TRACE the Evidence: Constructing Knowledge-Grounded Reasoning Chains for Retrieval-Augmented Generation

Jinyuan Fang
University of Glasgow
j.fang.2@research.gla.ac.uk

Zaiqiao Meng*
University of Glasgow
zaiqiao.meng@glasgow.ac.uk

Craig Macdonald
University of Glasgow
craig.macdonald@glasgow.ac.uk

Abstract

Retrieval-augmented generation (RAG) offers an effective approach for addressing question answering (QA) tasks. However, the imperfections of the retrievers in RAG models often result in the retrieval of irrelevant information, which could introduce noise and degrade the performance, especially when handling multi-hop questions that require multiple steps of reasoning. To enhance the multi-hop reasoning ability of RAG models, we propose **TRACE**[1]. TRACE constructs *knowledge-grounded reasoning chains*, which are a series of logically connected knowledge triples, to identify and integrate supporting evidence from the retrieved documents for answering questions. Specifically, TRACE employs a *KG Generator* to create a knowledge graph (KG) from the retrieved documents, and then uses a novel *Autoregressive Reasoning Chain Constructor* to build reasoning chains. Experimental results on three multi-hop QA datasets show that TRACE achieves an average performance improvement of up to 14.03% compared to using all the retrieved documents. Moreover, the results indicate that using reasoning chains as context, rather than the entire documents, is often sufficient to correctly answer questions.

1 Introduction

Retrieval-augmented generation (RAG) models have achieved remarkable performance on question answering (QA) task (Lewis et al., 2020; Izacard et al., 2023; Ram et al., 2023; Lin et al., 2024). These models employ a *retriever-reader* architecture (Karpukhin et al., 2020). The *retriever* is used to retrieve a set of documents relevant to the questions, and the *reader* generates answers based on these documents. Moreover, the reader is often instantiated with large language models (LLMs) due to their powerful in-context learning capabilities, leading to superior zero-shot performance. In this setting, the retrieved documents are prepended to the question, which is used as input to the LLMs to generate answers (Ram et al., 2023).

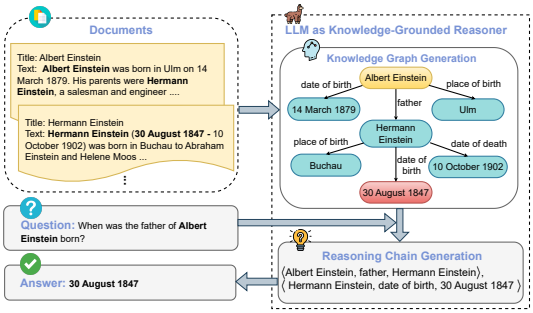

Figure 1: TRACE transforms documents into a KG and constructs reasoning chains to answer the question.

However, simply prepending all the documents returned by the retriever can result in suboptimal performance. This is because existing retrievers are not perfect and often include irrelevant documents in the retrieved set. These irrelevant documents introduce noises, which can mislead the reader and degrade performance (Shi et al., 2023a). This issue is particularly problematic when answering *multi-hop* questions, which involve multiple reasoning steps to obtain the correct answers. Previous study indicates that irrelevant documents can significantly impair the multi-hop reasoning ability of RAG models (Yoran et al., 2024).

Therefore, this work focuses on improving the multi-hop reasoning capability of RAG models by enhancing their ability to identify and integrate supporting evidence within documents. The supporting evidence refers to the information within documents that directly contributes to answering the questions (Ramesh et al., 2023). To this end, we propose **TRACE**, which cons**T**ructs knowledge-grounded **R**e**A**soning **C**hains to identify and integrate supporting **E**vidence from multiple documents. Figure 1 provides an illustration of TRACE.

*Corresponding Author.
[1]Code: https://github.com/jyfang6/trace

Findings of the Association for Computational Linguistics: EMNLP 2024, pages 8472–8494
November 12-16, 2024

Specifically, in order to identify supporting evidence, TRACE first transforms the retrieved documents into a knowledge graph (KG), i.e., a set of knowledge triples in the form of ⟨*head entity, relation, tail entity*⟩ that describe relationships between entities. This is achieved by using in-context learning to prompt an LLM instructed as a *KG Generator* to generate knowledge triples from the retrieved documents. The motivation for converting documents into a KG is that, compared with sentences or documents, which contain multiple pieces of information, knowledge triples offer a finer-grained and more concise way to express knowledge, where each triple only conveys a single piece of factual knowledge. Leveraging KG triples can reduce the impact of irrelevant data when identifying supporting evidence (Fang et al., 2024), leading to more accurate identification of relevant information. For example, the sentence "*Albert Einstein (14 March 1879-18 April 1955) was a German-born theoretical physicist.*" contains multiple pieces of information about Albert Einstein, including his birth and death days, nationality, and profession. One of the triples generated from this sentence could be ⟨*Albert Einstein, date of birth, 14 March 1879*⟩, which decouples the birthday information from the sentence. This finer granularity helps in minimising the inclusion of irrelevant information, making it easier to identify supporting evidence when answering questions related to Einstein's birthday.

Notably, the generation of the KG is independent of questions. TRACE next aims to identify and integrate supporting evidence from the KG to answer multi-hop questions. Specifically, TRACE employs an *Autoregressive Reasoning Chain Constructor* to construct reasoning chains from the KG. Each reasoning chain comprises several KG triples that logically connect pieces of supporting evidence to answer the questions. For example, for a multi-hop question "*When was the father of Albert Einstein born?*", "⟨*Albert Einstein, father, Hermann Einstein*⟩, ⟨*Hermann Einstein, date of birth, 3 July 1814*⟩" is a reasoning chain that provides the necessary information to answer the question. These reasoning chains facilitate the integration of dispersed pieces of supporting evidence, thereby enhancing the model's ability to generate correct answers.

Moreover, the reasoning chains are constructed in an autoregressive manner. At each step, TRACE uses in-context learning to prompt the constructor to select a triple from the KG based on both the question and the previously selected triples. The objective is to ensure that the selected triple forms a logically coherent reasoning chain with the previously selected triples. The autoregressive way of constructing reasoning chains is inspired by human reasoning, where each piece of information is considered in the context of what has already been understood. This step-by-step reasoning approach is particularly suitable for multi-hop questions, as it can trace the logical connections across multiple pieces of evidence, ensuring an accurate inference process. For example, in the previously mentioned multi-hop question, if TRACE has already identified one piece of supporting evidence: ⟨*Albert Einstein, father, Hermann Einstein*⟩, it can then focus on finding the next relevant piece of evidence, i.e., ⟨*Hermann Einstein, date of birth, 3 July 1814*⟩.

Consequently, compared to vanilla RAG models, TRACE creates a KG from the retrieved documents and constructs reasoning chains from the KG in an autoregressive manner to identify and integrate supporting evidence. Given the reasoning chains, the TRACE reader either directly uses them as context to generate the answer (TRACE-Triple), or use them to identify a subset of documents that are useful for answering the question (TRACE-Doc). We conduct experiments on three multi-hop QA datasets in a zero-shot setting, our results show that, compared to using all the retrieved documents, TRACE-Triple and TRACE-Doc achieve average improvements of 14.03% and 13.46% in terms of Exact Match (EM), respectively. Moreover, our results indicate that, in the RAG setting, constructing more condensed reasoning chains (i.e., KG triples) from the retrieved documents as context, rather than using the entire documents, is often sufficient to correctly answer questions.

Our contributions are summarised as follows: (1) We propose TRACE, which builds knowledge-grounded reasoning chains to enhance the multi-hop reasoning ability of RAG models; (2) We propose a novel autoregressive method to construct reasoning chains to identify and integrate supporting evidence; (3) Experimental results on three multi-hop QA datasets show that TRACE achieves average improvement of up to 14.03% in terms of EM compared to using all the retrieved documents.

2 Problem Formulation

This work focuses on tackling multi-hop questions. We denote a multi-hop question and its answer as q and a, respectively. Each question is associated

with a set of N documents: $\mathcal{D}_q = \{d_1, d_2, \ldots d_N\}$, which are obtained with a retriever model. Following previous work (Trivedi et al., 2023), the documents are often retrieved from Wikipedia, where each document comprises a title and a text. Given the question q and the document set $\mathcal{D}_q$, the goal is to correctly generate the answer a.

3 TRACE

This section begins by outlining the overall framework of TRACE in § 3.1. Next, we explain the details of each component in the following sections: KG generator in § 3.2, reasoning chain constructor in § 3.3 and finally, answer generation in § 3.4.

3.1 Overall Framework

Figure 2 provides an overview of TRACE. Given a multi-hop question q and a set of documents $\mathcal{D}_q$, TRACE follows these steps to generate the answer: **(1) KG Generation**: TRACE first leverages a *KG generator* to create a KG from $\mathcal{D}_q$, i.e., generating a set of knowledge triples in the form of ⟨*head entity*, *relation*, *tail entity*⟩; **(2) Reasoning Chain Construction**: Next, it uses an *autoregressive reasoning chain constructor* to construct reasoning chains (paths)[2] from the KG; **(3) Answer Generation**: Finally, TRACE generates the answer by either using the reasoning chains directly as context or leveraging the chains to further retrieve their original context documents (see § 3.4 for details).

Specifically, TRACE can be considered as:

$$p(a|q, \mathcal{D}_q) \sim p(a|q, z, \mathcal{D}_q) \cdot p(z|q, \mathcal{G}_q) \cdot p(\mathcal{G}_q|\mathcal{D}_q), \quad (1)$$

where $p(\mathcal{G}_q|\mathcal{D}_q)$ denotes the KG generator for creating the KG $\mathcal{G}_q$, $p(z|q, \mathcal{G}_q)$ represents the reasoning chain constructor for building the reasoning chain z, which consists of a series of logically connected KG triples, and $p(a|q, z, \mathcal{G}_q)$ denotes the reader that generates the answer. In the following, we introduce the details of each component.

3.2 KG Generator

To mitigate the impact of irrelevant data when identifying supporting evidence, TRACE employs a KG generator $p(\mathcal{G}_q|\mathcal{D}_q)$ to create a KG from $\mathcal{D}_q$. Following the recent practice of generating KGs with LLMs (Wei et al., 2023; Zhang and Soh, 2024), we use in-context learning (Wei et al., 2022) to prompt an LLM instructed as a KG generator to generate KG triples from the documents $\mathcal{D}_q$. A straightforward approach is to concatenate all the documents within $\mathcal{D}_q$ as inputs and prompt the LLM to generate KG triples. However, this approach may suffer from the "lost-in-the-middle" issue (Liu et al., 2024), where the LLM ignores information from documents placed within the long input. Therefore, TRACE independently generates KG triples for each document and constructs relationships across documents by common entities shared among these documents. This approach not only mitigates the lost-in-the-middle issue but also allows for offline precomputation of KG triples for all the documents.

The prompt used by the KG generator to generate KG triples from a document is detailed in Appendix B.1. In particular, since this work focuses on documents retrieved from Wikipedia[3], which consist of a title and a text, we consider the title as an entity and instruct the KG generator to jointly recognise the entities within the text and infer their relationships with the title entity. This approach leverages the natural relevance between the title and the text, as entities within the text are more likely to have meaningful relationships with the title. Our empirical findings show that this KG generation approach yields satisfactory performance.

3.3 Reasoning Chain Constructor

Given the generated KG $\mathcal{G}_q$, TRACE next employs a reasoning chain constructor $p(z|q, \mathcal{G}_q)$ to identify and integrate supporting evidence by constructing reasoning chains in an autoregressive manner:

$$p(z|q, \mathcal{G}_q) = \prod_{i=1}^{L} p(z_i|q, z_{<i}, \hat{\mathcal{G}}_i), \quad (2)$$

$$\hat{\mathcal{G}}_i = f(q, z_{<i}, \mathcal{G}_q), \ \forall i = 1, \ldots, L, \quad (3)$$

where L denotes the maximum length of reasoning chains, z_i denotes the i-th triple in the reasoning chains, $z_{<i}$ represents all the triples preceding the i-th triple, and $\hat{\mathcal{G}}_i$ denotes the candidate triples from which the i-th triple is chosen. Specifically, at each i-th step, the constructor first employs a *triple ranker* $f(q, z_{<i}, \mathcal{G}_q)$ to obtain a set of candidate triples $\hat{\mathcal{G}}_i$ from $\mathcal{G}_q$, and then uses a *triple selector* $p(z_i|q, z_{<i}, \hat{\mathcal{G}}_i)$ to select the i-th triple from the candidate set. Therefore, the reasoning chains are constructed by selecting triples one by one from

[2] We use the concept reasoning chain and reasoning path interchangeably, with reasoning path being a commonly used term in the KG reasoning domain (Zhang et al., 2022).

[3] For documents retrieved from other sources, one can design specific prompts to optimise the KG generation process.

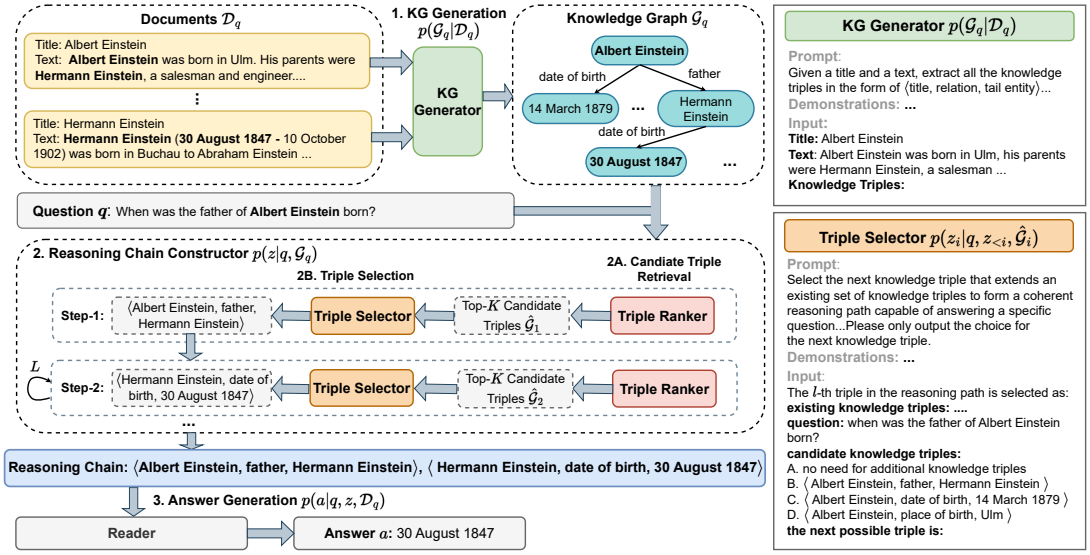

Figure 2: Overview of TRACE. Given a multi-hop question and the retrieved documents, TRACE first uses an LLM-based KG generator to create a KG based on the documents. It then employs an autoregressive reasoning chain constructor to build reasoning chains from the KG, which consists of a triple ranker for selecting candidate triples and an LLM-based triple selector for selecting a triple from the candidate set. The resulting reasoning chains are subsequently passed to a reader to generate the answer.

the KG. We next introduce the details of the triple ranker and the triple selector at each i-th step.

Triple Ranker. At each i-th step, triple ranker aims to select a subset of candidate triples from $\mathcal{G}_q$ that are relevant to q and $z_{<i}$. The triple ranker is implemented with a bi-encoder model $\text{Enc}(\cdot)$. Specifically, the triple ranker considers the concatenation of the question q and the previously selected triples $z_{<i}$ as the query. It then independently encodes the query and all the triples within $\mathcal{G}_q$ as:

$$\boldsymbol{q}_i = \text{Enc}(q, z_{<i}),\ \boldsymbol{t}_j = \text{Enc}(t_j),\ \forall t_j \in \mathcal{G}_q, \quad (4)$$

where $\boldsymbol{q}_i$ denotes the query embedding and $\boldsymbol{t}_j$ is the embedding of a triple t_j within $\mathcal{G}_q$. Subsequently, the triple ranker uses the inner product, i.e., $\boldsymbol{q}_i^\top \boldsymbol{t}_j$, to estimate the relevance of a triple to the query and selects the top-K triples with the highest relevance as the candidate triples for the i-th step, i.e., $\hat{\mathcal{G}}_i$.

Triple Selector. At each i-th step, triple selector aims to select a triple from the candidate set $\hat{\mathcal{G}}_i$ to form a coherent reasoning chain with existing triples ($z_{<i}$). We use in-context learning to prompt an LLM instructed as the triple selector to select the triple. In particular, we formulate this task as a multiple-choice task, where each candidate triple is formatted as an option, e.g., "*B. ⟨Albert Einstein, father, Hermann Einstein⟩*". This multiple-choice task formulation can mitigate the hallucination of the LLM, as all the selected triples are grounded in the KG. Given the question, the previously selected triples and a list of options, the triple selector is instructed to output a single token representing the option of the selected triple, e.g., "B". Additionally, we introduce a special "A" option for the chain termination strategy, which will be introduced later. The prompt used by the triple selector to select the i-th triple is detailed in Appendix B.2.

Moreover, at each i-th step, we use the logits of the option tokens, which are output by the triple selector, to define a distribution for the i-th triple[4]:

$$p(z_i|q, z_{<i}, \hat{\mathcal{G}}_i) = \text{Softmax}(l(c_1), \ldots, l(c_K)), \quad (5)$$

where c_1 and c_K represent option tokens and $l(\cdot)$ denotes the logit of the corresponding option token. For example, if there are two candidate triples for the i-th step: "B. ⟨*Hermann Einstein, date of birth, 3 July 1814*⟩, C. ⟨*Albert Einstein, born, 14 March 1879*⟩", we use the logits of the tokens "B" and "C" to obtain the distribution for the i-th triple.

Chain Construction. With the triple distributions at all steps, we use beam search to generate R reasoning chains. At each step of the beam search, we select the top-b triples with the highest probability, as we empirically found that this approach can achieve satisfactory performance. The pseudo code and computational complexity analysis of reasoning chain construction process are in Appendix A.

[4]We found that calculating the triple distribution at each step and using beam search to generate multiple reasoning chains performs better than selecting one triple at each step, which results in only one reasoning chain (see Appendix D.5).

Moreover, considering that different multi-hop questions may require a varying number of triples in the reasoning chains, we introduce a special option at each step of the reasoning chain generation process: "A. *no need for additional triples*". This option indicates that previously selected triples are sufficient to answer the question, and no additional triples are needed. Once the triple selector chooses this option, the generation of the current reasoning chain is terminated. We refer to this approach as the *adaptive chain termination* strategy. Our empirical results show that this strategy significantly improves the performance compared to using fixed-length reasoning chains (see Appendix D.6).

3.4 Answer Generation

Finally, TRACE leverages a reader to generate the answer, i.e., $p(a|q, z, \mathcal{D}_q)$. We propose two methods to use the reasoning chains for generating the answer. In the first method, termed *TRACE-Triple*, we directly use the reasoning chains as context to generate the answer. The second method, termed *TRACE-Doc*, uses the triples within the reasoning chains to retrieve their original documents from $\mathcal{D}_q$ and then uses these documents as the context to generate the answer. For both methods, we found that the ordering of the input context is important for the answer generation performance. The constructed reasoning chains naturally present an ideal order of the triples, therefore *TRACE-Triple* directly concatenate these triples as the input context, achieving satisfactory performance. However, multiple triples from the reasoning chains could appear in a single document, requiring an alternative ordering method for the retrieved documents in *TRACE-Doc*. In practice, we found that a majority voting approach achieves satisfactory results. Specifically, each triple in the reasoning chains casts a vote for the document from which the triple is generated. We aggregate all the votes to rank the documents in $\mathcal{D}_q$ based on the number of votes they receive. Documents that receive no votes are filtered out.

4 Experiments

4.1 Experimental Setup

Datasets. We conduct experiments using three multi-hop QA datasets: **HotPotQA** (Yang et al., 2018), **2WikiMultiHopQA** (Ho et al., 2020) and **MuSiQue** (Trivedi et al., 2022). These datasets typically require 2-4 hops of reasoning to answer the questions. Each question in HotPotQA, 2WikiMultiHopQA and MuSiQue is associated with 10, 10 and 20 documents, respectively, all retrieved from Wikipedia. More details and statistics about the datasets are provided in Appendix C.1.

Baselines. Since TRACE builds reasoning chains from $\mathcal{D}_q$ to enhance RAG models, we mainly compare it against naive RAG baselines as well as other baselines capable of recognising supporting documents within $\mathcal{D}_q$: (1) naive baselines: *w/o documents*, where no documents are used in reader models, *w. all documents*, where all the documents are used in readers (i.e., the vanilla RAG); (2) bi-encoders: Contriever (Gautier et al., 2022), E5 (Wang et al., 2022), DRAGON+ (Lin et al., 2023), E5-Mistral (Wang et al., 2023); (3) cross-encoders: monoT5 (Nogueira et al., 2020), RankLLaMA (Ma et al., 2023), BGE (Xiao et al., 2023); (4) chain-of-thought (CoT): IRCoT (Trivedi et al., 2023). We use both bi-encoders and cross-encoders to rank the documents in $\mathcal{D}_q$ based on their estimated relevance scores. The top-M documents are used as context, where M is selected from $\{1, \ldots, N\}$ and set to the number of documents that results in the best performance on the development set of each dataset. For fair comparisons, both TRACE and our baselines use the same reader to generate answers. More details about the baselines are provided in Appendix C.2.

Evaluation. To evaluate the QA performance, we follow previous work (Ramesh et al., 2023) and use Exact Match (EM) and F1 as evaluation metrics, which are the standard metrics for these datasets.

Implementation Details. `LLaMA3-8B-Instruct` is used for both the KG generator and the triple selector. We employ `e5-mistral-7b-instruct` as the triple ranker. The analyses of the generated KGs and reasoning chains are in Appendix C.3. For the reader, we use in-context learning to generate answers in a zero-shot setting (see Appendix B.3), and report the performance of different readers, such as `LLaMA3-8B-Instruct`, `Mistral-7B-v0.1` and `Gemma-7B`. We mainly report the performance using LLaMA3 as the reader unless otherwise stated. Details about prompts and hyperparameters can be found in Appendix C.4.

4.2 Results and Analysis

We provide our main experimental results in this section. Additional results are in Appendix D.

Model	HotPotQA			2WikiMultiHopQA			MuSiQue		
	# Tok	EM	F1	# Tok	EM	F1	# Tok	EM	F1
LLaMA3 Reader with Context from Naive Baselines									
w/o documents	56	19.28	26.81	53	19.53	25.11	57	3.85	8.32
w. all documents	1,430	45.40	60.49	1,056	28.35	46.07	2,551	16.14	23.68
LLaMA3 Reader with Context from Bi-Encoders									
Contriever	1,430	47.10	62.39	894	28.01	46.41	767	20.23	27.86
DRAGON+	1,430	46.47	61.52	900	27.51	46.04	1,387	19.94	27.21
E5	920	47.52	62.96	633	29.02	46.52	1,079	24.08	31.71
E5-Mistral	1,430	48.25	63.72	639	31.44	48.83	1,099	26.40	33.66
LLaMA3 Reader with Context from Cross-Encoders									
monoT5	1,430	47.10	62.39	654	29.84	47.46	1,121	24.12	31.96
BGE	1,430	48.40	63.65	481	32.64	48.93	1,406	25.53	33.27
RankLLaMA	1,430	46.41	61.74	474	32.46	48.19	1,189	23.00	29.89
LLaMA3 Reader with Context from Chain-of-Thought Model									
IRCoT	454	50.78	65.65	553	36.11	52.25	571	27.40	36.91
LLaMA3 Reader with Context from TRACE									
TRACE-Triple	160	53.05*	67.32*	164	**45.51***	**56.43***	169	**33.43***	**40.05***
TRACE-Doc	357	**55.08***	**69.99***	485	42.74*	55.30*	456	32.44*	40.03*

Table 1: Overall performance (%) of TRACE and baselines on the test sets of multi-hop QA datasets, where "# Tok" is the average number of tokens in the documents/reasoning chains used as context, * indicates p-value < 0.05 compared with IRCoT. The best performance per dataset per metric is marked in boldface.

(RQ1): How does TRACE perform against the baselines? The QA performance of TRACE and baselines is reported in Table 1, yielding the following findings: (1) First, TRACE-Triple and TRACE-Doc consistently achieve the best performance wrt. all baselines, on all the datasets, demonstrating the effectiveness of TRACE in multi-hop QA tasks. Compared to the vanilla RAG model (i.e., *w. all documents*) TRACE-Triple and TRACE-Doc achieve average improvements of 14.03% and 13.46% in terms of EM across all datasets, respectively. This superior performance is due to TRACE's ability to effectively identify supporting evidence within the documents while avoiding the introduction of irrelevant context; (2) Moreover, compared to the CoT model IRCoT, the best performing baseline, TRACE-Triple and TRACE-Doc achieve significantly better performance, with average improvements of 5.90% and 5.32% in EM, respectively. The suboptimal performance of IRCoT may be due to LLMs' tendency to hallucinate and generate inaccurate CoT sentences (Luo et al., 2023). In contrast, the KG triples in TRACE's reasoning chains are grounded in the documents and selected for their relevance to the questions, ensuring a more reliable and accurate reasoning process; (3) Furthermore, TRACE-Triple, which only uses reasoning chains as context, achieves the best performance on the 2WikiMultiHopQA and MuSiQue datasets, and the second-best performance on the HotPotQA dataset in EM. Notably, the reasoning chains used by TRACE-Triple contain fewer tokens compared to the documents used by other models, yet it achieves the best performance in most cases. This result indicates that it is unnecessary to use all the information within documents; instead, constructing reasoning chains to identify and integrate supporting evidence within the documents is often sufficient to correctly answer questions.

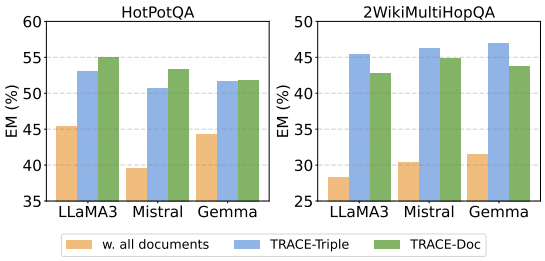

Figure 3: Performance of TRACE with different readers on the test sets of HotPotQA and 2WikiMultiHopQA.

Additionally, we report the QA performance of TRACE using different reader models. Due to the space limit, we provide the EM performance of vanilla RAG (*w. all documents*) and our models on the HotPotQA and 2WikiMultiHopQA datasets in Figure 3. The complete results for different readers are in Appendix D.1. The results in Figure 3 indicate that both TRACE-Triple and TRACE-Doc outperform the baseline by a large margin when using different readers, demonstrating that the reasoning chains generated by TRACE can effectively generalise across various readers.

Model	HotPotQA		2WikiMultiHopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1
TRACE-Triple	**67.00**	**76.16**	**66.40**	**72.52**	**40.40**	**46.92**
Effectiveness of KG Generator						
w. sentences	65.60*	74.72*	63.40*	71.00*	35.40*	41.95*
w. documents	65.00*	75.93*	58.60*	68.87*	30.80*	35.84*
Effectiveness of Reasoning Chain Constructor						
w. Top-10 Triples	55.40*	65.67*	43.80*	47.99*	30.00*	36.44*
w. Top-20 Triples	59.40*	69.84*	48.00*	53.12*	30.00*	36.77*
w. Top-25 Triples	59.40*	70.29*	48.40*	54.13*	30.60*	37.03*
Effectiveness of Triple Ranker						
w/o Triple Ranker	58.40*	67.71*	63.40*	69.05*	26.40*	34.55*
w. DRAGON+	62.80*	72.86*	64.40*	70.55*	36.30*	42.16*
w. E5	64.00*	73.88*	65.20	71.10	36.60*	43.18*
Effectiveness of Triple Selector						
w/o Triple Selector	53.80*	63.80*	47.00*	52.31*	26.20*	31.24*
w. Mistral	61.80*	71.04*	65.60	72.01	34.60*	41.49*
w. Gemma	59.00*	68.52*	63.40*	69.26*	28.80*	35.08*

Table 2: Performance (%) of TRACE-Triple and its variants on the development sets of three QA datasets, where * denotes p<0.05 compared with TRACE-Triple.

(RQ2): What is the advantage of generating KGs from documents? To investigate the advantage of generating KGs, we introduce two variants of TRACE-Triple[5]: *w. sentences* and *w. documents*, where the KG generator is removed and the triples are replaced with sentences and documents from $\mathcal{D}_q$, respectively. These sentences or documents are passed to the reasoning chain constructor to build sentence-based or document-based reasoning chains. The QA performance of TRACE-Triple and its variants is reported in Table 2. The results show that TRACE-Triple significantly outperforms the two variants on all datasets. This is because, compared with sentences or documents, KG triples provide a finer-grained and more concise way of storing knowledge, containing less irrelevant information. Therefore, using KG triples to identify supporting evidence can mitigate the impact of irrelevant information, improving the accuracy of the reasoning process. We conduct the same experiments on TRACE-Doc. The results are in Appendix D.2, which demonstrate similar outcomes.

(RQ3): Can the reasoning chain constructor effectively identify supporting evidence? To investigate the effectiveness of the chain constructor, we introduce a variant of TRACE-Triple: *w. Top-T Triples*, where we remove the constructor and directly use E5-Mistral to retrieve the top-T most relevant triples from the KG. These triples are used in a manner similar to reasoning chains. The results of TRACE-Triple and the variants are reported in Table 2, which indicate that removing

[5]An efficiency analysis of TRACE is in Appendix D.8.

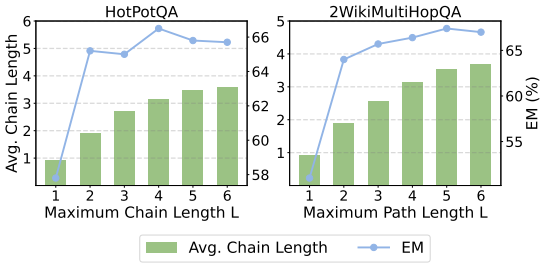

Figure 4: QA performance (%) and average chain length of TRACE-Triple under different values of L on the development sets of HotPotQA and 2WikiMultiHopQA.

the chain constructor significantly degrades the performance on all the datasets. This is because the reasoning chain constructor identifies supporting evidence in an autoregressive manner. The previously identified supporting evidence provides cues for identifying the subsequent evidence, thereby facilitating the identification of all the supporting evidence and leading to improved performance.

(RQ4): What are the effects of each component, i.e., triple ranker and triple selector, in the reasoning chain constructor? To examine the effectiveness of the triple ranker, we introduce a variant of TRACE-Triple: *w/o Triple Ranker*, where the triple ranker is removed and the whole graph is used as the candidate set. The results in Table 2 indicate that removing the triple ranker significantly degrades the performance on all the datasets. The superior performance of using the triple ranker is due to its ability to identify triples that are relevant to the current context while avoiding the introduction of irrelevant ones. We next examine the impact of different choices for the triple ranker. In addition to the E5-Mistral model used in TRACE-Triple, we also test DRAGON+ and E5 as the triple ranker, denoted as *w. DRAGON+* and *w. E5*, respectively. The results in Table 2 show that different triple rankers impact the final performance, with E5-Mistral achieving the best results.

Moreover, to investigate the effectiveness of the triple selector, we introduce a variant of TRACE-Triple: *w/o Triple Selector*, where the triple selector is removed and the top-b triples ranked by the triple ranker are used to construct reasoning chains. The results in Table 2 show that removing the triple selector significantly degrades performance on all the datasets, indicating the importance of using the triple selector's reasoning ability in constructing coherent reasoning chains. We also examine the

Question: Are both Blaise Cendrars and Julian Barnes are a citizen of the same country? **Reasoning Chain**: ⟨Blaise Cendrars; nationality; Swiss⟩, ⟨Julian Barnes; nationality; English⟩ **Generated Answer**: no
Question: What was the occupation of both Christina Stead and Nuruddin Farah? **Reasoning Chain**: ⟨Christina Stead; occupation; novelist and short-story writer⟩, ⟨Nuruddin Farah; occupation; novelis⟩ **Generated Answer**: novelist
Question: What is the birth date of this Spanish footballer, who was added as a holding midfielder in the 2012-13 FC Bayern Munich season? **Reasoning Chain**: ⟨2012–13 FC Bayern Munich season; new player signed after the first week of the Bundesliga season; Javi Martínez⟩, ⟨Javi Martínez; date of birth; 2 September 1988⟩ **Generated Answer**: 2 September 1988

Table 3: Case Study of TRACE on HotPotQA dataset.

impact of different choices for the triple selector. In addition to LLaMA3 used in TRACE-Triple, we also use Mistral and Gemma as the triple selector, denoted as *w. Mistral* and *w. Gemma*, respectively. The results in Table 2 suggest that different triple selectors also affect the final performance, with LLaMA3 achieving the best performance.

(RQ5): How does the maximum chain length L affect the performance? We conduct experiments to investigate the effects of the hyperparameter L. Specifically, we vary the value of L from 1 to 6 and report the corresponding results, including the average chain length and the QA performance. Figure 4 shows the results of TRACE-Triple on HotPotQA and 2WikiMultiHopQA. The results on MuSiQue are in Appendix D.3. The figure shows that when increasing L from 1 to 6, the average chain length does not increase linearly; instead, the growth of the average chain length gradually slows down. This is due to the effectiveness of our adaptive chain termination strategy in automatically determining the suitable lengths of reasoning chains. Moreover, the figure also shows that as we increase L, the QA performance initially increases but then decreases after a certain threshold, such as 4 on HotPotQA. This decline in performance may be due to the introduction of irrelevant or redundant information in longer reasoning chains, which can confuse the reader and degrade the performance.

(RQ6): How does the number of candidate triples K affect the performance? In the reasoning chain constructor of TRACE, a triple ranker

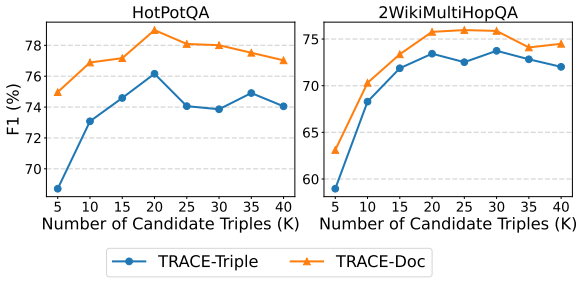

Figure 5: Performance (%) of TRACE with different numbers of candidate triples on the development sets of the HotPotQA and 2WikiMultiHopQA datasets.

is used to select a subset of candidate triples from the generated KGs. In order to investigate the effects of the number of candidate triples K, we vary the value of K from 5 to 40 and report the corresponding performance. The results are reported in Figure 5, which shows the performance of TRACE-Triple and TRACE-Doc on the HotPotQA and 2WikiMultiHopQA datasets. These results indicate that as K increases, the performance of both TRACE-Triple and TRACE-Doc initially improves but then declines after a certain threshold, such as 20 for HotPotQA and 30 for 2WikiMultiHopQA. This can be explained by the balance between information richness and noise. Initially, increasing K allows for the inclusion of potentially relevant triples, thereby improving the performance. However, beyond a certain threshold, using too many triples can introduce noise and irrelevant information, which can distract the triple selector and degrade the overall performance.

Case Study. We conduct a case study to investigate the reasoning chains generated by TRACE. Table 3 shows the reasoning chains and final answers for questions on HotPotQA[6]. These examples demonstrate that TRACE can construct effective reasoning chains to answer multi-hop questions. Moreover, these reasoning chains also provide an interpretable framework for understanding how the model generates its answers, enhancing the transparency and reliability of the reasoning process.

5 Related Work

RAG Models. RAG models have demonstrated impressive performance in QA tasks (Karpukhin et al., 2020; Lewis et al., 2020; Izacard and Grave,

[6]The complete reasoning chains and the identified relevant documents for these questions can be found in Appendix D.7.

2021; Ram et al., 2023; Shi et al., 2023b). These models typically employ a retriever-reader pipeline, which consists of a retriever (Karpukhin et al., 2020; Wang et al., 2022; Fang et al., 2023) for retrieving relevant information and a reader (Izacard and Grave, 2021; Jiang et al., 2023b) for generating answers. Recently, RAG models have been used to address multi-hop questions (Yavuz et al., 2022; Ramesh et al., 2023). For instance, the IR-CoT (Trivedi et al., 2023) uses CoT sentences to retrieve documents and generate answers. Since the LLMs are prone to hallucinate, the generated CoT sentences may be inaccurate (Luo et al., 2023; Nguyen et al., 2024) and lead to suboptimal performance. In contrast, our reasoning chains are grounded on the retrieved documents, ensuring a more reliable and accurate reasoning process. Moreover, recent works have shown that the retrieval of irrelevant documents can hinder the performance of RAG models (Petroni et al., 2020; Creswell et al., 2023; Shi et al., 2023a), especially when tackling multi-hop questions (Yoran et al., 2024). Previous works require training to mitigate the effects of irrelevant documents (Ramesh et al., 2023; Yoran et al., 2024), while our TRACE uses in-context learning and does not require training.

RAG Models with KGs. RAG models have been used to address the knowledge graph question answering task (Jiang et al., 2023a; Luo et al., 2023), where the answers are restricted to be entities in an existing KG (Xiong et al., 2023, 2024), while our work focuses on a more general setting where the answers can be any words or phrases. Under this setting, there are some works that leverage KGs to enhance RAG models (Min et al., 2019; Zhou et al., 2020; Oguz et al., 2022; Yu et al., 2022; Ju et al., 2022; Fang et al., 2024). For example, UniK-QA (Oguz et al., 2022) combine KGs into a corpus for retrieval and KG-FiD (Yu et al., 2022) uses KGs to construct passage graphs for passage reranking. However, these works all use information from existing KGs, such as Wikidata (Vrandečić and Krötzsch, 2014), while TRACE creates a KG based on the retrieved documents and constructs reasoning chains for multi-hop reasoning. Recently, GraphRAG (Edge et al., 2024) also propose to generate KGs from documents. However, GraphRAG and our TRACE focus on different tasks. Specifically, Graph RAG is designed for the global question answering task, which aims to generate a summary for open-ended questions that do no have clear answers. In comparison, TRACE focuses on the multi-hop QA task, which aims to generate answers for multi-hop questions that have clear answers.

6 Conclusion

This paper proposes TRACE to enhance the multi-hop reasoning ability of RAG models. Specifically, TRACE employs a KG generator to create a KG from the retrieved documents and then uses a novel autoregressive reasoning chain constructor to build reasoning chains from the KG. Given the reasoning chains, TRACE either directly uses them as context to generate answers or uses them to identify a subset of relevant documents. Experimental results on three multi-hop QA datasets show that, compared to using all the retrieved documents, TRACE achieves an average performance of up to 14.03% in EM. Moreover, the reasoning chains can effectively generalise across various reader models.

Limitations

We identify the following limitations of our work: (1) Since this work focuses on documents retrieved from Wikipedia, the KG generator primarily generates KG triples between the title and entities within the texts, rather than between all possible entities. This approach simplifies the task, which would otherwise be more challenging, and we defer such exploration to future work; (2) Due to the lack of annotated data, we are unable to directly and quantitatively evaluate the quality the generated KG triples and the constructed reasoning chains. Instead, we assess their performance through the final QA performance. Moreover, we provide the statistics of the generated KGs and reasoning chains, as well as some qualitative results in Appendix C.3 to offer some insights into their structure and effectiveness. We consider improving the evaluation methods for these two modules as one of our future work directions; (3) In the reasoning chain constructor, TRACE requires access to the logits of the triple selector to define a triple distribution. However, this may not be feasible when using a black-box LLM as the triple selector. In such cases, TRACE can use the option token output by the triple selector to select one triple at each step, leading to a single reasoning chain.

References

Antonia Creswell, Murray Shanahan, and Irina Higgins. 2023. Selection-inference: Exploiting large language models for interpretable logical reasoning. In *The Eleventh International Conference on Learning Representations*.

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

Jinyuan Fang, Zaiqiao Meng, and Craig Macdonald. 2023. KGPR: Knowledge graph enhanced passage ranking. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 3880–3885.

Jinyuan Fang, Zaiqiao Meng, and Craig Macdonald. 2024. REANO: Optimising retrieval-augmented reader models through knowledge graph generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2094–2112.

Izacard Gautier, Caron Mathilde, Hosseini Lucas, Riedel Sebastian, Bojanowski Piotr, Joulin Armand, and Grave Edouard. 2022. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

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.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, pages 874–880.

Gautier Izacard, Patrick S. H. Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24:251:1–251:43.

Jinhao Jiang, Kun Zhou, Xin Zhao, and Ji-Rong Wen. 2023a. UniKGQA: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graph. In *The Eleventh International Conference on Learning Representations*.

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 *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992.

Mingxuan Ju, Wenhao Yu, Tong Zhao, Chuxu Zhang, and Yanfang Ye. 2022. GRAPE: Knowledge graph enhanced passage reader for open-domain question answering. In *Findings of the Association for Computational Linguistics*, pages 169–181.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 6769–6781.

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. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Xiaonan Li, Kai Lv, Hang Yan, Tianyang Lin, Wei Zhu, Yuan Ni, Guotong Xie, Xiaoling Wang, and Xipeng Qiu. 2023. Unified demonstration retriever for in-context learning. *arXiv preprint arXiv:2305.04320*.

Sheng-Chieh Lin, Akari Asai, Minghan Li, Barlas Oguz, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. 2023. How to train your dragon: Diverse augmentation towards generalizable dense retrieval. In *Findings of the Association for Computational Linguistics: EMNLP*, pages 6385–6400.

Xi Victoria Lin, Xilun Chen, Mingda Chen, Weijia Shi, Maria Lomeli, Rich James, Pedro Rodriguez, Jacob Kahn, Gergely Szilvasy, Mike Lewis, et al. 2024. RA-DIT: Retrieval-augmented dual instruction tuning. In *International Conference on Learning Representations*.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

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

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2023. Fine-tuning llama for multi-stage text retrieval. *arXiv preprint arXiv:2310.08319*.

Sewon Min, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Knowledge guided text retrieval and reading for open domain question answering. *arXiv preprint arXiv:1911.03868*.

Minh-Vuong Nguyen, Linhao Luo, Fatemeh Shiri, Dinh Phung, Yuan-Fang Li, Thuy-Trang Vu, and Gholamreza Haffari. 2024. Direct evaluation of chain-of-thought in multi-hop reasoning with knowledge graphs. *arXiv preprint arXiv:2402.11199*.

Rodrigo Frassetto Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document ranking with a pretrained sequence-to-sequence model. In *Findings of the Association for Computational Linguistics: EMNLP*, pages 708–718.

Barlas Oguz, Xilun Chen, Vladimir Karpukhin, Stan Peshterliev, Dmytro Okhonko, Michael Sejr Schlichtkrull, Sonal Gupta, Yashar Mehdad, and Scott Yih. 2022. UniK-QA: Unified representations of structured and unstructured knowledge for open-domain question answering. In *Findings of the Association for Computational Linguistics*, pages 1535–1546.

Fabio Petroni, Patrick S. H. Lewis andAleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. In *Conference on Automated Knowledge Base Construction*.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331.

Gowtham Ramesh, Makesh Narsimhan Sreedhar, and Junjie Hu. 2023. Single sequence prediction over reasoning graphs for multi-hop QA. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11466–11481.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2655–2671.

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

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2023b. REPLUG: Retrieval-augmented black-box language models. *arXiv preprint arXiv:2301.12652*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multi-hop questions via single-hop question composition. *Trans. Assoc. Comput. 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.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

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. *arXiv preprint arXiv:2212.03533*.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2023. Improving text embeddings with large language models. *arXiv preprint arXiv:2401.00368*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022.

Xiang Wei, Xingyu Cui, Ning Cheng, Xiaobin Wang, Xin Zhang, Shen Huang, Pengjun Xie, Jinan Xu, Yufeng Chen, Meishan Zhang, et al. 2023. Zero-shot information extraction via chatting with ChatGPT. *arXiv preprint arXiv:2302.10205*.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighof. 2023. C-pack: Packaged resources to advance general chinese embedding. *arXiv preprint arXiv:2309.07597*.

Siheng Xiong, Ali Payani, Ramana Kompella, and Faramarz Fekri. 2024. Large language models can learn temporal reasoning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pages 10452–10470. Association for Computational Linguistics.

Siheng Xiong, Yuan Yang, Faramarz Fekri, and James Clayton Kerce. 2023. TILP: differentiable learning of temporal logical rules on knowledge graphs. In *The Eleventh International Conference on Learning Representations*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380.

Semih Yavuz, Kazuma Hashimoto, Yingbo Zhou, Nitish Shirish Keskar, and Caiming Xiong. 2022. Modeling multi-hop question answering as single sequence prediction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 974–990.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2024. Making retrieval-augmented language models robust to irrelevant context. In *International Conference on Learning Representations*.

Donghan Yu, Chenguang Zhu, Yuwei Fang, Wenhao Yu, Shuohang Wang, Yichong Xu, Xiang Ren, Yiming Yang, and Michael Zeng. 2022. KG-FiD: Infusing knowledge graph in fusion-in-decoder for open-domain question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 4961–4974.

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

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

Mantong Zhou, Zhouxing Shi, Minlie Huang, and Xiaoyan Zhu. 2020. Knowledge-aided open-domain question answering. *arXiv preprint arXiv:2006.05244*.

A Reasoning Chain Construction Algorithm and Complexity Analysis

The algorithm for the reasoning chain construction process is detailed in Algorithm 1, which illustrates the construction of R reasoning chains from a set of documents $\mathcal{D}_q$ to answer a question q. For clarity, we omit the "adaptive chain termination" strategy in the algorithm. However, it can be easily incorporated into the algorithm if needed.

The computational complexity of constructing the reasoning chains is $\mathcal{O}(N + LR)$, where N is the number of documents in $\mathcal{D}_q$ and L denotes the length of the reasoning chains. This complexity applies when the adaptive chain termination strategy is not used. Incorporating this strategy can improve the actual efficiency further. In most cases, LR tends to dominate N, especially when the number of reasoning steps L and the number of relevant documents R are relatively large.

B Prompts

In this section, we present the prompts used in our TRACE. Specifically, the prompt for generating KG triples is detailed in § B.1, the prompt for the triple selector is outlined in § B.2, and the prompt for generating answers in the reader is introduced in § B.3. Finally, we provide some examples of demonstrations in § B.4.

B.1 Prompt for Generating KG Triples

The KG generator independently generates KG triples for each document within $\mathcal{D}_q$. The prompt used for the KG generator to generate KG triples from a document is provided in Figure 6. Specifically, the prompt comprises three parts: instruction, demonstrations and input document. The instruction defines the task of generating KG triples from a document. The demonstrations are examples of documents and their corresponding KG triples. The input document is the document from which we expect the KG generator to generate KG triples. The output of the KG generator is all the possible KG triples identified within the input document.

B.2 Prompt for Triple Selector

The triple selector aims to select a triple one by one to construct reasoning chains. The prompt used by the triple selector to select the i-th triple is provided in Figure 7. Specifically, the prompt consists of three parts: instruction, demonstrations and inputs. The instruction defines the task of selecting a triple to form coherent reasoning chains. The demonstrations are examples of complete reasoning chains and how the i-th triples in these chains are selected. The inputs consist of the question q, existing knowledge triples $z_{<i}$ and the candidate set $\hat{\mathcal{G}}_i$. Moreover, the triple selector is instructed to output only the option of the selected triples, such as "A", "B".

B.3 Prompt for Generating Answers

In our experiments, we use in-context learning to prompt the reader to generate answers in a zero-shot setting. The prompt used for the answer generation is provided in Figure 8. Moreover, the reader is instructed to output only the answer to the question based on the given context, such as reasoning chains or documents.

B.4 Demonstration Data

Examples of the demonstrations for the KG generator and the triple selector are provided in Table 10 and Table 11, respectively. We manually labeled 20 examples for each dataset, and used them as demonstrations to guide the LLMs in generating desired outputs.

C Experimental Details

Due to the space limit, we provide additional experimental details in this section, which are com-

Algorithm 1: Knowledge Triple-Grounded Reasoning Chains Construction of TRACE.

Input : question q, a set of retrieved documents $\mathcal{D}_q = \{d_1, d_2, \ldots, d_N\}$, maximum reasoning chain length L, number of candidate triples K, number of reasoning chains R

```
/* Part I: Knowledge Graph Generation.                                    */
```
$\mathcal{G}_q = \{\}$; # Initialise KG
for $d_i \in \mathcal{D}_q$ **do**
  $\mathcal{G}_{q,d_i} = KG\_Generator(d_i)$;
  $\mathcal{G}_q = \mathcal{G}_q$.add($\mathcal{G}_{q,d_i}$);

```
/* Part II: Reasoning Chain Construction.                                 */
```
z =[[] for _ in range(R)]; c =[[q] for _ in range(R)]; s =[[1.0] for _ in range(R)];
for $i = 1, 2, \ldots, L$ **do**
  c_z, c_c, c_s =[], [], [];
  for $r = 1, 2, \ldots, R$ **do**
    $\hat{\mathcal{G}}_i = Triple\_Scorer(c[r], \mathcal{G}_q, K)$;
    $p(z_i|q, z_{<i}, \hat{\mathcal{G}}_i) = Triple\_Selector(c[r], \hat{\mathcal{G}}_i)$;
    for $z_{i,j} \in argmax_b(p(z_i|q, z_{<i}, \hat{\mathcal{G}}_i))$ **do**
      c_z.append($z[r]$ + $[z_{i,j}]$);
      c_c.append($c[r]$ + $[z_{i,j}]$);
      c_s.append($s[r] * p(z_i = z_{i,j}|q, z_{<i}, \hat{\mathcal{G}}_i)$);
  indices = $argmax_R(c\_s)$; $z = c\_z$[indices]; $c = c\_c$[indices];

Output : Reasoning Chains z.

Model	Huggingface Checkpoint
Contriever	`contriever`
DRAGON+	`dragon-plus`
E5	`e5-large-v2`
E5-Mistral	`e5-mistral-7b-instruct`
monoT5	`monot5-large-msmarco`
BGE	`bge-reranker-large`
RankLLaMA	`rankllama-v1-7b-lora-passage`

Table 4: The specific retrieval models used in our experiments.

plementary to § 4.1 in the main text. Specifically, we introduce further details about the documents and the data splits of the experimental datasets in § C.1. We then detail the specific parameterisations of the baselines in § C.2. Moreover, we provide the statistics and analyses of the generated KGs and reasoning chains in § C.3. Finally, we include additional implementation details, such as prompt demonstrations and hyperparameters, in § C.4.

C.1 Datasets

We use three multi-hop QA datasets: HotPotQA, 2WikiMultiHopQA, and MuSiQue. Each question in these datasets is provided with a set of documents retrieved from Wikipedia, which include both relevant and irrelevant documents. These documents are randomly shuffled in the original datasets. Additionally, the datasets provide annotations indicating which documents are relevant to each question. Since TRACE aims to identify supporting evidence within the retrieved documents, we directly use these documents as inputs and perform multi-hop reasoning over these documents to generate answers. Since the test sets of these datasets are not publicly available, we follow Ramesh et al. (2023) and report the performance on the original development sets, which contain 7,405 questions for HotPotQA, 12,576 questions for 2WikiMultiHopQA, and 2,417 questions for MuSiQue. We randomly sample 500 questions from the training set of each dataset to create our development sets for hyperparameter tuning.

C.2 Baselines

In order to evaluate the effectiveness of TRACE in identifying supporting evidence, we mainly compare with baselines that use different methods to identify relevant documents from the document set $\mathcal{G}_q$. Once the relevant documents are obtained, they are used as input to the same reader as TRACE to generate answers. Specifically, we compare with baselines from the following categories:

Instruction:

Given a title and a text, extract all the knowledge triples in the form of ⟨title; relation; tail entity⟩, where title is the provided title, tail entity is a phrase in the text and relation denotes a description of the relation between the title and the tail entity.

Demonstrations:

Title: Albert Einstein
Text: Albert Einstein (14 March 1879-18 April 1955) was a German-born theoretical physicist.
Knowledge Triples:
⟨Albert Einstein; date of birth; 14 March 1879⟩
⟨Albert Einstein; date of death; 18 April 1955⟩
⟨Albert Einstein; place of birth; German⟩
⟨Albert Einstein; occupation; theoretical physicist⟩
. . .

Input Document:

Title: Kelie McIver
Text: Kelie McIver is a Kansas-born actress and singer who has played classical stage roles such as Lady Macbeth and Nurse in "Romeo & Juliet" for Kingsmen Shakespeare Festival.
Knowledge Triples:

Figure 6: Prompt for generating knowledge triples from a document.

Instruction:

Select the next knowledge triple that extends an existing set of knowledge triples to form a coherent reasoning path capable of answering a specified question. If the current reasoning path is sufficient to answer the question, simply output A. Please only output the choice for the next knowledge triple.

The following are some examples of coherent reasoning paths capable of answering the specified question and how the l-th knowledge triples in these paths are selected:

Demonstrations:

coherent reasoning path: ⟨A Girl's Gotta Do (What a Girl's Gotta Do); artist; American country music artist Mindy McCready⟩, ⟨Mindy McCready; fifth album; I'm Still Here⟩
question: What is the 5th studio album released by the singer of "A Girl's Gotta Do (What a Girl's Gotta Do)"?

The l-th triple in the reasoning path is selected as:
existing knowledge triples: [Previously selected triples in the form of ⟨·⟩, ⟨·⟩, . . .]
question: What is the 5th studio album released by the singer of "A Girl's Gotta Do (What a Girl's Gotta Do)"?
candidate knowledge triples:
A. no need for additional knowledge triples
B. ⟨A Girl's Gotta Do (What a Girl's Gotta Do); artist; American country music artist Mindy McCready⟩
C. ⟨A Girl's Gotta Do (What a Girl's Gotta Do); release date; February 1997⟩
D. ⟨Ten Thousand Angels; fourth single; "A Girl's Gotta Do (What a Girl's Gotta Do)"⟩
E. ⟨A Girl's Gotta Do (What a Girl's Gotta Do); songwriters; Robert Byrne and Rick Bowles⟩
the next possible triple is:B
. . .

Input Document:

The l-th triple in the reasoning path is selected as:
existing knowledge triples: [Previously selected triples in the form of ⟨·⟩, ⟨·⟩, . . .]
question: Are Ellen Glasgow and Günter Grass both novelists?
candidate knowledge triples:
A. no need for additional knowledge triples
B. ⟨Ellen Glasgow; occupation; novelist⟩
C. ⟨Virginia (novel); author; Ellen Glasgow⟩
D. ⟨Ellen Glasgow; full name; Ellen Anderson Gholson Glasgow⟩
E. ⟨Günter Grass; occupation; novelist, poet, playwright, illustrator, graphic artist, sculptor⟩
the next possible triple is:

Figure 7: Prompt for selecting the i-th triples of the reasoning chains.

w/o documents: In this baseline, we remove all the documents and use only the questions as inputs for the reader to generate answers.

w. all documents: In this baseline, we use both the questions and all the documents as inputs for the reader (i.e., the vanilla RAG). Note that the

Instruction:
Given some contexts and a question, please only output the answer to the question.
context:
... (reasoning chains / documents)
the correct answer is:

Figure 8: Prompt for generating answers to the questions based on given context.

	HotPotQA		2WikiMultiHopQA		MuSiQue	
	Dev.	Test	Dev.	Test	Dev.	Test
Statistics of Multi-Hop QA datasets						
# Questions	500	7,405	500	12,576	500	2,417
# Documents per Question	10	10	10	10	20	20
Avg. Document Error Rate (%)	79.52	79.66	75.68	75.62	88.19	86.74
Statistics of the Generated KG						
Avg. # Entities per Question	83.21	83.63	65.67	71.21	159.87	164.03
Avg. # Triples per Question	79.06	79.04	62.75	67.46	150.53	154.70
Avg. Density per Question (%)	1.23	1.20	1.57	1.42	0.61	0.59
Statistics of the Constructed Reasoning Chains						
Avg. Reasoning Chain Length	3.15	3.16	3.14	3.36	3.14	3.17
Avg. # Relevant Documents per Question	2.63	2.57	2.73	2.82	2.70	2.84
Avg. Document Error Rate (%)	22.78	21.06	12.64	14.82	28.48	29.76

Table 5: Statistics of the generated KGs and reasoning chains, where "Avg. # Relevant Documents per Question" denotes the average number of relevant documents identified with reasoning chains and "Avg. Document Error rate (%)" denotes the average percentage of documents that are irrelevant to the questions.

Model	HotPotQA		2WikiMultiHopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1
TRACE-Doc	**69.20**	**78.46**	**68.00**	**75.96**	**41.00**	**47.09**
Effectiveness of KG Generator						
w. sentences	67.20*	77.56*	65.00*	73.06*	33.60*	39.62*
w. documents	66.20*	76.73*	57.60*	67.83*	30.80*	36.13*
Effectiveness of Reasoning Chain Constructor						
w. Top-10 Triples	63.80*	73.30*	49.80*	56.27*	32.40*	38.11*
w. Top-20 Triples	62.60*	73.60*	52.40*	59.97*	30.80*	37.48*
Effectiveness of Triple Ranker						
w/o Triple Ranker	63.40*	73.79*	62.20*	71.59*	29.40*	35.18*
w. DRAGON+	64.60*	74.92*	65.00*	73.67*	36.20*	41.76*
w. E5	67.80*	77.82*	67.20	75.27	39.00	45.96
Effectiveness of Triple Selector						
w/o Triple Selector	59.80*	69.38*	49.20*	55.39*	30.00*	35.56*
w. Mistral	64.40*	74.91*	65.60*	74.13	35.60*	42.65*
w. Gemma	63.60*	72.49*	63.80*	72.13*	31.00*	37.17*

Table 6: Performance (%) of TRACE-Doc and its variants on the development sets of three QA datasets, where * denotes p<0.05 compared with TRACE-Doc.

documents provided by the datasets, i.e., $\mathcal{D}_q$, are already randomly shuffled. We do not perform any ranking on these documents; instead, we directly concatenate all the documents in their original order.

bi-encoders/cross-encoders: To identify relevant documents within $\mathcal{G}_q$ with retrieval models, such as bi-encoders and cross-encoders, we use these models to estimate the relevance scores between a question and all the documents within $\mathcal{G}_q$. We then rank these documents in descending order based on the estimated relevance scores. The top-M documents are considered relevant and used as inputs for the reader, where M is selected from $\{1, 2, \ldots, N\}$ and is set to the number of documents that results in the best performance of the reader on the development set of each dataset. The checkpoints we use for different retrieval models are in Table 4.

IRCoT: IRCoT was originally proposed to leverage chain-of-thought (CoT) sentences for both document retrieval and answer generation. Here, we use it solely to retrieve relevant documents from $\mathcal{G}_q$. For a fair comparison with TRACE, we use the same `LLaMA3-8B-Instruct` model to generate CoT sentences. Following the original methodology, we alternate between CoT sentence generation and document retrieval to retrieve a set of relevant documents. The resulting documents are used as input to the reader to generate answers.

C.3 Statistics and Analyses of the Generated KGs and Reasoning Chains

The statistics of the generated KGs and reasoning chains for our experimental datasets are provided in Table 5. Specifically, we report the average number of entities and triples, as well as the average density in the KGs. For example, in the test set

of the HotPotQA dataset, the average number of entities and triples is 83.21 and 79.06, respectively. The corresponding average density is 1.20%, indicating that the KGs are highly sparse. The same results can also be observed in other datasets. This is because it is challenging to infer relationships between entities from different documents, or there may be no meaningful relationships between these entities, resulting in fewer connections and a lower overall density in the KGs.

Moreover, the results in Table 5 show that the average length of reasoning chains is relatively small, approximately 3 across all datasets. Leveraging these reasoning chains to identify a subset of relevant documents from $\mathcal{D}_q$ results in an average of around 2.6 documents across all datasets. Despite the small number of documents, the error rates of these documents are significantly lower than those of the original documents $\mathcal{D}_q$, e.g., 21.06% v.s. 79.66% on the test set of HotPotQA. These results demonstrate the effectiveness of leveraging reasoning chains to identify a subset of relevant documents from $\mathcal{D}_q$ while avoiding the introduction of irrelevant ones. This can also explain the superior performance of TRACE-Doc over the vanilla RAG model, as the documents identified by TRACE-Doc contain significantly less noise.

Furthermore, due to the lack of annotated data, we are unable to quantitatively evaluate the effectiveness of the generated KGs and reasoning chains. Instead, we provide some qualitative results to assess their performance. Particularly, we provide some examples of the generated KGs and reasoning chains on HotPotQA dataset in Table 13 and Table 14, respectively. The results in Table 13 show that our KG generate can produce high-quality knowledge triples. Moreover, almost all the generated knowledge triples are grounded in the documents, with minimal instances of hallucinations. This reliable grounding serves as a solid foundation for the subsequent reasoning, ensuring the accuracy and effectiveness of the reasoning process. Additionally, detailed analyses of the generated reasoning chains can be found in § D.7.

C.4 Additional Implementation Details

We conduct experiments in a zero-shot setting. The context, such as reasoning chains or documents, is prepended to questions, which are then passed to the reader for answer generation. We use greedy decoding to generate answers in the reader.

Moreover, to obtain demonstrations for the KG generator, we manually label 50 documents from the training set of each dataset. Examples of the labelled data are provided in Appendix B.4. The complete labelled data for each dataset are available in our Github repository. Following previous works (Rubin et al., 2022; Li et al., 2023), we use E5-Mistral to retrieve the top three most similar documents from the labelled set as demonstrations when generating the KG for a document. Similarly, for the reasoning chain constructor, we manually label 20 questions from the training set of each dataset. The demonstrations for the reasoning chain constructor are obtained in a similar manner to the KG generator, using E5-Mistral to retrieve the top three most similar questions from the labelled set as the demonstrations. We empirically verify the effectiveness of the demonstrations in Appendix D.4.

Furthermore, throughout the experiments, we set the maximum reasoning chain length L as 4, the number of reasoning chains R as 5 and the number of beams b as 5. The number of candidate triples K is chosen from $\{15, 20, 25, 30\}$ to achieve the best performance on the development set.

D Additional Experimental Results and Analysis

In this section, we first introduce the overall performance of TRACE using different readers in § D.1. We then introduce the ablation studies for TRACE-Doc in § D.2 and the effects of maximum chain length L on the MuSiQue dataset in § D.3. Subsequently, we investigate the effects of the demonstrations, the effects of the number of reasoning chains R in § D.4 and § D.5, respectively. In addition, we examine the effectiveness of the adaptive chain termination strategy in § D.6. Furthermore, we provide the results and analyses of the case study in § D.7. Finally, we provide the efficiency analysis of our model in § D.8.

D.1 Overall performance of TRACE using Different Readers

In order to examine the generalisation of reasoning chains, we report the performance of TRACE and baselines using different readers. Specifically, we leverage `Mistral-7B-v0.1` and `Gemma-7B` as the readers and report the corresponding performance in Table 7 and Table 8, respectively. The results are similar to those obtained using LLaMA3 as the reader, demonstrating that the reasoning chains

Model	HotPotQA			2WikiMultiHopQA			MuSiQue		
	# Tok	EM	F1	# Tok	EM	F1	# Tok	EM	F1
Mistral Reader with Context from Naive Baselines									
w/o documents	46	21.20	28.68	42	23.61	27.20	46	4.22	8.85
w. all documents	1,627	39.61	52.46	1,190	30.42	37.93	2,913	13.82	20.02
Mistral Reader with Context from Bi-Encoders									
Contriever	1,627	39.69	52.46	1,004	31.72	40.04	539	17.17	24.32
DRAGON+	1,627	39.59	52.47	726	32.98	40.98	520	15.72	23.21
E5	456	44.09	56.91	486	34.52	42.08	493	17.25	24.72
E5-Mistral	463	46.95	60.29	504	35.96	43.89	508	21.39	28.01
Mistral Reader with Context from Cross-Encoders									
monoT5	784	42.16	55.48	506	35.31	43.31	506	19.90	27.56
BGE	472	47.40	61.09	534	35.03	43.49	525	22.22	30.38
RankLLaMA	484	42.97	55.67	525	37.11	45.37	539	18.45	25.92
Mistral Reader with Context from Chain-of-Thought Model									
IRCoT	505	48.78	62.36	616	38.74	47.29	639	22.84	29.69
Mistral Reader with Context from TRACE									
TRACE-Triple	167	50.68*	65.49*	170	**46.30***	**55.49***	177	**31.86***	**40.05***
TRACE-Doc	395	**53.37***	**67.41***	538	44.90*	54.32*	508	30.66*	38.22*

Table 7: Overall performance (%) of TRACE and baselines on the test sets of three multi-hop QA datasets, where "# Tok" is the average number of tokens in the documents used as context, * indicates p-value <0.05 compared with IRCoT. The best performance per dataset per metric is marked in boldface.

constructed by TRACE can effectively generalise across different readers. This consistency across various readers indicates the robustness of TRACE in producing reasoning chains that are not tailored to a specific model but are broadly applicable.

D.2 Ablation Studies on TRACE-Doc

In the main text, we provide ablation study results for TRACE-Triple to verify the effectiveness of each component in TRACE, i.e., KG generator, reasoning chain constructor, triple ranker and triple selector. We further conduct ablation studies on TRACE-Doc to investigate the impacts of these components on the effectiveness of using reasoning chains to retrieve relevant documents from $\mathcal{D}_q$. These ablation studies are conducted in a manner similar to that of TRACE-Triple, as described in the main text. The experimental results, presented in Table 6, demonstrate consistent findings with the results of TRACE-Triple, highlighting the importance and effectiveness of each component in the overall performance of TRACE.

D.3 Effects of Maximum Chain Length L on MuSiQue

The experimental results regarding the effects of L on the MuSiQue dataset are presented in Figure 9. These results are consistent with those observed on the HotPotQA and 2WikiMultiHopQA datasets, demonstrating the effectiveness of the adaptive chain termination strategy. The results also highlight the importance of setting a proper maximum reasoning chain length L to achieve optimal performance.

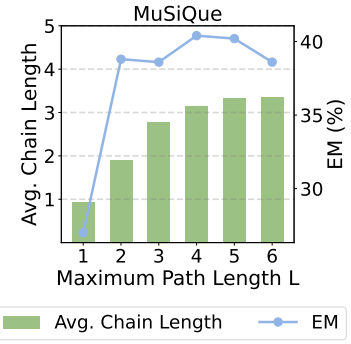

Figure 9: QA performance (%) and average chain length of TRACE-Triple under different values of L on the development set of MuSiQue dataset.

D.4 Effects of Demonstrations in Reasoning Chain Constructor

In the reasoning chain constructor of TRACE, we include demonstrations in the prompt to guide the construction of reasoning chains. We conduct ablation studies to investigate the effects of the demonstrations on the overall performance of TRACE.

Specifically, to examine the effectiveness of the demonstrations, we introduce a variant of TRACE-Triple, namely *w/o demonstrations*, where we remove the demonstrations and only use the instruction to guide the reasoning chain construction. The

Model	HotPotQA			2WikiMultiHopQA			MuSiQue		
	# Tok	EM	F1	# Tok	EM	F1	# Tok	EM	F1
Gemma Reader with Context from Naive Baselines									
w/o documents	43	20.95	29.11	40	22.63	26.60	44	4.05	9.28
w. all documents	1,450	44.32	58.81	1,069	31.48	40.05	2,613	21.10	29.87
Gemma Reader with Context from Bi-Encoders									
Contriever	1,450	45.66	60.08	1,069	33.18	42.58	1,443	23.05	31.76
DRAGON+	1,450	45.79	60.15	446	32.59	40.18	1,411	23.17	31.74
E5	681	45.46	59.32	904	33.23	42.55	705	23.21	31.35
E5-Mistral	688	47.47	61.83	450	35.43	43.72	1,112	25.94	34.59
Gemma Reader with Context from Cross-Encoders									
monoT5	1,450	45.71	60.27	1,069	32.73	41.87	1,136	25.28	33.81
BGE	709	47.24	61.59	683	34.62	43.37	754	26.89	35.54
RankLLaMA	1,450	45.59	59.84	915	32.69	41.45	1,476	25.90	33.91
Gemma Reader with Context from Chain-of-Thought Model									
IRCoT	452	48.47	62.89	551	38.11	47.05	572	27.70	36.27
Gemma Reader with Context from TRACE									
TRACE-Triple	147	51.68*	**67.13***	150	**46.94***	**56.49***	157	**34.17***	**42.49***
TRACE-Doc	353	**51.80***	66.81*	481	43.72*	53.70*	455	33.06*	41.25*

Table 8: Overall performance (%) of TRACE (using Gemma-7B as reader) and baselines on the test sets of three multi-hop QA datasets, where "# Tok" is the average number of tokens in the documents used as context, * indicates p-value <0.05 compared with IRCoT. The best performance per dataset per metric is marked in boldface.

Model	HotPotQA		2WikiMultiHopQA		MuSiQue	
	EM	F1	EM	F1	EM	F1
TRACE-Triple	**67.00**	**76.16**	**66.40**	**72.52**	**40.40**	**46.92**
Effectiveness of Using Demonstrations						
w/o Demonstrations	63.20*	73.57*	64.40*	70.78*	38.20*	44.77*
Effectiveness of Using Adaptive Demonstrations						
Fixed Demonstrations	65.20*	74.49*	66.20	71.85	37.60*	44.37*
Efffects of the Number of Demonstrations						
1 Demonstration	65.00*	74.76*	67.40	73.37	38.00*	44.84*
5 Demonstrations	65.00*	74.40*	68.20*	74.81*	39.20	46.41
10 Demonstrations	64.60*	74.31*	65.01	70.77	37.65*	43.33*

Table 9: Performance (%) of TRACE-Triple under different variants of the reasoning chain constructor on the development sets of three QA datasets, where * denotes p<0.05 compared with TRACE-Triple.

results presented in Table 9 show that removing the demonstrations significantly degrades the performance of TRACE-Triple on all the datasets. This indicates the effectiveness of the demonstrations in enhancing the reasoning chain construction and overall performance of TRACE-Triple.

Moreover, as described in Appendix C.4, we use E5-Mistral to retrieve the top three most similar questions and their corresponding reasoning chains as demonstrations when constructing reasoning chains for a given question. To examine the effectiveness of such an adaptive demonstration selection approach, we introduce a variant of TRACE-Triple, namely *Fixed Demonstrations*, where the same set of demonstrations is used for all the questions. The results, provided in Table 9, indicate that using fixed demonstrations significantly degrades the performance on the HotPotQA and MuSiQue datasets, and slightly degrades the performance on the 2WikiMultiHopQA dataset. These findings demonstrate the effectiveness of the adaptive demonstration selection approach, which is also consistent with previous works (Rubin et al., 2022; Li et al., 2023).

Furthermore, we conduct experiments to investigate the effects of the number of demonstrations on the overall performance of TRACE-Triple. Specifically, we vary the number of demonstrations to $1, 3, 5, 10$ and report the corresponding performance of TRACE-Triple. Note that TRACE-Triple uses 3 demonstrations by default. The results are presented in Table 9, which indicate that increasing the number of demonstrations does not necessarily improve the performance. For example, the performance of TRACE-Triple with 10 demonstrations is generally worse than with fewer demonstrations. This might due to the fact that adding more demonstrations can introduce irrelevant data, which may distract the reasoning chain constructor and lead to incorrect decisions.

D.5 Effects of the Number of Reasoning Chains R

To investigate the effects of the number of reasoning chains R, we vary the value of R from 1 to 20 and report the corresponding performance. Figure 10 shows the performance of both TRACE-Triple and TRACE-Doc on the development sets of HotPotQA and 2WikiMultiHopQA datasets. The

Title: Ellen Glasgow

Text: Ellen Anderson Gholson Glasgow (April 22, 1873 - 2013 November 21, 1945) was an American novelist who portrayed the changing world of the contemporary South.

Knowledge Triples:

<Ellen Glasgow; full name; Ellen Anderson Gholson Glasgow>, <Ellen Glasgow; date of birth; April 22, 1873>, <Ellen Glasgow; date of death; November 21, 1945>, <Ellen Glasgow; nationality; American>, <Ellen Glasgow; occupation; novelist>, <Ellen Glasgow; the theme of her literary work; changing world of the contemporary South>

Title: Heinrich von Bülow (Grotekop)

Text: Heinrich von Bülow also known as Big Top (Grotekop) was a knight born in the middle of the fourteenth century. He died either before 1395 or during 1415. He prospered as a warrior-supporter of Prince Albrecht of Mecklenburg (and of Sweden). Outside Mecklenberg, Heinrich Grotekop is still remembered in many quarters as an archetypal robber baron on account of his appetite for feuding.

Knowledge Triples:

<Heinrich von Bülow (Grotekop); also known as; Big Top (Grotekop)>, <Heinrich von Bülow (Grotekop); born in; middle of the fourteenth century>, <Heinrich von Bülow (Grotekop); died; before 1395 or during 1415>, <Heinrich von Bülow (Grotekop); occupation; warrior-supporter>, <Heinrich von Bülow (Grotekop); supported; Prince Albrecht of Mecklenburg (and of Sweden)>, <Heinrich von Bülow (Grotekop); remembered as; archetypal robber baron>, <Heinrich von Bülow (Grotekop); characterized by; appetite for feuding>

Title: Inaindha Kaigal

Text: Inaindha Kaigal (English: Conjoined Hands) is a 1990 Indian Tamil film, directed by N. K. Vishwanathan. The film features C. Arunpandian, Ramki, Nirosha and Sindhu in lead roles, with Nassar, Senthil, Srividya, Murali Kumar and Prabhakaran playing supporting roles. The film, produced by Aabavanan who also wrote the script and lyrics, had musical score by Gyan Varma and was released on 2 August 1990. The film is a blockbuster in the year 1990 and became a successful venture. The film has been dubbed in Hindi as "Aakhri Sangam" and in Telugu as Sahasa Ghattam.

Knowledge Triples:

<Inaindha Kaigal; English translation; Conjoined Hands>, <Inaindha Kaigal; the year when the film was released; 1990>, <Inaindha Kaigal; genre of the film; Indian Tamil film>, <Inaindha Kaigal; director of the film; N. K. Vishwanathan>, <Inaindha Kaigal; lead actors; C. Arunpandian, Ramki, Nirosha, Sindhu>, <Inaindha Kaigal; supporting actors; Nassar, Senthil, Srividya, Murali Kumar, Prabhakaran>, <Inaindha Kaigal; individual who produced and also wrote the script and lyrics for the film; Aabavanan>, <Inaindha Kaigal; composer of the film's musical score; Gyan Varma>, <Inaindha Kaigal; release date of the film; 2 August 1990>, <Inaindha Kaigal; the status of the film in its release year; blockbuster in 1990 and became a successful venture>, <Inaindha Kaigal; Hindi version name of the film; Aakhri Sangam>, <Inaindha Kaigal; Telugu version name of the film; Sahasa Ghattam>

Table 10: Examples of demonstration data on HotPotQA dataset for the KG generator.

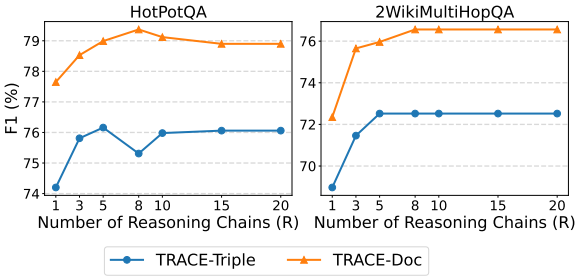

Figure 10: Performance (%) of TRACE with different numbers of reasoning chains on the development sets of the HotPotQA and 2WikiMultiHopQA datasets.

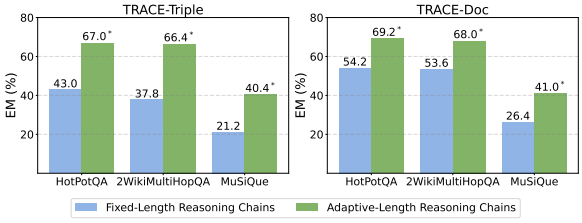

Figure 11: Performance (%) of TRACE under different reasoning chain settings on the development sets of three multi-hop QA datasets, where * indicates p-value<0.05.

results indicate that as R increases, the performance of both TRACE-Triple and TRACE-Doc initially improves and then becomes stable. This trend can be explained by the fact that initially increasing the number of reasoning chains helps to incorporate additional information that is missing in the previous reasoning chains. However, once a certain threshold is reached, no further relevant information can be added, leading to performance stabilisation.

D.6 Effects of Adaptive Chain Termination

In the reasoning chain constructor of TRACE, an adaptive chain termination strategy is employed to automatically determine the optimal lengths of reasoning chains. We conducted ablation studies to investigate the effectiveness of this approach. Specifically, we introduce a variant of TRACE, namely *Fixed-Length Reasoning Chains*, where the lengths of all the reasoning chains are set to a

fixed value L (4 in our experiments). The results are presented in Figure 11, which shows the QA performance of TRACE-Triple and TRACE-Doc on three multi-hop QA datasets. The results indicate that TRACE with adaptive-length reasoning chains (both TRACE-Triple and TRACE-Doc) significantly outperforms its fixed-length counterpart by a large margin. This is because the adaptive chain termination strategy allows the reasoning chain constructor to dynamically determine the required number of triples, thereby avoiding the introduction of unnecessary and redundant triples and leading to improved performance.

D.7 Case Study

The complete results of the case study are provided in Table 14, which shows the top-5 reasoning chains obtained with the reasoning chain constructor and the resulting relevant documents identified using these reasoning chains. These cases yield the following additional findings:

(1) Generating multiple reasoning chains is beneficial as it helps to incorporate relevant information that may be missing in the initial reasoning chains. For example, in the reasoning chains for the question "*What is the birth date of this Spanish footballer, who was added as a holding midfielder in the 2021-13 FC Bayern Munich season?*", the KG triples "⟨*Javi Martínez; position; defensive midfielder or a central defender*⟩" in the second reasoning chain and "⟨*Javi Martínez; nationality; Spanish*⟩" in the third reasoning chain provide complementary information to the first reasoning chain to correctly answer the question. Therefore, using multiple reasoning chains helps to provide a more enriched context, leading to improved performance. This finding is also consistent with the empirical results presented in Appendix D.5.

(2) It is effective to leverage reasoning chains to identify supporting documents. For example, for the multi-hop question "*Are both Blaise Cendrars and Julian Barnes are a citizen of the same country?*", only two documents are identified using the reasoning chains. The first document is about *Blaise Cendrars* and the second one is about *Julian Barnes*, both of which are relevant and highly useful for answering the question. Similar results are also observed in the other two examples. Therefore, these findings indicate that using reasoning chains helps to identify relevant documents while avoiding the introduction of noisy documents, leading to enhanced performance.

D.8 Efficiency Analysis

We conduct experiments to evaluate the latency of TRACE in comparison to the baseline model IRCoT. Specifically, we conducted experiments on a 3.5 GHZ, 32-cores AMD Ryzen Threadripper Process paired with an NVIDIA A6000 GPU. Both TRACE and IRCoT use the same LLaMA3-7B model to generate thoughts or select triples. We report the average runtime, i.e., the average time for indexing one document (**Indexing**) and generating one thought or constructing one reasoning chain (**Inference**), on the development sets. The results are provided in Table 12.

	HotPotQA	2WikiMultiHopQA	MuSiQue
Runtime of IRCoT			
Indexing	2.06e-3	4.17e-3	3.06e-3
Inference	4.27	6.15	6.31
Runtime of TRACE			
Indexing	12.34	14.83	13.17
Inference	4.13	4.49	4.84

Table 12: Average runtime (s) of TRACE during the indexing and inference stages on the development sets.

The results show that compared with IRCoT, TRACE has a higher indexing latency. This is because TRACE requires additional forward propagations through the LLaMA3-7B model to generate KG triples. However, the construction of KG triples can be done offline. Once these triples are generated, they can be directly used to construct reasoning chains, facilitating the subsequent processes. Therefore, the cost of indexing does not impact the real-time efficiency during inference.

Additionally, the results indicate that the inference runtime of TRACE is similar to IRCoT on HotPotQA and is even less on the other two datasets. This similarity arises because both TRACE and IRCoT use iterative methods to generate reasoning chains and chain-of-thoughts, respectively. However, there are differences that impact the runtime: IRCoT requires the LLM to generate thoughts, each containing multiple tokens. This process involves multiple forward propagations through the LLM (i.e., the autoregressive generation), which increase the latency. In contrast, TRACE instructs the model to generate a single token representing the selection of KG triples, which requires only one forward propagation through the LLM, leading to a reduced inference runtime.

question: Which magazine published papers more often; The Wittenburg Door or Sports Collectors Digest?
reasoning chain: <Sports Collectors Digest; type; American advertising weekly paper>, <The Wittenburg Door; publication frequency; bimonthly>
Step-1:
existing knowledge triples:
candidate knowledge triples:
A. no need for additional knowledge triples
B. <Sports Collectors Digest; purpose; provides an avenue through which sellers, traders and avid buyers of sports memorabilia may interact>
C. <The Wittenburg Door; type; Christian satire and humor magazine>
D. <Mike Yaconelli; role in The Wittenburg Door; satirical magazine>
E. <Sports Collectors Digest; type; American advertising weekly paper>
the next possible triple is: E
Step-2:
existing knowledge triples: <Sports Collectors Digest; type; American advertising weekly paper>
candidate knowledge triples:
A. no need for additional knowledge triples
B. <The Wittenburg Door; type; Christian satire and humor magazine>
C. <Mike Yaconelli; role in The Wittenburg Door; satirical magazine>
D. <The Wittenburg Door; publication frequency; bimonthly>
E. <The Wittenburg Door; start year of publication; 1971>
the next possible triple is: D
Step-3:
existing knowledge triples: <Sports Collectors Digest; type; American advertising weekly paper>, <The Wittenburg Door; publication frequency; bimonthly>
candidate knowledge triples:
A. no need for additional knowledge triples
B. <Sports Collectors Digest; purpose; provides an avenue through which sellers, traders and avid buyers of sports memorabilia may interact>
C. <The Wittenburg Door; type; Christian satire and humor magazine>
D. <Mike Yaconelli; role in The Wittenburg Door; satirical magazine>
E. <The Wittenburg Door; reference to; the door of the All Saints' Church in Wittenberg>
the next possible triple is: A

---

question: What is the 5th studio album released by the singer of "A Girl's Gotta Do (What a Girl's Gotta Do)"?
reasoning chain:
Step-1:
existing knowledge triples:
candidate knowledge triples:
A. no need for additional knowledge triples
B. <A Girl's Gotta Do (What a Girl's Gotta Do); artist; American country music artist Mindy McCready>
C. <A Girl's Gotta Do (What a Girl's Gotta Do); release date; February 1997>
D. <Ten Thousand Angels; fourth single; "A Girl's Gotta Do (What a Girl's Gotta Do)">
E. <A Girl's Gotta Do (What a Girl's Gotta Do); songwriters; Robert Byrne and Rick Bowles>
the next possible triple is: B
Step-2:
existing knowledge triples: <A Girl's Gotta Do (What a Girl's Gotta Do); artist; American country music artist Mindy McCready>
candidate knowledge triples:
A. no need for additional knowledge triples
B. <Mindy McCready; number of studio albums; five>
C. <Mindy McCready; fourth album; self-titled>
D. <Mindy McCready; debut album release year; 1996>
E. <Mindy McCready; fifth album; I'm Still Here>
the next possible triple is: E
Step-3:
existing knowledge triples: <A Girl's Gotta Do (What a Girl's Gotta Do); artist; American country music artist Mindy McCready>, <Mindy McCready; fifth album; I'm Still Here>
candidate knowledge triples:
A. no need for additional knowledge triples
B. <Mindy McCready; third album; I'm Not So Tough>
C. <Mindy McCready; debut album release year; 1996>
D. <Mindy McCready; fourth album; self-titled>
E. <Mindy McCready; number of studio albums; five>
the next possible triple is: A

Table 11: Examples of demonstration data on HotPotQA dataset for the triple selector.

Title: Ellen Glasgow

Text: Ellen Anderson Gholson Glasgow (April 22, 1873 – November 21, 1945) was an American novelist who portrayed the changing world of the contemporary South.

Generated Knowledge Triples:

<Ellen Glasgow; full name; Ellen Anderson Gholson Glasgow>, <Ellen Glasgow; date of birth; April 22, 1873>, <Ellen Glasgow; date of death; November 21, 1945>, <Ellen Glasgow; nationality; American>, <Ellen Glasgow; occupation; novelist>, <Ellen Glasgow; the theme of her literary work; changing world of the contemporary South>

Title: Julian Barnes

Text: Julian Patrick Barnes (born 19 January 1946) is an English writer. Barnes won the Man Booker Prize for his book "The Sense of an Ending" (2011), and three of his earlier books had been shortlisted for the Booker Prize: "Flaubert's Parrot" (1984), "England, England" (1998), and "Arthur & George" (2005). He has also written crime fiction under the pseudonym Dan Kavanagh. In addition to novels, Barnes has published collections of essays and short stories.

Generated Knowledge Triples:

<Julian Barnes; nationality; English>, <Julian Barnes; date of birth; 19 January 1946>, <Julian Barnes; occupation; writer>, <Julian Barnes; award won; Man Booker Prize for "The Sense of an Ending" (2011)>, <Julian Barnes; books shortlisted for the Booker Prize; "Flaubert's Parrot" (1984), "England, England" (1998), "Arthur & George" (2005)>, <Julian Barnes; pseudonym; Dan Kavanagh>, <Julian Barnes; genre; crime fiction>, <Julian Barnes; type of writing; novels, essays, short stories>

Title: Emarosa

Text: Emarosa ( ) is an American post-hardcore band from Lexington, Kentucky. The band currently consists of founding members ER White (lead guitar) and Jordan Stewart (keyboards), as well as lead vocalist Bradley Walden and rhythm guitarist Marcellus Wallace.

Generated Knowledge Triples:

<Emarosa; genre; post-hardcore>, <Emarosa; location; Lexington, Kentucky>, <Emarosa; members; ER White (lead guitar), Jordan Stewart (keyboards), Bradley Walden (lead vocalist), Marcellus Wallace (rhythm guitarist)>

Title: Tantalizers

Text: Tantalizers is a leading Nigerian fast food restaurant chain. It opened its first location c. 1997 Festac Town, Lagos. This first location was initially a small neighborhood restaurant serving hamburgers. Success at this first location led to an expansion that has seen the company and its franchisees open additional locations in cities such as Lagos, Ibadan, Abuja, and Port Harcourt. As of 2015, the restaurant has 50 outlets across Nigeria.

Generated Knowledge Triples:

<Tantalizers; type; fast food restaurant chain>, <Tantalizers; location; Festac Town, Lagos>, <Tantalizers; year of opening; c. 1997>, <Tantalizers; initial location; small neighborhood restaurant>, <Tantalizers; initial menu item; hamburgers>, <Tantalizers; expansion; additional locations in cities such as Lagos, Ibadan, Abuja, and Port Harcourt>, <Tantalizers; number of outlets; 50>, <Tantalizers; location of outlets; across Nigeria>

Title: Julius Caesar Chappelle

Text: Julius Caesar Chappelle (1852–1904) was an African-American politician born into slavery in South Carolina. After the American Civil War, he lived for a time with his family in LaVilla, Florida, helping develop the new town. In 1870 he was one of numerous Southern black migrants to Boston, Massachusetts, which had a thriving black community and strong abolitionist history. He later joined the Republican Party that was founded by abolitionists, and Chappelle was elected to two terms in the Massachusetts state legislature, serving 1883-1886. Julius Caesar Chappelle was also the first African-American to serve on the Massachusetts State Senate Committee where he served three terms. Chappelle was active in supporting civil rights, trying to reduce discrimination, and consumer affairs. His speeches were frequently covered by newspapers. Throughout his life and political career, he held secondary supervisory government positions in maintenance, such as at the United States Post Office and US Boston Custom House. Although Julius Caesar Chappelle may have graced the same pages in newspapers as Frederick Douglass, Chappelle is not as well-known because he is not known to have left much of a literary footprint such as writing manuscripts or for pamphlets.

Generated Knowledge Triples:

<Julius Caesar Chappelle; date of birth; 1852>, <Julius Caesar Chappelle; date of death; 1904>, <Julius Caesar Chappelle; place of birth; South Carolina>, <Julius Caesar Chappelle; nationality; African-American>, <Julius Caesar Chappelle; occupation; politician>, <Julius Caesar Chappelle; place of residence; LaVilla, Florida>, <Julius Caesar Chappelle; place of residence; Boston, Massachusetts>, <Julius Caesar Chappelle; political party; Republican Party>, <Julius Caesar Chappelle; served in; Massachusetts state legislature, 1883-1886>, <Julius Caesar Chappelle; served in; Massachusetts State Senate Committee, three terms>, <Julius Caesar Chappelle; role in civil rights; active in supporting civil rights, trying to reduce discrimination>, <Julius Caesar Chappelle; role in consumer affairs; active in consumer affairs>, <Julius Caesar Chappelle; occupation in government; held secondary supervisory government positions in maintenance, such as at the United States Post Office and US Boston Custom House>, <Julius Caesar Chappelle; notable achievement; first African-American to serve on the Massachusetts State Senate Committee>, <Julius Caesar Chappelle; notable achievement; served two terms in the Massachusetts state legislature>, <Julius Caesar Chappelle; notable achievement; served three terms in the Massachusetts State Senate Committee>, <Julius Caesar Chappelle; notable achievement; his speeches were frequently covered by newspapers>

Table 13: Examples of generated KGs for documents on the HotPotQA dataset.

Question: Are both Blaise Cendrars and Julian Barnes are a citizen of the same country?

Reasoning Chains:

1: <Blaise Cendrars; nationality; Swiss>, <Julian Barnes; nationality; English>
2: <Blaise Cendrars; nationality; French>, <Julian Barnes; nationality; English>
3: <Julian Barnes; nationality; English>, <Blaise Cendrars; nationality; French>, <Blaise Cendrars; nationality; Swiss>, <Blaise Cendrars; event; became a naturalized French citizen in 1916>
4: <Julian Barnes; nationality; English>, <Blaise Cendrars; nationality; Swiss>
5: <Blaise Cendrars; nationality; French>, <Julian Barnes; nationality; English>, <Blaise Cendrars; event; became a naturalized French citizen in 1916>

Relevant Documents Identified Using Reasoning Chains:

1. Title: Blaise Cendrars

Frédéric-Louis Sauser (1 September 1887 – 21 January 1961), better known as Blaise Cendrars, was a Swiss-born novelist and poet who became a naturalized French citizen in 1916. He was a writer of considerable influence in the European modernist movement.

2. Title: Julian Barnes

Julian Patrick Barnes (born 19 January 1946) is an English writer. Barnes won the Man Booker Prize for his book "The Sense of an Ending" (2011), and three of his earlier books had been shortlisted for the Booker Prize: "Flaubert's Parrot" (1984), "England, England" (1998), and "Arthur & George" (2005). He has also written crime fiction under the pseudonym Dan Kavanagh. In addition to novels, Barnes has published collections of essays and short stories.

Generated Answer: no

Question: What was the occupation of both Christina Stead and Nuruddin Farah?

Reasoning Chains:

1: <Christina Stead; occupation; novelist and short-story writer>, <Nuruddin Farah; occupation; novelist>
2: <Christina Stead; occupation; novelist and short-story writer>, <Nuruddin Farah; written works; plays, short stories, essays>
3: <Nuruddin Farah; occupation; novelist>, <Christina Stead; occupation; novelist and short-story writer>
4: <Nuruddin Farah; birthdate; 24 November 1945>, <Christina Stead; occupation; novelist and short-story writer>
5: <Nuruddin Farah; written works; plays, short stories, essays>, <Christina Stead; occupation; novelist and short-story writer>

Relevant Documents Identified Using Reasoning Chains:

1. Title: Christina Stead

Christina Stead (17 July 190231 March 1983) was an Australian novelist and short-story writer acclaimed for her satirical wit and penetrating psychological characterisations. Christina Stead was a committed Marxist, although she was never a member of the Communist Party. She spent much of her life outside Australia.

2. Title: Nuruddin Farah

Nuruddin Farah (Somali: "Nuuradiin Faarax") (born 24 November 1945) is a Somali novelist. He has also written plays both for stage and radio, as well as short stories and essays. Since leaving Somalia in the 1970s he has lived and taught in numerous countries, including the United States, England, Germany, Italy, Sweden, Sudan, India, Uganda, Nigeria and South Africa.

Generated Answer: novelist

Question: What is the birth date of this Spanish footballer, who was added as a holding midfielder in the 2012-13 FC Bayern Munich season?

Reasoning Chains:

1: <2012–13 FC Bayern Munich season; new player signed after the first week of the Bundesliga season; Javi Martínez>, <Javi Martínez; date of birth; 2 September 1988>
2: <Javi Martínez; position; defensive midfielder or a central defender>, <Javi Martínez; date of birth; 2 September 1988>
3: <Javi Martínez; nationality; Spanish>, <Javi Martínez; date of birth; 2 September 1988>
4: <Javi Martínez; date of birth; 2 September 1988>, <Javi Martínez; club; FC Bayern Munich>
5: <Javi Martínez; club; FC Bayern Munich>, <Javi Martínez; date of birth; 2 September 1988>

Relevant Documents Identified Using Reasoning Chains:

1. Title: Javi Martínez

Javier "Javi" Martínez Aginaga (born 2 September 1988) is a Spanish footballer who plays for German club FC Bayern Munich as a defensive midfielder or a central defender.

2. Title: 2012–13 FC Bayern Munich season

The 2012–13 FC Bayern Munich season was the 114th season in the club's history and the 48th consecutive season in the top flight of German football, the Bundesliga, since the promotion of the team from the Regionalliga Süd in 1965. Before the start of the season, Bayern signed Xherdan Shaqiri, Dante, Claudio Pizarro, Mitchell Weiser, Tom Starke and Mario Mandžukić. Bayern also added holding midfielder Javi Martínez after the first week of the Bundesliga season at the transfer deadline. The club started the season with a nine-match winning streak. The club would end the season claiming the Treble, winning the Bundesliga, the UEFA Champions League and the DFB-Pokal.

Generated Answer: 2 September 1988

Table 14: Case study of TRACE on the HotPotQA dataset, where the information relevant to the questions in the reasoning chains and the documents is marked in blue and orange, respectively.