text{pred}}, \mathcal{E}_{\mathcal{P}_{\text{pred}}}\}$,
Gold Plan $\mathcal{G}_{\mathcal{P}_{\text{gold}}} = \{\mathcal{P}_{\text{gold}}, \mathcal{E}_{\mathcal{P}_{\text{gold}}}\}$,
Similarity Threshold θ^*.
Output: Mapping $\mathcal{M}$ between $\mathcal{P}_{\text{pred}}$ and $\mathcal{P}_{\text{gold}}$.

Function `map_next`($p_{\text{gold}}, p_{\text{pred}}, \mathcal{G}_{\mathcal{P}_{gold}}, \mathcal{G}_{\mathcal{P}_{pred}}, \mathcal{M}'$):
  Find the p_{gold}'s successor p'_{gold} on $\mathcal{G}_{\mathcal{P}_{\text{gold}}}$;
  Find the p_{pred}'s successor p'_{pred} on $\mathcal{G}_{\mathcal{P}_{\text{pred}}}$;
  if p'_{gold} *is* None *or* p'_{pred} *is* None **then**
    return
  end
  if $\text{sim}(p'_{\text{pred}}, p'_{\text{gold}}) \geq \theta^*$ **then**
    $\mathcal{M}'[p'_{\text{gold}}] \leftarrow p'_{\text{pred}}$;
    `map_next`($p'_{\text{gold}}, p'_{\text{pred}}, \mathcal{G}_{\mathcal{P}_{gold}}, \mathcal{G}_{\mathcal{P}_{pred}}, \mathcal{M}'$);
  end
Initialize: $\mathcal{M} \leftarrow \emptyset, \mathcal{M}' \leftarrow \emptyset$;
Get 0-depth sub-plans: $\mathcal{P}^{(0)}_{\text{gold}} \subset \mathcal{P}_{\text{gold}}, \mathcal{P}^{(0)}_{\text{pred}} \subset \mathcal{P}_{\text{pred}}$;
foreach $p_{\text{gold}} \in \mathcal{P}^{(0)}_{\text{gold}}$ **do**
  $p^*_{\text{pred}} \leftarrow \max_{p_{\text{pred}} \in \mathcal{P}_{\text{pred}}} \text{sim}(p_{\text{pred}}, p_{\text{gold}})$;
  if $p^*_{\text{pred}} \in \mathcal{P}^{(0)}_{\text{pred}}$ *and* $\text{sim}(p^*_{\text{pred}}, p_{\text{gold}}) \geq \theta^*$ **then**
    $\mathcal{M}[p_{\text{gold}}] \leftarrow p^*_{\text{pred}}$;
  end
end
if $\mathcal{M} \neq \emptyset$ **then**
  foreach $(p_{\text{gold}}, p_{\text{pred}}) \in \mathcal{M}$ **do**
    `map_next`($p_{\text{gold}}, p_{\text{pred}}, \mathcal{G}_{\mathcal{P}_{gold}}, \mathcal{G}_{\mathcal{P}_{pred}}, \mathcal{M}'$);
  end
end
$\mathcal{M} \leftarrow \mathcal{M} \cup \mathcal{M}'$;
return $\mathcal{M}$;

The impact of α on PSE-P_α. α controls the balance between the semantic score and the structural score. As defined in Equation 4, a higher α assigns greater importance to the semantic alignment, while a lower α shifts the focus towards the structural consistency. In the stage of global evaluation, our primary focus is on the semantic similarity of the global planning and the answering quality of the intermediate steps. Therefore, when calculating PSE-P_α, we set $\alpha = 1$, i.e., PSE-P_1, thereby giving exclusive consideration to the semantic score and disregarding the structural score.

The impact of θ^* on PSE-P_α. We analyze the similarity threshold θ^*, which plays a crucial role in plan mapping. To determine an appropriate θ^*, we conduct experiments across a range from 0 to 1, with $\alpha = 1$ during the evaluation. As shown in Figure 9, lower θ^* allow more data points to pass the threshold, resulting in easier plan alignment and higher PSE-G_{F1} scores. In contrast, when θ^* is large, even approaching 1, the threshold imposes

Backbone of PER-QA$_{rag}$	2WQA Bri.^{2H}		2WQA Comp.^{2H}		MSQ Bri.^{3H}	
	F1	PSE-G$_{F1}$	F1	PSE-G$_{F1}$	F1	PSE-G$_{F1}$
Llama3.1-8B-instruct	41.3	55.8	68.6	54.9	17.9	31.6
w GPT-4o *planner*	**42.9**	**58.2**	**70.4**	**55.4**	**23.8**	**38.7**
Qwen2.5-7B-instruct	35.7	48.2	63.0	55.5	15.5	28.2
w GPT-4o *planner*	**40.5**	**58.2**	**63.2**	**55.9**	**19.2**	**33.1**
DeepSeek-R1-Distill-Llama-8B	23.2	35.2	40.3	42.5	7.8	11.9
w GPT-4o *planner*	**30.7**	**47.4**	**42.7**	**49.3**	**12.6**	**25.9**

Table 10: Experimental results of using different models (Llama, Qwen, and Deepseek) as backbones for three representative question-types. "*w* GPT-4o *planner*" represents the use of GPT-4o as the backbone of the *planner*. Otherwise, the *planner*, *executor*, and *reasoner* all use the same backbone.

Sub-questions	Plan & Step-answer (PER-QA$_{rag}$)	Plan & Step-answer (Ground-truth)	Sub-plan Similarity ($\theta^* = 0.7$)
q_1	What 2014 film was created by Phoebe Ruguru? **Saidia**	What smartphone did Phoebe Ruguru use to create her 2014 film? **iPhone 4s**	0.78 > 0.70 **(Mapped)**
q_2	What smartphone was used to create **Saidia**? **iPhone 4s**	What company marketed **iPhone 4s**? **Apple Inc.**	0.45 < 0.70 **(Unmapped)**
q_3	What company marketed **iPhone 4s**? **Apple Inc.**	/	/

Table 11: A qualitative case on the analysis of plan mapping errors. Original multi-hop question: *"Phoebe Ruguru created a 2014 film on a smartphone marketed by what company?"*. The sub-question decompositions of ground-truth and prediction are both reasonable. However, the mapping fails due to misalignment of plans.

stricter requirements for plan mapping, necessitating near-perfect alignment. This results in lower PSE-G$_{F1}$ scores due to the increased difficulty in achieving such exact matches. Based on this, we ultimately select $\theta^* = 0.7$. When the semantic similarity score between two plans is lower than this threshold, we consider the plans to convey different meanings. For instance, consider the following two sub-plans: *"What county is Jackson Township located in?"* and *"Which administrative territorial entity does Jackson Township belong to?"*. Despite their different phrasing, the plans have a semantic similarity score of 0.81, demonstrating that $\theta^* = 0.7$ effectively identifies semantically equivalent sentences. This threshold achieves a balance between stringency and flexibility, ensuring that plans with different meanings are accurately distinguished while allowing for some robustness in cases where the semantic alignment is not perfect but reasonable.

E.4 Analysis of the Structural Score

As shown in Table 12, we analyze the structural scores of plans and the evaluation time of plan alignment on three datasets. We observe that: The average PSE-P$_0$ (i.e., s_{struc}) scores the highest on 2WikiMultihopQA, at 99.11%, while on HotpotQA

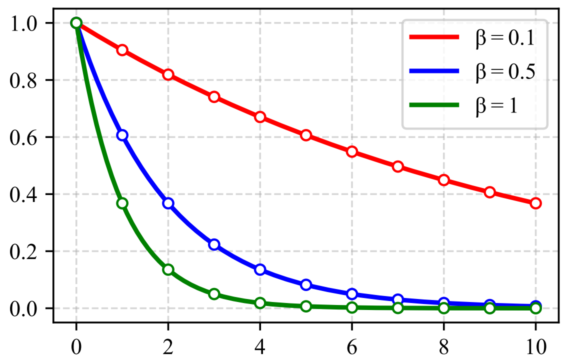


Figure 8: The change of $y = e^{-\beta \cdot x}$ under different β.

and MuSiQue, the average scores are 91.26% and 79.67% respectively. We believe this is caused by the construction of the original dataset. 2WikiMultihopQA uses templates to construct the dataset, and its format is the clearest. HotpotQA is constructed through crowdsourcing, and its format is slightly chaotic. Although MuSiQue is constructed based on graph, due to the relatively complex 3-hop or 4-hop questions, the difficulty of question decomposition is also relatively large. In addition, it takes ≈ 16.32ms to evaluate a single sample.

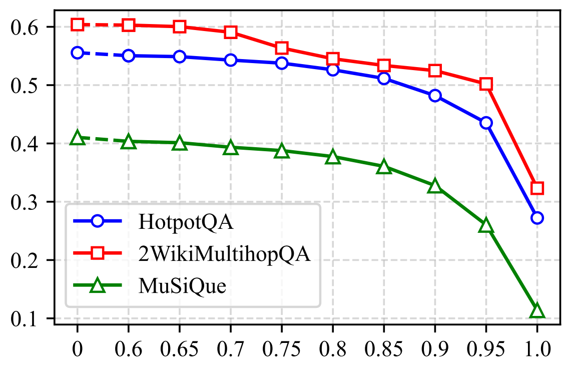


Figure 9: The average PSE-G_{F1} score of three datasets under different θ^* (x-axis: θ^*, y-axis: PSE- G_{F1}). When $\theta^* = 1$, it means that the predicted atomic plan and the gold atomic plan are completely consistent.

	Question Type	PSE-P_0 (%)	Eval. Volume	Eval. Time (s)
HQA	Bri.^{2H}	87.37	500	7.48
	Comp.^{2H}	95.14	500	7.51
2WQA	Bri.^{2H}	100.00	500	7.11
	Comp.^{2H}	97.20	500	7.12
	Infer.^{2H}	98.49	500	7.32
	Comp.^{4H}	100.00	125	2.48
	B.C.^{4H}	99.84	500	10.70
MSQ	Bri.^{2H}	94.60	500	7.03
	Bri.^{3H}	81.29	500	8.02
	Bri.^{4H}	63.12	313	7.64

Table 12: The plan structural scores (PSE-P_0, i.e., s_{struc}) and evaluation times on three datasets. We set $\theta^* = 0.7$. The evaluation time is for the entire test set.

F Case Studies and Discussions

F.1 The Reasoning Processes of Different Multi-hop QA Systems

As shown in Table 18, we compare the reasoning processes of Self-Ask (Press et al., 2023), IRCoT QA (Trivedi et al., 2023), and our method. We choose a 2-hop bridge question *"Which country the director of film Littlerock (Film) is from?"* as an example. The case shows that: (i) Our reasoning process is clearer, decoupling the planning part and the question-answering part, and decoupling the retrieval part and the reasoning part, ensuring the explainability of MHQA to the greatest extent. (ii) Compared with IRCoT QA, in the reasoning stage, we do not need to input all retrievals, but only need to input the evidence. Too many retrievals in the reasoning stage may cause the attention of the LLM to be dispersed and lead to incorrect answers.

F.2 Fortuitous Reasoning Continuance

We define *fortuitous reasoning continuance* as: in multi-hop reasoning, even if there are wrong intermediate steps in the reasoning process, it still does not affect the final answer of multi-hop reasoning. This phenomenon often occurs in comparison questions. We present two examples on Bridge and Comparison types in Table 13 and 14.

As shown in Table 13, in the reasoning process of bridge questions, incorrect entities will be passed on to the next sub-question. This error propagation will directly affect the retrieval. However, in a few cases, incorrect retrieval may lead to accidental correct reasoning, such as "William Keighley" and "Marion Gering" having the same place of birth.

As shown in Table 14, during the reasoning process of comparison questions, incorrect intermediate steps may not affect the consistency of the comparison relationship, thus luckily resulting in the correct final answer. For example, "June 25, 1954" is later than "1 October 1899", and "1947" is also later than "1 October 1899".

Question: Where was the place of death of the director of film Easy To Love (1934 Film)? *(2-hop Bridge)*

Gold Intermediate Steps:
Step 1: Who was the director of the film Easy To Love (1934 Film)? **William Keighley**
Step 2: Where was the place of death of **William Keighley**? **New York City**
Gold Final Answer: **New York City**

Predicted Intermediate Steps:
Step 1: Who was the director of the film Easy To Love (1934 Film)? **Marion Gering** ✗
(Retrievals) ... Ready for Love is a 1934 American romantic comedy film directed by ***Marion Gering*** *...*
Step 2: Where was the place of death of **Marion Gering**? **New York City** ✓
(Retrievals) ... Marion Gering (June 9, 1901 in Rostov-on-Don April 19, 1977 in ***New York City****) was a ...*
Predicted Final Answer: **New York City** ✓

Table 13: An example of *fortuitous reasoning continuance* occurring in a 2-hop bridge question.

F.3 Latent Reasoning Suspension

We define *latent reasoning suspension* as: in multi-hop reasoning, even if there are incorrect intermediate reasoning steps in the reasoning process, multi-hop reasoning does not terminate. This phenomenon often occurs in bridge questions. We present two examples on Bridge and Bridge-Comparison types in Table 15 and 16.

As shown in Table 15 and 16, the incorrect bridge entity transfer causes the subsequent rea-

Question: Who was born later, Amin Ahmed or Nancy Ditz? *(2-hop Comparison)*
Gold Intermediate Steps: **Step 1:** When was Amin Ahmed born? **1 October 1899** **Step 2:** When was Nancy Ditz born? **June 25, 1954** **Gold Final Answer:** **Nancy Ditz**
Predicted Intermediate Steps: **Step 1:** When was Amin Ahmed born? **1 October 1899** ✓ *(Retrievals) ... Amin Ahmed NPk, MBE (; **1 October 1899** - 5 December 1991) was a jurist and chief justice of ...* **Step 2:** When was Nancy Ditz born? **1947** ✗ *(Retrievals) ... Nancy Turner Nancy Jean Turner (born **1947**) is a notable North American ethnobiologist ...* **Predicted Final Answer:** **Nancy Ditz** ✓

Table 14: An example of *fortuitous reasoning continuance* occurring in a 2-hop comparison question.

soning to deviate from the semantics of the original multi-hop question. However, the MHQA system fails to identify this issue, leading the subsequent reasoning to proceed in the wrong direction. In fact, the reasoning beneficial to the multi-hop question has already ceased. Therefore, this situation only occurs in question types with a "Bridge" structure, such as Bridge-Comparison.

Question: Which country the director of film Our Emden is from? *(2-hop Bridge)*
Gold Intermediate Steps: **Step 1:** Who is the director of the film Our Emden? **Louis Ralph** **Step 2:** Which country is **Louis Ralph** from? **Austria** **Gold Final Answer:** **Austria**
Predicted Intermediate Steps: **Step 1:** Who is the director of the film Our Emden? **Ken G. Hall** ✗ *(Retrievals) ... German film, Our Emden, with additional sequences shot in Australia by director **Ken G. Hall**. ...* **Step 2:** Which country is **Ken G. Hall** from? **Australia** ✗ *(Retrievals) ... known as Ken G. Hall, was an **Australian** film producer and director, considered one of the most ...* **Predicted Final Answer:** **Australia** ✗

Table 15: An example of *latent reasoning suspension* occurring in a 2-hop bridge question.

F.4 Data Contamination

We define *contaminated data* as: In the case of only using the internal knowledge (i.e., the LLM directly generates answers through auto-regression) of the LLM, if the final answer obtained by directly answering a multi-hop question without decomposition is correct, but the intermediate steps of answering through decomposition are incorrect, then this question is contaminated. Table 17 shows how a contaminated data sample is identified.

Question: Which film has the director who died earlier, The Undercover Woman or Way of a Gaucho? *(4-hop Bridge-Comparison)*
Gold Intermediate Steps: **Step 1:** Who was the director of the film The Undercover Woman? **Thomas Carr** **Step 2:** Who was the director of the film Way of a Gaucho? **Jacques Tourneur** **Step 3:** When did **Thomas Carr** die? **April 23, 1997** **Step 4:** When did **Jacques Tourneur** die? **December 19, 1977** **Gold Final Answer:** **Way of a Gaucho**
Predicted Intermediate Steps: **Step 1:** Who was the director of the film The Undercover Woman? **Sergei Nolbandov** ✗ *(Retrievals) .. Undercover is a 1943 British war film ... The film was ... and directed by **Sergei Nolbandov** ...* **Step 2:** Who was the director of the film Way of a Gaucho? **Jacques Tourneur** ✓ *(Retrievals) ... Way of a Gaucho is a 1952 American western film directed by **Jacques Tourneur** and starring ...* **Step 3:** When did **Sergei Nolbandov** die? **1905** ✗ *(Retrievals) ... In **1905**, Grand Duke Sergei Alexandrovich was assassinated right next to the Kremlin Senate ...* **Step 4:** When did **Jacques Tourneur** die? **December 19, 1977** ✓ *(Retrievals) ... Jacques Tourneur (; November 12, 1904 - **December 19, 1977**) was a French film director known ...* **Predicted Final Answer:** **The Undercover Woman** ✗

Table 16: An example of *latent reasoning suspension* occurring in a 4-hop bridge-comparison question.

The gray part shown in Table 17 is our criterion for identifying contaminated data. Different from "wrong reasoning", wrong reasoning refers to the situation where, during the multi-hop reasoning process executed by the LLM, the final answer is wrong due to errors in the intermediate steps.

G Prompts

(i) Table 20 shows the prompts of the *planner* for PER-DP of HotpotQA, 2WikiMultihopQA, and MuSiQue. Table 21 shows the prompts of the *planner* for PER-QA, which include type classification and planning. (ii) Table 22 shows the prompts of the *executor* for PER-QA, which include Vanilla LLM and RAG. The prompt of PER-DP's *executor* is similar to that of PER-QA's *executor*, except that the retrieval is changed to the gold passage. (iii) Table 23 shows the prompts of the *reasoner* for PER-QA, which include few-shot prompting (direct answering) and zero-shot CoT prompting. For other prompts, see the code we released.

H Licenses

HotpotQA, 2WikiMultihopQA, `all-MiniLM-L6-v2` model and Qwen2.5-7B-Instruct model are re-

Question: Where was the director of film The Plaything Of Broadway born? *(2-hop Bridge)*
Ground Truth: **Step 1:** Who was the director of the film The Plaything Of Broadway? **John Francis Dillon** **Step 2:** Where was John Francis Dillon born? **New York** **Final Answer:** **New York City**
Vanilla LLM (without Question Decomposition): **Final Answer:** **New York City** ✓
PER-QA$_{vanilla}$ (with Question Decomposition): **Step 1:** Who was the director of the film The Plaything Of Broadway? **D.W. Griffith** ✗ **Step 2:** Where was D.W. Griffith born? **La Grange, Kentucky** ✗ **Final Answer:** **La Grange, Kentucky** ✗

Table 17: An example of a contaminated 2-hop bridge question. The gray part is our identification criterion.

leased under Apache License 2.0. MuSiQue is released under Creative Commons Attribution 4.0 International. Meta Llama-2, DeepSeek-R1-Distill-Llama-8B model and FlashRAG are released under MIT License. Meta Llama-3 is licensed under the Meta Llama 3 Community License. They are all for research purposes, and our experiments are consistent with their intended usage.

Question: Which country the director of film Littlerock (Film) is from? *(2-hop bridge question)*

An example of PER-QA$_{rag}$ (ours).
Stage1: ***Planner***
{"Q1": ["Who is the director of the film Littlerock?", "<A1>"], "Q2": ["Which country is <A1> from?", "<A2>"]}
Stage3: ***Executor***
Q1: Who is the director of the film Littlerock?
(Retrieve 5 passages)
A1: Mike Ott
Q2: Which country is Mike Ott from?
(Retrieve 5 passages)
A2: Philippines
Stage3: ***Reasoner***
Input: (evidence) Who is the director of the film Littlerock? Mike Ott; Which country is Mike Ott from? Philippines.
Output: Philippines ✓

An example of Self-Ask.
Follow up: Who is the director of the film Littlerock (Film)?
(Retrieve 5 passages)
Intermediate answer: The director of the film Littlerock is Mike Ott.
Follow up: Where is Mike Ott from?
(Retrieve 5 passages)
Intermediate answer: Mike Ott was born in Munich, Germany.
So the final answer is: Germany. ✗

An example of IRCoT QA
Stage1: ***IRCoT***
The director of the film Littlerock is Mike Ott.
(Retrieve 5 passages)
Mike Ott is an American film and music video director.
(Retrieve 5 passages)
So the answer is: America.
Stage2: ***Reader***
Input: (retrievals) 10 passages
Output: United States ✗

Table 18: Case study. Compared with Self-Ask and IRCoT QA, our PER-QArag reasoning structure is clearer. In addition, compared with IRCoT providing all retrievals as context to the LLM in the *Reader* stage, our method only provides a linearized evidence graph in the *Reasoner* stage, with fewer and more compact tokens, which can avoid the distraction of the LLM's attention.

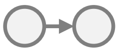
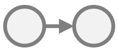
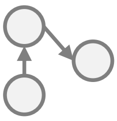
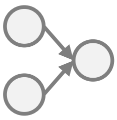
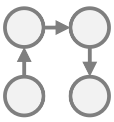
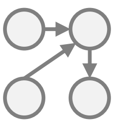
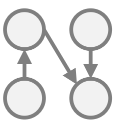
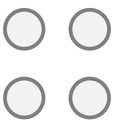
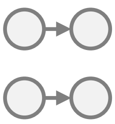
Type		Question and Plan (Question Decomposition)
Bri.^{2H}		**Who is the spouse of the owner of the British Virgin Islands?** 1. Who does the British Virgin Islands belong to? <A1> 2. Who is the spouse of <A1>? <A2>
Comp.^{2H}		**Were Ransom Riggs and John Berry from the same country?** 1. What country is Ransom Riggs from? <A1> 2. What country is John Berry from? <A2>
Infer.^{2H}		**Who is Elizabeth Of Pomerania's father-in-law?** 1. Who is the spouse of Elizabeth of Pomerania? <A1> 2. Who is the father of <A1>? <A2>
Bri.^{3H1}		**What is the name of the castle in the city where the performer of Never Too Loud was formed?** 1. Who is the performer of Never Too Loud? <A1> 2. What city was <A1> formed in? <A2> 3. What is the name of the castle in <A2>? <A3>
Bri.^{3H2}		**What was the language Auctor comes from during the era of the king who united the tribes in the 9th century later known as?** 1. In what language is Auctor? <A1> 2. What king united the tribes in the 9th century? <A2> 3. What was the <A1> of <A2>'s era later known as? <A3>
Bri.^{4H1}		**What date did the explorer reach the location of the headquarters of the only company larger than BMG's partner from 2005-2007?** 1. Who did BMG partner with from 2005 to 2007? <A1> 2. What is the only company larger than <A1>? <A2> 3. Where is the headquarters of <A2> located? <A3> 4. What date did the explorer reach <A3>? <A4>
Bri.^{4H2}		**When was the region immediately north of the region home to Israel and the location of Operation Earnest Will established?** 1. What region of the world is Israel located in? <A1> 2. Where did Operation Earnest Will take place? <A2> 3. What region lies immediately north of <A1> and <A2>? <A3> 4. When was <A3> established? <A4>
Bri.^{4H3}		**Who burned down the city where Dunn Dunn's artist died in the war during which income tax started?** 1. What label was responsible for Dunn Dunn? <A1> 2. Where did <A1> die? <A2> 3. When did income tax start in the United States? <A3> 4. Who burned down <A2> during <A3>? <A4>
Comp.^{4H}		**Who lived longer, Stefan Henze or Omar Rayo?** 1. When was Stefan Henze born? <A1> 2. When did Stefan Henze die? <A2> 3. When was Omar Rayo born? <A3> 4. When did Omar Rayo die? <A4>
B.C.^{4H}		**Which film has the director who was born later, The Fbi Story or Mayrig?** 1. Who was the director of the film The FBI Story? <A1> 2. Who was the director of the film Mayrig? <A2> 3. When was <A1> born? <A3> 4. When was <A2> born? <A4>

Table 19: Ten different types of multi-hop questions. Each question has a different graph structure. We use <A> to represent the answer to a certain sub-question and a placeholder in the next sub-question.

(2-hop bridge question) Prompt for the PER-DP's planner (2WikiMultihipQA and MuSiQue)
Your job is to decompose the question into two sub-questions. The sub-questions must be described using natural language, which is complete in one sentence. The sub-questions should start with W/H and be as concise as possible. In addition, some Tips will be provided to you, which outline the steps and logic of the decomposition process. Tips are not entirely natural language, but the output should be natural language. The output format should satisfy the following: {"Q1": ["A sub-question described in natural language. You can't start with 'When'.", "#1"], "Q2": ["A sub-question described in natural language and the placeholder #1 must be included.", "#2"]} **Examples:** Question: When was the institute that owned The Collegian founded? Tips: {"Q1": ["The Collegian » owned by", "#1"], "Q2": ["When was #1 founded?", "#2"]} Output: {"Q1": ["Which institute owned The Collegian?", "#1"], "Q2": ["When was #1 founded?", "#2"]} Question: What city is the person who broadened the doctrine of philosophy of language from? Tips: {"Q1": ["who broadened the doctrine of philosophy of language", "#1"], "Q2": ["What city is #1 from?", "#2"]} Output: {"Q1": ["Who broadened the doctrine of philosophy of language?", "#1"], "Q2": ["What city is #1 from?", "#2"]} Question: What language was used by Renana Jhabvala's mother? Tips: {"Q1": ["Who was Renana Jhabvala's mother?", "#1"], "Q2": ["#1 » languages spoken, written or signed", "#2"]} Output: {"Q1": ["Who was Renana Jhabvala's mother?", "#1"], "Q2": ["What language was used by #1?", "#2"]} **Question:** `{question}` **Tips:** `{tips}` **Output:**
(2-hop bridge question) Prompt for the PER-DP's planner (HotpotQA)
Given a bridge question. A bridge question involves two facts that are connected by an intermediate entity. Your job is to decompose the bridge question into two sub-questions. The sub-questions must be described using natural language, which is complete in one sentence. The sub-questions should start with W/H and be as concise as possible. In addition, some Tips will be provided to you, which outline the steps and logic of the decomposition process. Output is not allowed to leak any factual information from the Tips. The output format should satisfy the following: {"Q1": ["A sub-question described in natural language. You can't start with 'When'.", "#1"], "Q2": ["A sub-question described in natural language and the placeholder #1 must be included.", "#2"]} **Examples:** Question: When was the institute that owned The Collegian founded? Tips: 1. The Collegian (Houston Baptist University): The Collegian is the bi-weekly official student publication of Houston Baptist University in Houston, Texas. 2. Houston: Houston Baptist University, affiliated with the Baptist General Convention of Texas, offers bachelor's and graduate degrees. It was founded in 1960 and is located in the Sharpstown area in Southwest Houston. Output: {"Q1": ["Which institute owned The Collegian?", "#1"], "Q2": ["When was #1 founded?", "#2"]} Question: What city is the person who broadened the doctrine of philosophy of language from? Tips: 1. Philosophy of language: In the early 19th century, the Danish philosopher Soren Kierkegaard insisted that language ought to play a larger role in Western philosophy. 2. Soren Kierkegaard: Kierkegaard was born to an affluent family in Copenhagen. His mother, Ane Sorensdatter Lund Kierkegaard, had served as a maid in the household before marrying his father, Michael Pedersen Kierkegaard. Output: {"Q1": ["Who broadened the doctrine of philosophy of language?", "#1"], "Q2": ["What city is #1 from?", "#2"]} Question: What language was used by Renana Jhabvala's mother? Tips: 1. Renana Jhabvala: Renana Jhabvala was born in Delhi to the Booker Prize winning novelist and screen-writer, Ruth Prawer Jhabvala, and well-known architect Cyrus S. H. Jhabvala. Her grandparents were active in public life during the early to mid part of the twentieth century. 2. The Householder (novel): The Householder is a 1960 English language novel by Ruth Prawer Jhabvala. It is about a young man named Prem who has recently moved from the first stage of his life, a student, to the second stage of his life, a householder. Output: {"Q1": ["Who was Renana Jhabvala's mother?", "#1"], "Q2": ["What language was used by #1?", "#2"]} **Question:** `{question}` **Tips:** `{tips}` **Output:**

Table 20: Prompt for the *planner* of PER-DP. Teal represents **Known Constraints**. For HotpotQA, we set the structure constraint and passage constraint; for 2WikiMultihopQA and MuSiQue, we set the structure constraint and evidence constraint.

Prompt for the PER-QA's planner (type classification)

Given a question, determine what type the question belongs to. Types include:
1. Bridge: A bridge question involves two or more facts that are connected by an intermediate entity (usually an associative link). The bridge question requires finding the intermediary entity, then using it to answer the question.
2. Comparison: A comparison question involves comparing two or more independent facts. The comparison question requires analyzing and comparing the differences or similarities between different facts to draw a conclusion.

Examples:
Question: What language was used by Renana Jhabvala's mother?
Output: {"Response": "Bridge"}

Question: Which film came out first, The Love Route or Engal Aasan?
Output: {"Response": "Comparison"}

NOTE: Always respond with the JSON object.
Now it's your turn!

Question: `{question}`
Output:

(bridge-type) Prompt for the PER-QA's planner (planning)

Given a bridge question, split it into smaller, independent, and individual subqueries. A bridge question involves two or more facts that are connected by an intermediate entity (usually an associative link). The bridge question requires finding the intermediary entity, then using it to answer the question. For the subquery generation, input a tag "<A>" where the answer of the parent query should come to make the query complete. Specifically,
1. Subquery is NOT allowed to ask open-ended question. For example, for question "What language was used by Renana Jhabvala's mother?", it is NOT allowed to decompose and ask "Who is Renana Jhabvala?".
2. Each subquery is a simple fact question, not a question that requires reasoning. For example, "Who lives longer, <A1> or <A2>?" is NOT allowed.

Examples:
Question: What language was used by Renana Jhabvala's mother?
Output: {"Response": {"Q1": ["Who was Renana Jhabvala's mother?", "<A1>"], "Q2": ["What language was used by <A1>?", "<A2>"]}}

Question: Who is Sobe (Sister Of Saint Anne)'s child-in-law?
Output: {"Response": {"Q1": ["Who is the child of Sobe (Sister Of Saint Anne)?", "<A1>"], "Q2": ["Who is the spouse of <A1>?", "<A2>"]}}

Question: What followed the last person to live in Versailles in the country that became allies with America after the battle of Saratoga?
Output: {"Response": {"Q1": ["Who became allies with America after the Battle of Saratoga?", "<A1>"], "Q2": ["Who was the last person to live in Versailles?", "<A2>"], "Q3": ["What followed <A2> in <A1>?", "<A3>"]}}

NOTE: Always respond with the JSON object.
Now it's your turn!

Question: `{question}`
Output:

Table 21: Prompt for the *planner* of PER-QA

Prompt for the PER-QA's executor (Vanilla LLM)

You are a question answering system. Given a question, create an answer to the question.

Examples:
Question: In which state is Hertfordshire located?
Answer: {"Response": "East of England"}

Question: When was PolyGram Filmed Entertainment abolished?
Answer: {"Response": "1999"}

Question: Who plays michael myers in halloween by Rob Zombie?
Answer: {"Response": "Tyler Mane"}

NOTE: Always respond with the JSON object.
Now it's your turn!

Question: `{question}`
Answer:

Prompt for the PER-QA's executor (RAG) / Prompt for the PER-DP's executor (gold passage)

You are a question answering system. Use the retrievals while generating the answers and keep the answers grounded in the retrievals.

Examples:
Query: In which state is Hertfordshire located?
Retrievals: 1. Hertfordshire: Hertfordshire is the county immediately north of London and is part of the East of England region, a mainly statistical unit. A significant minority of the population across all districts are City of London commuters. To the east is Essex, to the west is Buckinghamshire and to the north are Bedfordshire and Cambridgeshire. ***(Other 4 retrievals. Omitted here. If it is PER-DP, we will not retrieve and will directly provide the gold passage.)***
Answer: {"Response": "East of England"}

Query: Who plays michael myers in halloween by Rob Zombie?
Retrievals: 1. Halloween (2007 film): Halloween is a 2007 American slasher film written, directed, and produced by Rob Zombie. The film stars Tyler Mane as the adult Michael Myers, Malcolm McDowell as Dr. Sam Loomis. Rob Zombieś ""reimagining"" follows the premise of John Carpenterś original, with Michael Myers stalking Laurie Strode and her friends on Halloween night. ***(Other 4 retrievals. Omitted here. If it is PER-DP, we will not retrieve and will directly provide the gold passage.)***
Answer: {"Response": "Tyler Mane"}

Query: Who wrote the theme song to Charlie Brown?
Retrievals: 1. Todd Dulaney: Todd Dulaney Todd Anthony Dulaney (born December 20, 1983) is an American gospel musician, and former baseball player. His music career started in 2011, with the release of the CD version, "Pulling Me Through". This would be his breakthrough released upon the "Billboard" Gospel Albums chart. He would release another album, "A Worshipper's Heart", in 2016 with EntertainmentOne Nashville, ***(Other 4 retrievals. Omitted here. If it is PER-DP, we will not retrieve and will directly provide the gold passage.)***
Answer: {"Response": "Vince Guaraldi"}

NOTE: Always respond with the JSON object.
Now it's your turn!

Question: `{question}`
Retrievals: `{retrievals}`
Answer:

Table 22: Prompt for the *executor* of PER-QA and PER-DP

(bridge-type) Prompt for the PER-QA's reasoner (few-shot prompting)
You are a question answering system. Use the evidence while generating the answer and keep the answer grounded in the evidence. Each piece of evidence is represented as "Question » Answer", where "»" means "the Answer to the Question is...". **Examples:** Question: When was the baseball team winning the world series in 2015 baseball created? Evidence: 1. Who won the world series in 2015 baseball? » Kansas City Royals 2. When was Kansas City Royals created? » 1969 Answer: {"Response": "1969"} Question: When did the French come to the region where Philipsburg is located? Evidence: 1. Where is Philipsburg located? » Sint Maarten 2. What terrain feature is located in the Sint Maarten region? » Great Bay and Great Salt Pond 3. When did the French come to Great Bay and Great Salt Pond? » 1625 Answer: {"Response": "1625"} Question: How many people who started the great migration of the Slavs live in the country the football tournament is held? Evidence: 1. Who started the Great Migration of the Slavs? » Germans 2. Where was the football tournament held? » Brazil 3. How many of Germans live in Brazil? » 5 million Answer: {"Response": "5 million"} NOTE: Always respond with the JSON object. Now it's your turn! **Question:** `{question}` **Evidence:** `{evidence}` **Answer:**
Prompt for the PER-QA's reasoner (zero-shot CoT prompting)
You are a question answering system. Use the evidence while generating the answer and keep the answer grounded in the evidence. The final step must start with "So the answer is:". **Question:** `{question}` **Evidence:** `{evidence}` **Answer:** Let's think step-by-step.

Table 23: Prompt for the *reasoner* of PER-QA