Graph Elicitation for Guiding Multi-Step Reasoning in Large Language Models

Anonymous ACL submission

Abstract

Chain-of-Thought (CoT) prompting along with sub-question generation and answering has enhanced multi-step reasoning capabilities of Large Language Models (LLMs). However, prompting the LLMs to directly generate sub-questions is suboptimal since they sometimes generate redundant or irrelevant questions. To deal with them, we propose a GE-Reasoning method, which directs LLMs to generate proper sub-questions and corresponding answers. Concretely, given an input question, we first prompt the LLM to generate knowledge triplets, forming a graph representation of the question. Unlike conventional knowledge triplets, our approach allows variables as head or tail entities, effectively representing a question as knowledge triplets. Second, for each triplet, the LLM generates a corresponding sub-question and answer along with using knowledge retrieval. If the prediction confidence exceeds a threshold, the sub-question and prediction are incorporated into the prompt for subsequent processing. This approach encourages that sub-questions are grounded in the extracted knowledge triplets, reducing redundancy and irrelevance. Our experiments demonstrate that our approach outperforms previous CoT prompting methods and their variants on multi-hop question answering benchmark datasets.

1 Introduction

Large Language Models (LLMs) (Brown et al., 2020; Ouyang et al., 2022; Chowdhery et al., 2022; Zhang et al., 2022; Touvron et al., 2023a,b; Meta, 2024) have shown remarkable performance on various natural language processing tasks even without fine-tuning for the target tasks. Specifically, Chain-of-Thought (CoT) prompting approach (Wei et al., 2022; Kojima et al., 2022) has improved the reasoning capability of the LLMs by generating intermediate rationales before making the final answer. Although CoT prompting and the variants have shown better performance on various reasoning tasks (Wei et al., 2022; Wang et al., 2023; Jung et al., 2022; Liu et al., 2022), they have difficulty in answering complex multi-hop questions (Press et al., 2023) with two problems: lack of knowledge and properly decomposing the question.

To deal with them, one approach is generating relevant sub-questions that are easier to answer than the original question and answering them. Using them as the context information has been shown effective to improve the reasoning performance (Patel et al., 2022; Radhakrishnan et al., 2023; Lyu et al., 2023; Kim et al., 2023; Ling et al., 2023; Qi et al., 2023). However, previous methods sometimes generate irrelevant, redundant, or insufficient sub-questions since they mostly rely on in-context learning (ICL) (Brown et al., 2020) with raw text exemplars to decompose the original question to the sub-questions without concrete guidances.

To address the issues of the sub-question generation, we propose a graph-guide prompting method, which leads LLMs to generate proper sub-questions and the answers by using graphs elicited from the question and the contexts. Our method first construct a question graph by leveraging LLM prompting with in-context learning to extract knowledge triplets from the question. Differently from conventional knowledge triplets, our approach let each triplet include variables as the head or the tail entities. This relaxed triplet representation facilitates represent question sentences as triplets. Second, for each triplet, the LLM generates a corresponding sub-question grounded to the triplet. Then, the LLM answers it along with intermediate rationales and external knowledge retrieval. Those sub-question and the answer pairs with low answer confidences are filtered out by confidence thresholding. We repeat the intermediate sub-question/rationale/answer generation step, and the filtering step until reaching the final answer to the original question or the maximum number of reasoning steps.

Using knowledge triplets in the sub-question generation has three benefits. First, since each sub-question is grounded by a knowledge triplet extracted from the question, the sub-questions are highly likely to be relevant to the original question. Second, as each sub-question is generated from a distinct triplet, the sub-questions are not redundant. Third, we can better track how each sub-question is grounded and composed. Additionally, we allow representing triplet entities as variables, which acts as the entity placeholders and facilitates representing question sentences as the triplets. Allowing variables in the knowledge triplets resembles first-order logic (FOL) (Barker-Plummer et al., 2011) representations, which are useful for specific tasks such as claim verification (Wang and Shu, 2023), but we do not require strict formal representations or external theorem provers for the reasoning process (Wang and Shu, 2023; Olausson et al., 2023; Pan et al., 2023a).

For each generated sub-question, the LLM generates the answer. We filter out uninformative sub-question/answer pairs if the answer confidence is below a threshold, similarly to (Jiang et al., 2023b). In the open-book settings, where we can retrieve external knowledge, we allow retrieving relevant paragraphs by using the sub-question as the query so that the answer can be better generated leveraging the retrieved information.

Some previous works also leverage extracted entities or triplets. Sun et al. (2023a); Jiang et al. (2023a) uses extracted entities for traversing given external knowledge graphs in knowledge-base question answering. Fu et al. (2021) uses entities and relations for sub-question generation, and it is evaluated on 2WikiMultiHopQA (Ho et al., 2020), but their approach is less flexible since it requires fine-tuning of multiple different components, the question type should be separately estimated, and the final answer is limited as an aggregation of the sub-questions' answers. Recent works (Li and Du, 2023; Liu et al., 2024) extract entities and their relations from the knowledge documents and augment them to the input prompt for reasoning. Different from them, we elicit a graph with variables from the input question and leverage it to decompose the complex question into multiple simple and relevant sub-questions.

We evaluate the effectiveness of our proposed methods on three multi-hop reasoning benchmark datasets: 2WikiMultihopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023). Our experiments are conducted with Llama-3 (Meta, 2024), which is a widely used open source LLM, with two model sizes (8B and 70B), and GPT-3.5 Turbo (Ouyang et al., 2022), which is a popular proprietary LLM. From the experiments, our method shows the best performance compared to the other prompting methods on top of the LLMs in both the closed-book (no retrieval) and the open-book (knowledge retrieval) settings.

Our main contributions are as follows:

- We propose a GE-Reasoning method that elicits knowledge triplets of the questions and utilize them for generating relevant and distinctive sub-questions.
- We propose an in-context learning method for extracting knowledge triplets with variable entities, which allows suitable triplet representations of the input questions.
- We present retrieval augmented generation with structural knowledge refinement that filters out irrelevant knowledge information.
- Our extensive experiments demonstrate that our proposed approach outperform the baselines on three multi-hop question answering benchmark datasets: 2WikiMultihopQA, MuSiQue, and Bamboogle.

2 Related Works

2.1 Prompts for Multi-Step Reasoning

Chain-of-thoughts prompting (Wei et al., 2022; Chowdhery et al., 2022; Kojima et al., 2022) has been successfully applied to various reasoning tasks by providing the reasoning steps in the demonstrations. The other approach for multi-step reasoning (Jung et al., 2022; Press et al., 2023; Gao et al., 2023; Creswell et al., 2023; Parisi et al., 2022; Schick et al., 2023) is applying symbolic functions to the prompting. Further, least-to-most prompting (Zhou et al., 2023) proposes a multi-stage prompting approach where one prompt is designed for generating sub-questions, and the other prompt is used for answering the sub-questions. In addition, some works (Hu et al., 2023; Pan et al., 2023b; Lyu et al., 2023) explore code-based approaches with external compilers to execute the code. Different from these works, our prompting method provides explicit guidance with elicited graphs to the LLMs to reason for the complex multi-hop questions. Additionally, we iteratively generate the intermediate rationales and then verify them to reach the correct answer from the question without external programs.

2.2 Prompting with Knowledge Retrieval

Prompting has also been widely applied to open-book question answering tasks requiring external knowledge information. Lazaridou et al. (2022); Sun et al. (2023b); Yu et al. (2023) use prompting methods for single retrieval for each question, which is suboptimal for knowledge intensive multi-hop questions. Self-Ask (Press et al., 2023) is proposed to improve the reasoning capability of LLMs by decomposing the question into sub-questions and simply answers the sub-questions using Google Search API. However, this approach is not based on multi-step reasoning, which is still with limited capabilities. ReAct (Yao et al., 2023) prompting generates a sequence of reasoning steps and action steps, but its performance highly depends on the scale of language models and it requires fine-tuning to outperform conventional chain-of-thought prompting methods on multi-hop question answering in the open domain setting. IR-CoT (Trivedi et al., 2023) uses knowledge retrieval given the intermediate thought as a query. Therefore, it sometimes replaces an accurate rationale with an incorrect rationale influenced by the noisy knowledge. Compared to these related works, our method shows the effectiveness for diverse sizes of LLMs on multi-hop question answering tasks in the open-domain settings.

3 Graph Elicitation for Guiding Multi-Step Reasoning

The goal of our **G**raph **E**licitation for guiding multi-step **Reasoning** (**GE-Reasoning**) is to solve complex multi-hop question-answering by decomposing the complex question into multiple sub-questions and generating the sub-answers with guidance based on elicited graphs. The overview of our GE-Reasoning is depicted in Figure 1. Concretely, we start with constructing a question graph by eliciting a graph from the question (**Step1. Question graph construction (Sec. 3.1)**). Then, we iterate the following steps. First, we generate a sub-question $\boldsymbol{q}^{(j)}$ based on one of the extracted triplets to obtain the information required to answer the input question $\boldsymbol{x}$ by referring to the question graph (**Step2. Sub-question generation (Sec. 3.2)**). After generating the sub-question, we predict its answer $\boldsymbol{r}^{(j)}$ by prompting LLMs (**Step3. Sub-answer generation (Sec. 3.3)**) along with **Step4. retrieval augmented generation (Sec. 3.4)** if needed. Next, we evaluate the confidence of the subanswer and use it to fill the variable entity (**Step5. Filtering and Variable assignment (Sec. 3.5)**). If there exist no remaining question triplets with variables or the maximum number of the repetitions is reached, we stop the iterative generation and predicts the final answer of the input question. The pseudocode of GE-Reasoning is in Algorithm 1.

3.1 Question Graph Construction.

We construct the question graph $\mathcal{G}_q = (\mathcal{V}_q, \mathcal{R}_q, \mathcal{T}_q)$ by extracting a set of triplets from the question q, where $\mathcal{V}_q$ and $\mathcal{R}_q$ are sets of nodes and relations, respectively. $\mathcal{T}_q$ denotes a set of triplets, $\boldsymbol{t} = (\boldsymbol{v}_h, \boldsymbol{r}, \boldsymbol{v}_t)$, where $\boldsymbol{v}_h, \boldsymbol{v}_t$ are head and tail nodes, respectively, and $\boldsymbol{r}$ is the relation between the two nodes. This question graph is a *heterogeneous* graph that consists of various types (relations) of edges connecting entities, which can be represented as (head, relation, tail) triplets. One way to construct the question graph is using relation extraction models (Fu et al., 2019, 2021; Melnyk et al., 2022). However, they require additional training step and their graph constructions do not generalize well beyond question sentences and diverse datasets.

To address these limitations, we introduce an in-context learning method to harness the power of LLMs. Given a question sentence $\boldsymbol{x}$, we prompt the language model to extract triplets from the sentence as follows:

$$\mathcal{G}_{\boldsymbol{x}} = \text{LM}\left(\mathcal{E}_G, \boldsymbol{x}\right), \tag{1}$$

where $\mathcal{G}_{\boldsymbol{x}}$ the question graph represented by a set of triplets and $\mathcal{E}_G = \{(\boldsymbol{x}_i, \mathcal{G}_i)\}_{i=1}^{|\mathcal{E}_G|}$ is a set of exemplars consisting of pairs of input $\boldsymbol{x}_i$ and the corresponding graph $\mathcal{G}_i$. The input prompt for the question graph construction is in Table 8. For example, given the question “Who is the spouse of the director of film The Golden Calf (1930 Film)", we extract the token sequence of triplets $\mathcal{G}_{\boldsymbol{x}} = \{$ `(The Golden Calf; director; name: #1)`, `(name: #1; spouse; name: #2)` $\}$ by Eq. (1), where `name: #1`, `name: #2` are variables with their type denoting the entities required to be answered via sub-questions. The type annotation adds

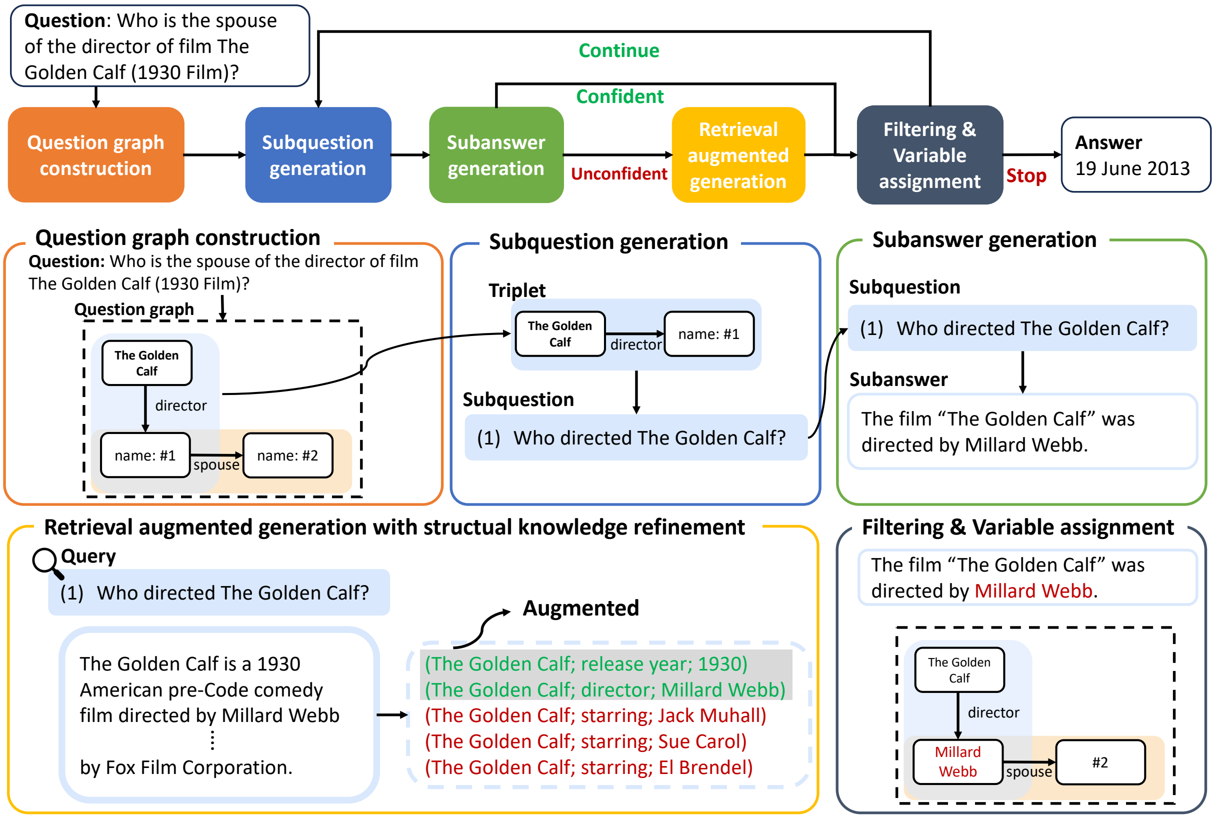

Question: Who is the spouse of the director of film The Golden Calf (1930 Film)?
Question graph construction
Subquestion generation
Subanswer generation
Retrieval augmented generation
Filtering & Variable assignment
Continue
Confident
Unconfident
Stop
Answer
19 June 2013
Question graph construction
Question: Who is the spouse of the director of film The Golden Calf (1930 Film)?
Question graph
The Golden Calf
director
name: #1
spouse
name: #2
Subquestion generation
Triplet
The Golden Calf
director
name: #1
Subquestion
(1) Who directed The Golden Calf?
Subanswer generation
Subquestion
(1) Who directed The Golden Calf?
Subanswer
The film "The Golden Calf" was directed by Millard Webb.
Retrieval augmented generation with structual knowledge refinement
Query
(1) Who directed The Golden Calf?
The Golden Calf is a 1930 American pre-Code comedy film directed by Millard Webb
⋮
by Fox Film Corporation.
Augmented
(The Golden Calf; release year; 1930)
(The Golden Calf; director; Millard Webb)
(The Golden Calf; starring; Jack Muhall)
(The Golden Calf; starring; Sue Carol)
(The Golden Calf; starring; El Brendel)
Filtering & Variable assignment
The film "The Golden Calf" was directed by Millard Webb.
The Golden Calf
director
Millard Webb
spouse
#2

Figure 1: The overview of our graph-guided prompting method for the complex reasoning task. We first construct the question graph by extracting the triplets from the question with in-context learning. Then, we sequentially repeat the following steps: (1) sub-question generation step (Sec. 3.2) that generates an intermediate sub-question based on a triplet with one variable (`(The Golden Calf; director; name: #1)`) of the question graph. (2) sub-answer generation step (Sec. 3.3) that generates the sub-answer by answering the previously generated sub-question. (3) retrieval augmented generation step (Sec. 3.4) that retrieves the knowledge given the sub-question as a query and then extracts triplets followed by filtering them to augment informative ones, based on the confidence thresholding. (4) variable assignment step (Sec. 3.5) that fills the variable based on the sub-answer. If there are no remaining question triplets with single variables and the repetition limit is reached, we stop the iteration. After the end of the iterations, the final answer to the original question is generated.

the details of variables, which helps the model to comprehend them. Once all the knowledge triplets are generated, we filter out invalid triplets such as those without any variable entities or with invalid formats.

3.2 Sub-Question Generation

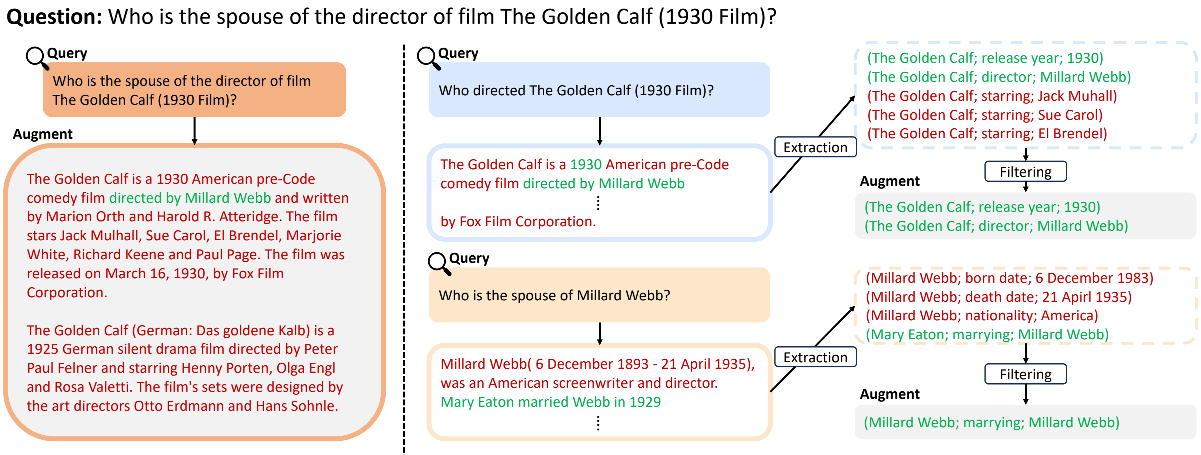
We generate sub-questions $\boldsymbol{q}^{(j)}$ with concrete guidance by the generated graph $\mathcal{G}_{\boldsymbol{x}}$. While the previous sub-question generation works such as Radhakrishnan et al. (2023); Yoran et al. (2023) generate subquestions only dependent on in-context demonstrations, we explicitly guide the model to generate subquestions with the graph structure $\mathcal{G}_{\boldsymbol{x}}$ (*i.e.*, knowledge triplets). Specifically, we first sample a triplet with one variable from the graph $\mathcal{G}_{\boldsymbol{x}}$ and construct a candidate set $\mathcal{C}_{\boldsymbol{x}} \subseteq \mathcal{G}_{\boldsymbol{x}}$. Then, the LLM generates a sub-question $\boldsymbol{q}^{(j)}$ based on the question triplet $\boldsymbol{t}^{(j)} \in \mathcal{C}_{\boldsymbol{x}}$, which can be formulated as:

$$\boldsymbol{q}^{(j)} = \mathrm{LM}\left(\mathcal{E}_{t\rightarrow q}, \boldsymbol{t}^{(j)}\right), \qquad (2)$$

where $\mathcal{E}_{t\rightarrow q} = \{(\boldsymbol{t}_i, \boldsymbol{q}_i)\}_{i=1}^{|\mathcal{E}_{t\rightarrow q}|}$ is a set of exemplars consisting of pairs of triplet $\boldsymbol{t}_i$ and the corresponding sub-question $\boldsymbol{q}_i$. The triplets with two variables are not used for sub-question generation, but they can be sub-question generatable if one of the variables are assigned later as described in Section 3.5. The graph-based guidance facilitates generating relevant and distinct sub-questions. Our experimental result in Section 4.5 shows that the proposed approach effectively suppresses generating irrelevant sub-questions.

3.3 Sub-Answer Generation

Given each sub-question $\boldsymbol{q}^{(j)}$ along with previously generated sub-questions and sub-answers, the LLM


Question: Who is the spouse of the director of film The Golden Calf (1930 Film)?
Query
Who is the spouse of the director of film The Golden Calf (1930 Film)?
Augment
The Golden Calf is a 1930 American pre-Code comedy film directed by Millard Webb and written by Marion Orth and Harold R. Atteridge. The film stars Jack Mulhall, Sue Carol, El Brendel, Marjorie White, Richard Keene and Paul Page. The film was released on March 16, 1930, by Fox Film Corporation.
The Golden Calf (German: Das goldene Kalb) is a 1925 German silent drama film directed by Peter Paul Felner and starring Henny Porten, Olga Engl and Rosa Valetti. The film's sets were designed by the art directors Otto Erdmann and Hans Sohnle.
Query
Who directed The Golden Calf (1930 Film)?
The Golden Calf is a 1930 American pre-Code comedy film directed by Millard Webb
⋮
by Fox Film Corporation.
Extraction
(The Golden Calf; release year; 1930)
(The Golden Calf; director; Millard Webb)
(The Golden Calf; starring; Jack Muhall)
(The Golden Calf; starring; Sue Carol)
(The Golden Calf; starring; El Brendel)
Filtering
Augment
(The Golden Calf; release year; 1930)
(The Golden Calf; director; Millard Webb)
Query
Who is the spouse of Millard Webb?
Millard Webb(6 December 1893 - 21 April 1935), was an American screenwriter and director.
Mary Eaton married Webb in 1929
⋮
Extraction
(Millard Webb; born date; 6 December 1983)
(Millard Webb; death date; 21 Apirl 1935)
(Millard Webb; nationality; America)
(Mary Eaton; marrying; Millard Webb)
Filtering
Augment
(Millard Webb; marrying; Millard Webb)

(a) Conventional Retrieval augmented generation

(b) Retrieval augmented generation with structural knowledge refinement

Figure 2: A comparison of conventional and our retrieval augmented generation approaches. In the conventional approaches, the input query is compound and complex to limit retrieving the full knowledge sufficient to reason over multi-hop question. It also augments all the information, including irrelevant ones, to the input prompt without filtering them. Our retrieval augmented generation approach uses a sub-question of each step as an input query for the retrieval system. After the retrieval, knowledge triplets are extracted from the knowledge paragraphs, and then relevant triplets are augmented to the input prompt after the filtering.

generates the corresponding answer $\boldsymbol{r}^{(j)}$:

$$\boldsymbol{r}^{(j)} = \mathrm{LM}\left(\mathcal{E}, \boldsymbol{x}, \boldsymbol{q}^{(1)}, \boldsymbol{r}^{(1)}, \ldots \boldsymbol{q}^{(j)}\right). \quad (3)$$

We use stochastic decoding with temperature 0.6 for the diverse answers during the iteration process. Since each sub-question $\boldsymbol{q}^{(i)}$ is simpler and shorter than the original question, it is easier for the LLM to predict the correct answer.

However, the LLM may still provide uncertain sub-answers due to the lack of knowledge. Motivated by Jiang et al. (2023b), we maintain the generated sub-answer if the confidence is above a threshold. Otherwise, we use retrieval augmented generation, where knowledge retrieval is leveraged for the sub-answer generation, as described in the next section.

3.4 Retrieval Augmented Generation with Structural Knowledge Refinement

A common approach for the retrieval augmented generation is single-level retrieval, which directly uses the input question $\boldsymbol{x}$ as the query for the retrieval system and then uses both the input and the retrieved knowledge documents $\mathcal{D}_{\boldsymbol{x}} = \mathrm{Ret}\left(\boldsymbol{x}\right)$ to generate the answer (Chen et al., 2017; Guu et al., 2020; Lewis et al., 2020). However, if the question is compound or requires multi-step reasoning, where important questions emerge amidst the reasoning process, a single retrieval might return irrelevant content or miss crucial information.

To deal with these issues, our retrieval method uses multi-level retrieval with sub-questions and refines the retrieved knowledge. Existing multi-level retrieval approaches such as IRCoT (Trivedi et al., 2023) use a sentence of the previous reasoning step as a query for the retrieval system. However, the previous sentence may be unrelated to the current reasoning step, which results in the retrieving irrelevant content. Thus, we use the generated sub-question $\boldsymbol{q}^{(j)}$ (*e.g.*, 'Who directed The Golden Calf (1930 Film)?') as the query instead of the complex input question or the sentence of the previous reasoning step. The sub-question reflects the information needed in the current reasoning step, facilitating the retrieval of informative knowledge.

Since we retrieve paragraphs, those retrieved contents may contain both relevant and irrelevant information to the sub-question. Therefore, we extract only relevant information from the paragraphs to help answering the sub-questions. The overall procedure is depicted in Figure 2. Specifically, we extract all possible knowledge triplets from each retrieved raw paragraph using the prompt in Table 11. Then, we filter out irrelevant triplets using the prompt in Table 12. Finally, as the outcome of the retrieval, we append the remaining triplets to the input prompt instead of the raw knowledge paragraphs. Some recent works (Li and Du, 2023; Liu et al., 2024) also extract knowledge triplets from the additional context, but they do not consider the relevance between the extracted triplets and the input question, and use all the triplets including noisy ones. Different from them, we use only informative triplets as additional contexts to the reasoning.

3.5 Filtering and Variable Assignment

The filtering step evaluates the confidence of the sub-answer. If the LLM generates low-confident answers even when retrieval augmentation is used, we filter out the current sub-question and the corresponding sub-answer. Note that even if the current sub-question and the answer are filtered out due to low confidence, the stochastic sub-answer generation may produce high confident answers in subsequent iterations.

If the answer has a high-confidence, we assign the answer to the variable of the triplet used for generating the sub-question. If the same variable exists in other triplets, we also update them accordingly. For example, in Figure 1, 'Millard Webb' is assigned to the variable entity `name: #1` after getting the answer of the sub-question 'Who directed The Golden Calf?'. Since `name: #1` also exists in `(name: #1; spouse; name: #2)`, the triplet is changed to `(Millard Webb; spouse; name: #2)`, which becomes eligible for sub-question generation.

3.6 Iterative Generation

We stop generating sub-questions and their answers when no remaining question triplets with single variables exist, or the repetition limit[1] is reached. Once the iteration is over, we generate the answer for the original question with the following instruction at the end of the prompt: "So the answer is `answer`".

4 Experiments

4.1 Experimental Setup

We evaluate the effectiveness of our proposed methods on three multi-hop question answering benchmark datasets: **2WikiMultihopQA** (2WikiMHQA) (Ho et al., 2020), **MuSiQue** (Trivedi et al., 2022), and **Bamboogle** (Bamboo) (Press et al., 2023). For the open-domain setting, we use the set of paragraphs provided in 2WikiMHQA and MuSiQue to curate an external knowledge corpus following the other existing works (Trivedi et al., 2023). For Bamboogle dataset, we use a retrieval based on Google search following existing works (Yoran et al., 2023; Zhao et al., 2023). In addition, we follow Trivedi et al. (2023); Jiang et al. (2023b) to randomly subsample 500 questions out of the dev set for 2WikiMHQA and MuSiQue datasets. We use all 125 questions for Bamboo dataset. We provide 6 exemplars for the in-context learning to predict the answer on all the datasets. We evaluate the performance of the approaches with the answer-level exact match (EM) and token-level F1.

4.2 Models

We experiment with open source Llama-3 (8B and 70B) (Meta, 2024) and proprietary GPT 3.5 Turbo (Ouyang et al., 2022) as the base LLMs. For the knowledge retrieval, we employ BM25 (Robertson et al., 2009) implemented with Elasticsearch and use the top-2 retrieved documents following other RAG works (Jiang et al., 2023b).

4.3 Baselines

In the open-book setting, which leverages the external knowledge, we use the following baselines: **No retrieval**: predicting the answer using CoT prompting without any external knowledge. **One retrieval**: predicting the answer using CoT prompting with the context retrieved with the input question as the query. **Verify-and-Edit** (Zhao et al., 2023): generating the reasoning steps and then predicting the answer after editing inaccurate sentences with the retrieved knowledge. **FLARE** (Jiang et al., 2023b): actively retrieving the knowledge context based on confidence and predicting an answer with the retrieved context. **IRCoT** (Trivedi et al., 2023): interleaving the retrieval with sentences and answer generation with the retrieved context. **ERA-CoT** (Liu et al., 2024): capturing relationships between entities and adding the relationships to the input prompt for better reasoning. While Liu et al. (2024) uses ERA-CoT with gold knowledge as the additional contexts, we evaluate ERA-CoT with retrieved knowledge for fair comparisons with the other approaches.

To validate the effectiveness of our proposed prompting method without knowledge retrieval (*i.e.,* closed book setting), we use the following baselines: **Chain-of-Thoughts** (Wei et al., 2022): predicting the answer with the reasoning steps with ICL exemplars. **Zero-Plus-Few-Shot CoT** (Kojima et al., 2022): including "Let's think step by step." before the reasoning steps of CoT. **Self-Consistency** (Wang et al., 2023): sampling five

[1] 10 in our experiments.

	Llama3-8B					Llama3-70B					GPT-3.5 Turbo				
	2WikiMHQA		MusiQue		Bamboo	2WikiMHQA		MusiQue		Bamboo	2WikiMHQA		MusiQue		Bamboo
Methods	EM	F1	EM	F1	EM	EM	F1	EM	F1	EM	EM	F1	EM	F1	EM
No retrieval	31.6	38.94	11.4	21.11	45.4	48.0	56.41	21.4	32.18	64.8	37.0	45.28	15.2	25.49	56.6
One retrieval	35.0	44.66	16.2	24.99	52.8	53.8	65.75	25.2	35.92	67.2	40.6	50.80	16.4	25.75	57.0
Verify-and-Edit	36.4	43.00	18.8	27.54	52.6	55.8	66.17	30.0	41.41	66.8	42.8	52.54	16.6	25.91	58.0
FLARE	41.8	50.70	22.6	31.72	54.0	65.2	72.62	32.0	42.86	69.0	50.8	61.31	19.4	31.33	57.2
IRCoT	45.6	57.35	22.6	31.94	53.2	63.8	73.39	29.6	40.96	69.6	48.2	58.53	17.8	29.06	57.6
ERA-CoT	35.2	42.93	18.0	26.64	52.8	54.8	65.78	26.2	37.60	66.2	42.0	51.67	16.4	26.02	56.2
GE-Reasoning (*Ours*)	**53.0**	**60.99**	**23.8**	**33.06**	**55.8**	**66.2**	**77.54**	**33.0**	**43.95**	**72.0**	**52.8**	**64.03**	**23.2**	**32.83**	**59.0**

Table 1: Performance comparisons on multi-hop question answering datasets in the open-book setting.

		2WikiMHQA		MusiQue	
Ret. Query	Subq Gen.	EM	F1	EM	F1
Input question	w/o Subq Gen.	35.0	44.66	16.2	24.99
Input question	w/o G-Guidance	35.8	44.22	16.4	24.70
Input question	with G-Guidance	**37.6**	**47.61**	**17.0**	**26.13**
Subquestion	w/o G-Guidance	49.2	57.32	22.4	31.76
Subquestion	with G-Guidance	**53.0**	**60.99**	**23.8**	**33.06**

Table 2: Performance comparison based on the retrieval query types and subquestion generation methods on multi-hop question answering datasets with Llama3-8B. (Ret. Query: Retrieval query type, Subq. Gen.: Subquestion generation).

		2WikiMHQA		MusiQue	
Knowl. Refine.	Filter.	EM	F1	EM	F1
		52.0	58.68	22.8	31.63
✓		**53.4**	60.41	23.2	32.18
✓	✓	53.0	**60.99**	**23.8**	**33.06**

Table 3: Ablation studies of our GE-Reasoning on multi-hop question answering datasets with Llama3-8B. (Knowl. Refine.: Knowledge Refinement, Filter.: Filtering).

	2WikiMHQA		MusiQue	
Methods	EM	F1	EM	F1
Chain-of-Thoughts	31.6	38.94	11.4	21.11
Zero-Plus-Few-Shot CoT	34.2	40.09	11.4	20.57
Self-Consistency	33.8	40.87	11.6	21.21
Self-ask	33.4	39.47	13.0	21.64
GE-Reasoning (*ours*)	**36.4**	**42.75**	**15.4**	**24.91**

Table 4: Performance comparison on multi-hop question answering datasets using Llama-3-8B without knowledge retrieval.

	Graph quality	QA	
Methods	Acc.	EM	F1
w/o type	95.4	52.2	60.16
with type	**96.0**	**53.0**	**60.99**

Table 5: Evaluation of graph quality and question answering with and without using entity type when constructing a question graph on 2WikiMHQA dataset using Llama-3 8B.

reasoning paths with a decoding temperature of 0.7 and using majority voting to get the answer. **Self-ask** (Press et al., 2023): sequentially asking questions until reaching the final answer.

4.4 Experimental Results

We evaluate our proposed methods using Llama3-8B, Llama3-70B, and GPT-3.5 Turbo in Table 1. From the table, our GE-Reasoning shows the best performance compared to the other baseline prompting methods on all the datasets with various LLMs. This results indicate that our prompting methods are widely applicable to diverse LLMs with different sizes for multi-hop QA tasks.

4.5 Additional Experimental Results.

Importance of retrieval query and subquestion generation. To demonstrate the importance of the retrieval query type and the graph-guidance, we conduct additional experiments with different retrieval queries and subgraph generation approaches in Table 2. The table shows that using the subquestions as the retrieval query substantially outperforms using the input question. This demonstrates that the input question is insufficient to retrieve full knowledge to solve the multi-step reasoning problems. For the subgraph generation, (w/o Subq Gen.) denotes the standard CoT without subgraph generation, and (w/o G-Guidance) decomposes the question into sub-questions only with an in-context learning scheme with a few demonstrations. The table shows that the graph-guidance consistently helps the model reason over multi-hop questions by guiding the reasoning process on different retrieval query types. When using the sub-question as a retrieval query, the graph-guidance improves the prompting without graph-guidance with a margin of 3.67 F1 score on 2WikiMHQA dataset. It leads to that the graph guidance is more effective in decomposing the question into sub-questions compared to the question decomposition, prompting only with a few exemplars.

Contribution of structural knowledge refinement and filtering. We provide the ablation stud-

Question & Answer	w/o G-guidance	with G-guidance (Ours)
Question: Which film whose director is younger, Running For Grace or Irulum Oliyum? **Answer:** Running For Grace	(1) Who directed Irulum Oliyum? Irulum Oliyum was directed by Puttanna Kanagal. (2) Who directed Running For Grace? Running For Grace was directed by David L. Cunningham. (3) Who directed Who is Running? Who is Running was directed by Oxide Pang Chun. (4) Who is the youngest director among Oxide Pang Chun, Puttanna Kanagal, and David L. Cunningham? Oxide Pang Chun is the youngest director. So the answer is **Who is Running**.	(1) Who directed Irulum Oliyum? (Irulum Oliyum; director; name: #1) Irulum Oliyum was directed by Puttanna Kanagal. (2) Who directed Running For Grace? (Running For Grace; director; name: #2) Running For Grace was directed by David L. Cunningham. (3) Who is younger, Puttanna Kanagal or David L. Cunningham? David L. Cunningham is younger. So the answer is **Running For Grace.**
Question: Who is Ahaziah Of Israel's maternal grandfather? **Answer:** Ithobaal I	(1) Who is Ahaziah Of Israel's mother? Ahaziah Of Israel's mother is Jezebel. (2) Who is Jezebel's husband? Jezebel's husband is Ahab, King of Israel. (3) Who is Ahab's father? Ahab's father is Omri, King of Israel. (4) Who is Ahaziah Of Israel's maternal grandfather? Ahaziah Of Israel's maternal grandfather is Omri, King of Israel. So the answer is **Omri**.	(1) Who is Ahaziah Of Israel's mother? Ahaziah Of Israel's mother is Jezebel. (2) Who is Jezebel's father? Jezebel's father is Ithobaal I of Sidon. (3) Who is Ahaziah Of Israel's maternal grandfather? Ahaziah Of Israel's maternal grandfather is Ithobaal I of Sidon. So the answer is **Ithobaal I of Sidon**.

Table 6: Comparison on reasoning steps and answers generated by prompting without and with the graph guidance.

ies to explore the contribution of structural knowledge refinement and filtering in Table 3. The table shows that both structural knowledge refinement and filtering contribute to the performance improvement of our GE-Reasoning. Especially, the structural knowledge refinement achieves the significant performance improvement of 1.4 on EM metric in 2WikiMHQA dataset. This result indicates that the retrieved knowledge paragraphs contain irrelevant information in many cases and the refinement process can mitigate the problem.

Performance comparison without knowledge retrieval. We also compare the performance of our prompting approach with other prompting methods without using knowledge retrieval (*i.e.,* closed book setting.) Table 4 shows that our method achieves the best performance, indicating our propose method is effective in both the retrieval augmented setting and the pure LLM setting for multi-step reasoning.

Quality of the constructed question graph. To empirically prove that our question graph construction generates an accurate question graph and denoting type improves the quality of the question graph and sub-questions based on the question graph, we evaluate the quality of the question graph and question-answering with and without type on 2WikiMHQA dataset using Llama-3 8B in Table 5. We evaluate the quality of the question graph with ground-truth graph of 2WikiMHQA. The table shows that our question graph construction accurately generates a question graph with and without type even using the smallest LLM we tried. Also, it demonstrates using the type noticeably improves the reasoning capability of the prompting method with the 0.8 performance gain on EM score.

4.6 Qualitative Analysis

Here, we perform qualitative analysis by comparing the reasoning steps and answers generated by prompting without and with graph guidance in Table 6. (w/o G-Guidance) decomposes the question into sub-questions only with in-context learning without the graph guidance. Given the question “Which film whose director is younger, Running for Grace or Irulum Oliyum?", (w/o G-Guidance) generates irrelevant sub-question “Who directed Who is Running?" while the prompting with graph-guidance generates sub-questions and their answers relevant to solve the main question. Rather, (w/o G-Guidance) gives the answer “Who is Running", instead of “Running for Grace" or “Irulum Oliyum". This case shows that the question decomposition without the guidance is prone to generating the wrong sub-question and our graph-guidance addresses it and effectively helps LLMs reason on multi-hop questions.

5 Conclusion

We have proposed a GE-Reasoning method to explicitly guide the large language models to reach the correct answer. We repeat the sub-question generation, answer generation, and answer filtering steps until predicting the final answer. We use retrieval augmentation using intermediate sub-questions as queries to obtain the external knowledge triplets helpful for the intermediate reasoning processes. Our experimental results on three multi-hop question answering benchmark datasets demonstrate the effectiveness of our GE-Reasoning methods.

6 Limitations

Similar to other prompting methods, the performance of our GE-Reasoning method relies on the large language models and the quality of the demonstrations. In the open-book setting, the quality of retrieved knowledge is highly dependent on BM25 retriever. Therefore, using advanced retrieval methods help our model improve the performance.

7 Ethics Statement

Our GE-Reasoning prompting addresses the potential ethical issues of the large language models, such as the hallucination issue. Some remaining concerns are that it could suffer from the ethical issue of the large language models such as Llama-3 and GPT-3.5 since it depends on the large language models to reason on multi-hop questions.

References

David Barker-Plummer, Jon Barwise, and John Etchemendy. 2011. *Language, Proof, and Logic, 2nd edition*. Center for the Study of Language and Information.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *NeurIPS*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *ACL*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv:2204.02311*.

Antonia Creswell, Murray Shanahan, and Irina Higgins. 2023. Selection-inference: Exploiting large language models for interpretable logical reasoning. In *ICLR*.

Ruiliu Fu, Han Wang, Xuejun Zhang, Jun Zhou, and Yonghong Yan. 2021. Decomposing complex questions makes multi-hop qa easier and more interpretable. In *EMNLP Findings*.

Tsu-Jui Fu, Peng-Hsuan Li, and Wei-Yun Ma. 2019. Graphrel: Modeling text as relational graphs for joint entity and relation extraction. In *ACL*, pages 1409–1418.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. In *ICML*, pages 10764–10799. PMLR.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *ICML*, pages 3929–3938. PMLR.

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 *COLING*, pages 6609–6625.

Yi Hu, Haotong Yang, Zhouchen Lin, and Muhan Zhang. 2023. Code prompting: a neural symbolic method for complex reasoning in large language models. *arXiv preprint arXiv:2305.18507*.

Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Wayne Xin Zhao, and Ji-Rong Wen. 2023a. Structgpt: A general framework for large language model to reason over structured data. In *EMNLP*.

Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023b. Active retrieval augmented generation. In *EMNLP*.

Jaehun Jung, Lianhui Qin, Sean Welleck, Faeze Brahman, Chandra Bhagavatula, Ronan Le Bras, and Yejin Choi. 2022. Maieutic prompting: Logically consistent reasoning with recursive explanations. In *EMNLP*.

Jiho Kim, Yeonsu Kwon, Yohan Jo, and Edward Choi. 2023. Kg-gpt: A general framework for reasoning on knowledge graphs using large language models. In *EMNLP Findings*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *NeurIPS*, pages 22199–22213.

Angeliki Lazaridou, Elena Gribovskaya, Wojciech Stokowiec, and Nikolai Grigorev. 2022. Internet-augmented language models through few-shot prompting for open-domain question answering. *arXiv preprint arXiv:2203.05115*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *NeurIPS*, volume 33, pages 9459–9474.

Ruosen Li and Xinya Du. 2023. Leveraging structured information for explainable multi-hop question answering and reasoning. In *EMNLP Findings*.

Zhan Ling, Yunhao Fang, Xuanlin Li, Zhiao Huang, Mingu Lee, Roland Memisevic, and Hao Su. 2023. Deductive verification of chain-of-thought reasoning. In *NeurIPS*.

Jiacheng Liu, Skyler Hallinan, Ximing Lu, Pengfei He, Sean Welleck, Hannaneh Hajishirzi, and Yejin Choi. 2022. Rainier: Reinforced knowledge introspector for commonsense question answering. In *EMNLP*.

Yanming Liu, Xinyue Peng, Tianyu Du, Jianwei Yin, Weihao Liu, and Xuhong Zhang. 2024. Era-cot: Improving chain-of-thought through entity relationship analysis. In *ACL*.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. Faithful chain-of-thought reasoning. In *AACL*.

Igor Melnyk, Pierre Dognin, and Payel Das. 2022. Knowledge graph generation from text. In *EMNLP Findings*.

AI Meta. 2024. Introducing meta llama 3: The most capable openly available llm to date. *Meta AI*.

Theo X. Olausson, Alex Gu, Benjamin Lipkin, Cedegao E. Zhang, Armando Solar-Lezama, Joshua B. Tenenbaum, and Roger Levy. 2023. Linc: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. In *EMNLP*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. In *NeurIPS*, volume 35, pages 27730–27744.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. 2023a. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. In *EMNLP Findings*.

Liangming Pan, Xiaobao Wu, Xinyuan Lu, Anh Tuan Luu, William Yang Wang, Min-Yen Kan, and Preslav Nakov. 2023b. Fact-checking complex claims with program-guided reasoning. In *ACL*.

Aaron Parisi, Yao Zhao, and Noah Fiedel. 2022. Talm: Tool augmented language models. *arXiv preprint arXiv:2205.12255*.

Pruthvi Patel, Swaroop Mishra, Mihir Parmar, and Chitta Baral. 2022. Is a question decomposition unit all we need? In *EMNLP*.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *EMNLP Findings*.

Jingyuan Qi, Zhiyang Xu, Ying Shen, Minqian Liu, Di Jin, Qifan Wang, and Lifu Huang. 2023. The art of socratic questioning: Recursive thinking with large language models. In *EMNLP*.

Ansh Radhakrishnan, Karina Nguyen, Anna Chen, Carol Chen, Carson Denison, Danny Hernandez, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamilė Lukošiūtė, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Sam McCandlish, Sheer El Showk, Tamera Lanham, Tim Maxwell, Venkatesa Chandrasekaran, Zac Hatfield-Dodds, Jared Kaplan, Jan Brauner, Samuel R. Bowman, and Ethan Perez. 2023. Question decomposition improves the faithfulness of model-generated reasoning. *arXiv preprint 2307.11768*.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. In *NeurIPS*.

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

Zhiqing Sun, Xuezhi Wang, Yi Tay, Yiming Yang, and Denny Zhou. 2023b. Recitation-augmented language models. In *ICLR*.

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

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv:2307.09288*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *TACL*, 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 *ACL*, pages 10014–10037.

Haoran Wang and Kai Shu. 2023. Explainable claim verification via knowledge-grounded reasoning with large language models. In *EMNLP Findings*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *ICLR*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, volume 35, pages 24824–24837.

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

Ori Yoran, Tomer Wolfson, Ben Bogin, Uri Katz, Daniel Deutch, and Jonathan Berant. 2023. Answering questions by meta-reasoning over multiple chains of thought. In *EMNLP*.

Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. 2023. Generate rather than retrieve: Large language models are strong context generators. In *ICLR*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv:2205.01068*.

Ruochen Zhao, Xingxuan Li, Shafiq Joty, Chengwei Qin, and Lidong Bing. 2023. Verify-and-edit: A knowledge-enhanced chain-of-thought framework. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5823–5840, Toronto, Canada. Association for Computational Linguistics.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. 2023. Least-to-most prompting enables complex reasoning in large language models. *ICLR*.

A Datasets and Licenses

We use following three benchmark multi-hop question answering datasets to evaluate the performance of the baselines and our method. To the best of our knowledge, these datasets do not have any privacy issue.

- **2WikiMultihopQA**[2] (Ho et al., 2020) consists of 2-hop complex questions requiring the composition or comparison.
- **MuSiQue**[3] (Trivedi et al., 2022) is a more challenging dataset where the problems include 2 to 4 hop questions that can be decomposed into simpler questions.
- **Bamboogle**[4] (Trivedi et al., 2022) is a dataset consisting of 125 multi-hop questions where the supporting evidence is from Wikipedia.

B Implementaion Details

Due to the heavy computational costs, we perform experiments with a single run on 2WikiMHQA and Musique datasets and 4 runs on Bamboogle dataset. All the experiments are conducted using in-context learning (Brown et al., 2020) with 6 demonstrations to predict the answer on all the datasets. For the knowledge retrieval, we employ BM25 (Robertson et al., 2009) implemented with Elasticsearch and use top-2 retrieved documents following other RAG works (Jiang et al., 2023b).

C Additional Experiments

We conduct additional experiments to compare the efficiency of our GE-Reasoning with other baseline prompting methods in Table 7. The time is measured as seconds per input question. From the table, our GE-Reasoning shows the best performance on the 2WikiMHQA dataset with a comparable time compared to other retrieval-based methods. In particular, GE-Reasoning improves 10.29 F1 score and 5.0 seconds per question compared to FLARE.

Methods	EM	F1	Time
No retrieval	31.6	38.94	**2.8**
One retrieval	35.0	44.66	8.2
Verify-and-Edit	36.4	43.00	14.6
FLARE	41.8	50.70	21.9
IRCoT	45.6	57.35	12.8
ERA-CoT	35.2	42.93	11.2
GE-Reasoning (*Ours*)	**53.0**	**60.99**	16.9

Table 7: Performance and efficiency comparisons on 2WikiMHQA dataset using Llama-3 8B. The time is evaluated as seconds per question. **Bold** indicates the best performance.

D Input Prompts

Table 8 shows the prompts for question graph construction. For the subquestion and subanswer generation, we use prompts in Table 9 and Table 10, respectively. Table 11 and Table 12 are prompts for extracting triplets from the knowledge passages and filtering out irrelevant triplets in Section 3.4.

[2] Copyright (c) 2020 Xanh Ho, Licensed under Apache-2.0 license

[3] Copyright (c) Licensed under Apache-2.0 license

[4] Copyright (c) 2022 Ofir Press, Licensed under MIT license

Algorithm 1 Algorithm of GE-Reasoning

```
Input: $x$: input question, $k_1, k_2$: hyper-parameter
Output: Answer: answer of the input question
$\mathcal{G}_{\boldsymbol{x}} \leftarrow$ ConstructGraph $(\boldsymbol{x})$
Stop $\leftarrow$ False
$j \leftarrow 1$
$\mathcal{C}_{\boldsymbol{x}} \leftarrow$ FindTripletswithOneVariable $(\mathcal{G}_{\boldsymbol{x}})$
while Stop is False do
    $\boldsymbol{t}^{(j)} \leftarrow$ Sample $(\mathcal{C}_{\boldsymbol{x}})$
    $\boldsymbol{q}^{(j)} \leftarrow$ SubquestionGenerate $\left(\boldsymbol{t}^{(j)}\right)$
    $\boldsymbol{r}^{(j)} \leftarrow$ SubanswerGenerate $\left(\boldsymbol{q}^{(1)}, \ldots \boldsymbol{q}^{(j-1)}, \boldsymbol{r}^{(j-1)}, \boldsymbol{q}^{(j)}\right)$
    if Confidence($\boldsymbol{r}^{(j)}$) $< k_1$ then
        $\mathcal{D}_{\boldsymbol{q}^{(j)}} \leftarrow$ Ret $\left(\boldsymbol{q}^{(j)}\right)$
        $\tilde{\mathcal{D}}_{\boldsymbol{q}^{(j)}} \leftarrow$ KnowledgeRefine $\left(\mathcal{D}_{\boldsymbol{q}^{(j)}}, \boldsymbol{q}^{(j)}\right)$
        $\boldsymbol{r}^{(j)} \leftarrow$ SubanswerGenerate $\left(\tilde{\mathcal{D}}_{\boldsymbol{q}^{(j)}}, \boldsymbol{q}^{(j)}\right)$
    end if
    if Confidence($\boldsymbol{r}^{(j)}$) $< k_2$ then
        $\boldsymbol{q}^{(j)}, \boldsymbol{r}^{(j)} \leftarrow$ “”, “”
    else
        $\mathcal{G}_{\boldsymbol{x}} \leftarrow$ VariableAssignment $\left(\mathcal{G}_{\boldsymbol{x}}, \boldsymbol{r}^{(j)}, \boldsymbol{t}^{(j)}\right)$
    end if
    $j \leftarrow j + 1$
    $\mathcal{C}_{\boldsymbol{x}} \leftarrow$ FindTripletswithOneVariable $(\mathcal{G}_{\boldsymbol{x}})$
    if $j >= 10$ or $\mathcal{C}_{\boldsymbol{x}} = \emptyset$ then
        Stop $\leftarrow$ True
    end if
end while
Answer $\leftarrow$ FinalAnswer $\left(\boldsymbol{q}^{(1)}, \ldots \boldsymbol{q}^{(j-1)}, \boldsymbol{r}^{(j-1)}, \boldsymbol{q}^{(j)}, \boldsymbol{r}^{(j)}\right)$
return Answer
```

Given a sentence, and all entities within the sentence. Extract all relationships between entities which directly stated in the sentence. Every relationship stated as a triple: (E_A; Relation; E_B).
Sentence: When did the director of film Hypocrite (Film) die? Relation: (Hypocrite (Film); director; name: #1), (name: #1; death date; date: #2)
...

Given a sentence, and all entities within the sentence. Extract all relationships between entities which directly stated in the sentence. Every relationship stated as a triple: (E_A; Relation; E_B).
Sentence: {sentence}
Triplets:

Table 8: A question graph construction prompt for eliciting a graph from the question.

Given the triplet, generate a subquestion based on the triplet.
Triplet: (Hypocrite (Film); director; name: #1)
Subquestion: Who directed Hypocrite (Film)?

...

Given the triplet, generate a subquestion based on the triplet.
Triplet: {triplet}
Subquestion:

Table 9: A subquestion generation prompt.

Question: When did the director of film Hypocrite (Film) die?
To answer this question, we answer the following subquestions:
(1) Who directed Hypocrite (Film)?
The film Hypocrite was directed by Miguel Morayta.
(2) When did Miguel Morayta die?
Miguel Morayta died on 19 June 2013.
So the answer is 19 June 2013.

...

Question: {question}
To answer this question, we answer the following subquestions:
{subquestion}

Table 10: A subanswer generation prompt.

Extract triplets from the following paragraph:
Maheen Khan is a Pakistani fashion and costume designer, also an award winner fashion designer for fashion labels

...

Triplets:
(Maheen Khan; nationality; Pakistan)
(Maheen Khan; profession; fashion and costume designer)
(Maheen Khan; award winner; The Embroidery HouseMaheen)

...

Extract triplets from the following paragraph:
{paragraph}
Triplets:

Table 11: A prompt for extracting triplets from the knowledge passage.

Given knowledge triplets and a question, select triplets that are relevant to the question.
Triplets:
{triplets extracted from the knowledge passage} Question:
{question}
Filtered triplets:

Table 12: A prompt for filtering out irrelevant triplets.