

Article

CRP-RAG: A Retrieval-Augmented Generation Framework for Supporting Complex Logical Reasoning and Knowledge Planning

Kehan Xu ^{2,†} , Kun Zhang ^{3,†}, Jingyuan Li ^{1,*}, Wei Huang ² and Yuanzhuo Wang ^{4,*}¹ School of Computer and Artificial Intelligence, Beijing Technology and Business University, Beijing 102401, China² School of Information Science and Engineering, Yanshan University, Qinhuangdao 066104, China; 202222040185@stumail.ysu.edu.cn (K.X.);³ Tencent WeChat AI—Pattern Recognition Center Tencent Inc., Beijing 100085, China⁴ Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100005, China

* Correspondence: li.jingyuan.jerry@btbu.edu.cn (J.L.); wangyuanzhuo@ict.ac.cn (Y.W.)

† These authors contributed equally to this work.

Abstract: The Retrieval-Augmented Generation (RAG) framework enhances Large Language Models (LLMs) by retrieving relevant knowledge to broaden their knowledge boundaries and mitigate factual hallucinations stemming from knowledge gaps. However, the RAG Framework faces challenges in effective knowledge retrieval and utilization; invalid or misused knowledge will interfere with LLM generation, reducing reasoning efficiency and answer quality. Existing RAG methods address these issues by decomposing and expanding queries, introducing special knowledge structures, and using reasoning process evaluation and feedback. However, the linear reasoning structures limit complex thought transformations and reasoning based on intricate queries. Additionally, knowledge retrieval and utilization are decoupled from reasoning and answer generation, hindering effective knowledge support during answer generation. To address these limitations, we propose the CRP-RAG framework, which employs reasoning graphs to model complex query reasoning processes more comprehensively and accurately. CRP-RAG guides knowledge retrieval, aggregation, and evaluation through reasoning graphs, dynamically adjusting the reasoning path based on evaluation results and selecting knowledge-sufficiency paths for answer generation. CRP-RAG outperforms the best LLM and RAG baselines by 2.46 in open-domain QA, 7.43 in multi-hop reasoning, and 4.2 in factual verification. Experiments also show the superior factual consistency and robustness of CRP-RAG over existing RAG methods. Extensive analyses confirm its accurate and fact-faithful reasoning and answer generation for complex queries.

Keywords: retrieval-augmented generation (RAG); large language models (LLMs); reasoning graph; complex thought transformation; knowledge utilization



Academic Editor: Fabio Grandi

Received: 20 November 2024

Revised: 12 December 2024

Accepted: 23 December 2024

Published: 26 December 2024

Citation: Xu, K.; Zhang, K.; Li, J.; Huang, W.; Wang, Y. CRP-RAG: A Retrieval-Augmented Generation Framework for Supporting Complex Logical Reasoning and Knowledge Planning. *Electronics* **2025**, *14*, 47. <https://doi.org/10.3390/electronics14010047>

Correction Statement: This article has been republished with a minor change. The change does not affect the scientific content of the article and further details are available within the backmatter of the website version of this article.

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Large Language Models (LLMs) [1–3] have demonstrated remarkable performance in knowledge reasoning and exhibited proficiency across various task domains [4,5]. However, due to the uneven distribution of knowledge in the training data of LLMs, the parameterized knowledge stored within them constitutes only a subset of the world's knowledge [6], indicating the existence of knowledge boundaries for LLMs. When LLMs attempt to answer questions beyond their knowledge boundaries, they may suffer from factual hallucinations

due to the lack of corresponding knowledge, leading to the generation of content inconsistent with facts and compromising the accuracy of their answers. The Retrieval-Augmented Generation (RAG) framework [7,8] addresses this by retrieving external knowledge bases composed of external information, extracting non-parameterized knowledge, and incorporating it into model prompts, thereby embedding new knowledge into LLMs to expand their knowledge boundaries [9].

Despite its significant success in open-domain question answering tasks, the RAG framework still faces two major challenges. **Challenge 1: The reasoning process of the RAG framework is disturbed by irrelevant knowledge.** The retrieval results of the RAG framework often contain documents related to the query but irrelevant to reasoning [10]. As illustrated in Figure 1, a paragraph describing the color of horses owned by Joséphine de Beauharnais (Napoleon’s first wife) cannot serve as evidence to answer “What color was Napoleon’s horse?”, yet it may still be retrieved. This irrelevant knowledge will interfere with the reasoning process of the RAG framework, ultimately leading to hallucination phenomena during answer generation. **Challenge 2: The RAG framework lacks the ability for knowledge planning.** The RAG framework lacks the ability to understand relationships between knowledge and plan strategies for their utilization, resulting in the misuse of relevant knowledge during reasoning. Incorrect timelines and scopes of knowledge usage will lead to erroneous generation by the RAG framework. To address Challenge 1, current research advocates decomposing the reasoning process of complex queries [11,12] into several independent linear reasoning paths [13,14] to explain retrieval intent, aiming to enhance the reasoning relevance of the RAG framework’s retrieval results. To tackle Challenge 2, current research introduces special knowledge structures to pre-construct relationships between knowledge during the knowledge base construction phase [15,16]. On the other hand, some studies propose designing and integrating new RAG workflows, dynamically planning and formulating knowledge and reasoning strategies through dynamic programming and evaluating system states [17,18]. However, current RAG research still faces two issues. **Issue 1: Existing RAG research does not support modeling the reasoning process and thought transformations for complex queries.** The linear reasoning structures employed in current RAG research often do not support complex thought transformations involving multiple linear reasoning paths, such as switching between paths, backtracking, and other complex reasoning behaviors [19]. This results in RAG methods based on linear reasoning structures being unable to adjust reasoning strategies when knowledge is insufficient, leading to factual hallucinations during the generation process. **Issue 2: There is a gap between the knowledge acquisition and usage process and the reasoning process in existing RAG research.** Existing RAG research relies on static criteria unrelated to reasoning, such as pre-constructed knowledge structures and independent strategy evaluation modules, for knowledge retrieval and planning. However, the knowledge retrieval and planning within the reasoning process changes dynamically with the reasoning strategy. Therefore, static knowledge retrieval and planning often lead to these processes being isolated from the reasoning process, ultimately making it difficult for relevant knowledge to effectively support the reasoning process of the RAG framework.

Addressing these key issues, our work’s aims are as follows: (1) design a complex reasoning structure for a RAG method to model complex queries; (2) guide the knowledge retrieval and utilization of the RAG method based on this structure; and (3) evaluate knowledge effectiveness within the reasoning process, dynamically adjusting the reasoning strategy of the RAG method to select the reasoning paths supported by valid knowledge. We propose CRP-RAG, a RAG framework supporting complex reasoning and knowledge planning. CRP-RAG models complex query reasoning through a reasoning graph, guiding knowledge retrieval, utilization planning, and strategy adjustment. It consists of the

following three modules: Reasoning Graph Construction (GC), Knowledge Retrieval and Aggregation (KRA), and Answer Generation (AG). The GC module constructs a reasoning graph for the comprehensive and flexible representation of reasoning path relationships. The KRA module builds complex connections among knowledge based on the reasoning graph structure, conducting knowledge retrieval and aggregation at the level of reasoning graph nodes to ensure relevance between knowledge utilization and the reasoning process. The AG module evaluates knowledge effectiveness and selects valid reasoning paths for LLM reasoning and answer generation. Compared to existing RAG methods that rely on knowledge structure construction, sub-query decomposition, and self-planning, CRP-RAG expands and optimizes the solution space for complex queries by constructing nonlinear reasoning structures, aiming to enhance the reasoning capability of the RAG framework in response to such queries. Furthermore, CRP-RAG guides knowledge retrieval, evaluation, and reasoning strategy formulation based on reasoning logic, aligning relevant knowledge with the reasoning process to ensure the effective support of the reasoning process. Compared to the best-performing state-of-the-art LLM and RAG baselines, CRP-RAG achieves an average performance gain of 2.46 in open-domain question answering tasks—7.43 in multi-hop reasoning tasks, and 4.2 in factual verification tasks. Furthermore, experiments on factual consistency and robustness demonstrate that CRP-RAG exhibits superior factual consistency and robustness compared to the existing RAG methods. Extensive analyses show that CRP-RAG can perform accurate and fact-faithful reasoning and answer generation based on complex queries.

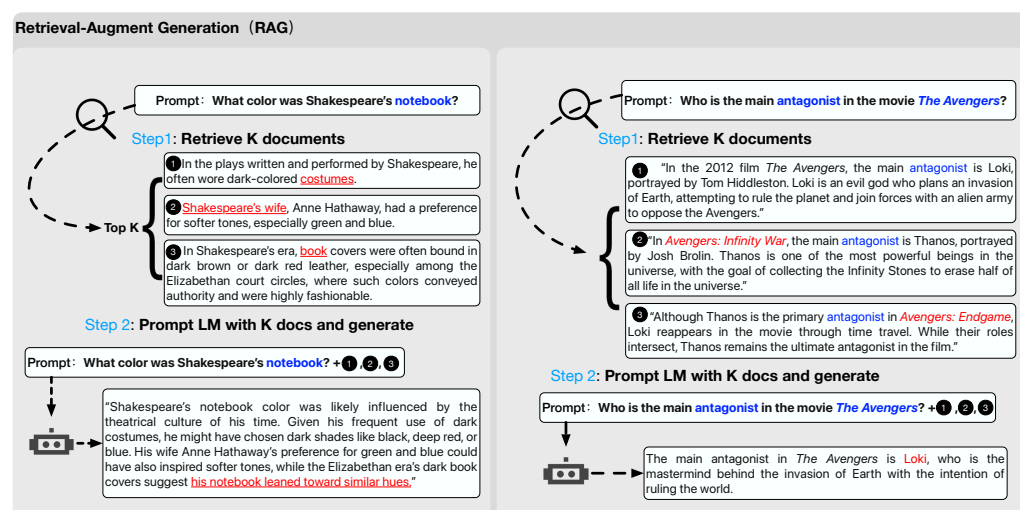


Figure 1. Two challenges faced by the RAG Framework: (i) **Left:** The inference process is disturbed by irrelevant knowledge in the retrieved results. (ii) **Right:** The complex associations among the knowledge in the retrieved results cannot be analyzed and understood.

We summarize our contributions as follows:

- (1) We are the first to propose a comprehensive modeling of the RAG reasoning process based on a reasoning graph and introduce the reasoning graph into the RAG approach to support more complex thought transformations. This provides a novel perspective for improving RAG in the context of complex queries.
- (2) We introduce a reasoning graph-based approach to guide knowledge retrieval, knowledge utilization, and answer generation, enhancing the rationality of reasoning and knowledge strategy formulation within the RAG framework during complex reasoning processes.
- (3) We conduct extensive experiments to demonstrate the effectiveness of our method. The experimental results showcase the excellent performance of the reasoning-guided

RAG framework in complex reasoning and question-answering tasks, further proving the significant role of reasoning-guided knowledge planning in improving the reasoning capabilities of RAG for complex queries.

This paper will sequentially introduce the related work pertinent to this study (Section 2), the detailed methodology design of CRP-RAG (Section 3), the primary experimental details and results (Section 4), the experiments and discussions regarding other aspects of CRP-RAG (Section 5), and the conclusions drawn from our research (Section 6).

2. Related Work

2.1. Retrieval-Augmented Generation Based on the Knowledge Structure

The retrieval results of the RAG framework often consist of multiple unstructured knowledge documents, which are concatenated within prompt templates to assist the reasoning and generation processes of LLMs. Consequently, the knowledge associations among these unstructured documents necessitate additional reasoning by LLMs during the generation process. LLMs are prone to errors when understanding implicitly expressed knowledge associations, leading to the misuse of knowledge and, ultimately, erroneous decisions in the generation process. Therefore, some studies advocate for modeling the associations between knowledge through predefined knowledge structures, thereby forming a structured knowledge system. This system is then utilized as a prompt to guide LLMs in deeply understanding the interconnections among knowledge during the reasoning process and planning optimal knowledge utilization strategies within the framework of the knowledge system.

In terms of knowledge structure selection, existing research often employs knowledge structures such as text templates, knowledge trees, and knowledge graphs to model the associations between knowledge. Text templates distill and summarize knowledge collections through the textual reconstruction of knowledge, during which LLMs perform knowledge distillation, summarization, and structuring according to instructions, explicitly expressing and expanding important knowledge association information through natural language [20,21]. Fine-tuning models based on text-template instructions enhance their understanding of users' knowledge preferences [22]. Knowledge trees model the parent-child and hierarchical relationships between knowledge, improving efficiency in retrieval and knowledge utilization [16,23]. On the other hand, knowledge graphs model the entity associations between knowledge to assist LLMs in understanding the detailed associations of relevant knowledge within retrieval results [24–28]. Additionally, some studies design specific knowledge structures tailored to specific generation task goals to improve the performance of RAG in those tasks. CANDLE [29] extends existing knowledge bases by constructing concepts and instances of existing knowledge and establishing associations between abstract and instance knowledge. Thread [15] constructs associations between existing knowledge across different action decisions for action decision-making problems. Buffer of Thoughts [30] and ARM-RAG [31] extract general principles from knowledge and model the logical relationships between knowledge and experience.

Due to the computational cost associated with dynamic knowledge modeling, most existing research tends to separate knowledge modeling from the reasoning process, performing the static modeling of knowledge during the knowledge base construction phase. However, some studies argue that the interaction between dynamic knowledge modeling and the knowledge retrieval process can further enhance model generation performance and improve the flexibility of knowledge utilization in the RAG method. They advocate for knowledge modeling after obtaining retrieval results. RECOMP [20] and BIDER [21] propose knowledge distillation based on existing retrieval results, obtaining

more precise and abundant relevant knowledge and its associated information through knowledge aggregation.

However, the knowledge structures employed and designed using existing knowledge structure modeling methods are independent of the answer generation and reasoning processes of RAG, which leads to the omission of logical relationships among knowledge during the reasoning process in the modeling stage. This triggers the improper use of knowledge by LLMs.

2.2. Retrieval-Augmented Generation Based on Query Decomposition

The queries input into the RAG framework often exhibit ambiguity in expression and complexity in knowledge requirements. These complex knowledge needs and expressions are not represented at the semantic level, making it difficult for the retrieval process to understand them. When faced with complex queries, query decomposition methods typically perform reasoning and logical analysis in natural language to obtain an explicit representation of the user's knowledge needs, thereby guiding and expanding the content of the retrieval results.

Existing research on query decomposition in RAG methods includes reconstructive decomposition and expansive decomposition. Reconstructive decomposition focuses on ambiguous expressions and logical information in user queries, guiding LLMs to reformulate queries based on prompts [12,32]. LLMs deconstruct and analyze the knowledge needs of queries based on their own parametric knowledge and reasoning abilities. Compared to expansive decomposition, reconstructive decomposition demonstrates stronger preference alignment and self-correction capabilities. During the decomposition process, LLMs can achieve self-evolution and correction based on their own feedback [33–35] or refine reconstructive results through iterative query reformulation [36]. On the other hand, expansive decomposition decomposes queries into several sub-queries to expand the retrieval solution space based on specific decomposition rules [37,38] or structures [39,40]. By defining specific decomposition rules, processes, and structures, expansive decomposition can better ensure the thematic consistency of sub-queries and exhibit greater robustness.

However, existing query decomposition methods often rely on LLMs to perform decomposition and reconstruction based on prompts. The implicit reasoning of LLMs may pose two issues as follows: (1) The decomposition and reconstruction of queries by LLMs are independent of the reasoning process, lacking explicit reasoning constraints, which can easily lead to errors during the decomposition of user queries. (2) The inexplicability of LLMs' implicit reasoning results in the potential for topic drift in the reconstructed results of existing query decomposition and reconstruction methods, which will affect the effectiveness of knowledge retrieval and use.

2.3. Thinking and Planning in Retrieval-Augmented Generation

The RAG framework expands the knowledge boundaries of LLMs. However, due to their “knowledge retrieval–answer generation” workflow, RAG must perform knowledge retrieval for each query, leading to the neglect of LLMs' intrinsic parametric knowledge and potential adverse effects from irrelevant knowledge [18]. Therefore, planning for knowledge retrieval and utilization and assessing and perceiving own knowledge boundaries can enhance the efficiency of knowledge retrieval and utilization in RAG. The RAG frameworks based on planning and self-reflection extend the workflow of RAG into a nonlinear evaluation and decision-making process. By calculating and assessing metrics such as the knowledge adequacy of RAG and the factual consistency of generated answers during knowledge retrieval and answer generation, and making subsequent behavioral

decisions based on the evaluation results, these methods dynamically adjust the workflow of RAG, thereby improving their efficiency.

The current self-planning and reflective RAG framework primarily aims to plan and select retrieval occasions, as well as plan and correct generated content. The planning and selection of retrieval timing involve assessing metrics such as the adequacy of model parametric knowledge [17,18,41,42] and the effectiveness of retrieval results [43,44], thereby evaluating the value of knowledge retrieval and planning the timing and scope of retrieval. On the other hand, planning and correcting generated content involves assessing the quality of answers based on metrics such as the factual consistency [18,44] and accuracy [45,46] of the generated content. Based on these evaluations, the framework determines whether the generated content requires correction and employs iterative retrieval, answer expansion, and decomposition to expand and correct the answer content.

Current RAG planning and self-reflection methods primarily focus on evaluating the effectiveness of the knowledge retrieval process and retrieval results, thereby adjusting the generation strategy. Based on the idea of self-reflection in RAG frameworks, we believe that the knowledge-based reasoning process of LLMs should also be evaluated. By incorporating process evaluation results, RAG frameworks should gain the ability to dynamically adjust their reasoning strategies, ensuring the rationality of path decisions during the reasoning process.

2.4. Reasoning Structure of LLMs

LLMs possess powerful reasoning abilities, but their reasoning processes during answer generation are often uninterpretable. Therefore, explaining and enhancing LLMs' reasoning capabilities pose significant challenges for improving their performance and practical applications. Based on LLMs' instruction-following abilities, prompt engineering for reasoning enhancement has found that specific reasoning-enhanced prompts [47] can significantly improve the interpretability and accuracy of LLMs' reasoning. Following these findings, some studies propose guiding LLMs to perform explicit instruction-based reasoning through prompts, achieving remarkable experimental results. However, reasoning rules in reasoning prompts often fail to fully guide LLMs in modeling the complete reasoning process. Hence, current research advocates for guiding LLMs to achieve more complete and accurate reasoning modeling through the design of reasoning structures. Unlike the linear reasoning structure represented by Chain of Thought (CoT) [13,48], CoT-SC [49] combines linear reasoning structures into a set of linear reasoning chains through the extensive sampling of reasoning steps, thereby expanding LLMs' reasoning path selection space, enhancing the representation ability of reasoning structures, and broadening the range of reasoning operations that LLMs can choose. Meanwhile, Tree of Thought (ToT) [14] constructs the reasoning process as a tree, combining linear reasoning paths into a traceable multi-linear reasoning structure, further improving the representation ability of reasoning structures and expanding the reasoning operations available to LLMs. Graph of Thought (GoT) [19] defines and simulates reasoning graph structures, using nonlinear reasoning structures to support complex reasoning operations such as collaborative path reasoning among LLMs and reasoning backtrace among different paths. Inspired by GoT, this study designs a reasoning graph construction method suitable for the RAG method, avoiding the possibility of circular reasoning in GoT and further improving the efficiency of LLMs in complex reasoning. We believe that reasoning graphs can represent complex reasoning processes comprehensively and flexibly. Therefore, we use reasoning graphs to guide the reasoning path selection, knowledge retrieval, and utilization planning in the RAG method.

3. Method

In this section, we introduce the framework design and reasoning process of CRP-RAG (Section 3.1), along with the structures and workflows of its three primary modules, the Reasoning Graph Construction (GC) Module (Section 3.2), the Knowledge Retrieval and Aggregation (KRA) Module (Section 3.3), and the Answer Generation (AG) Module (Section 3.4). The overall architecture of CRP-RAG is illustrated in Figure 2.

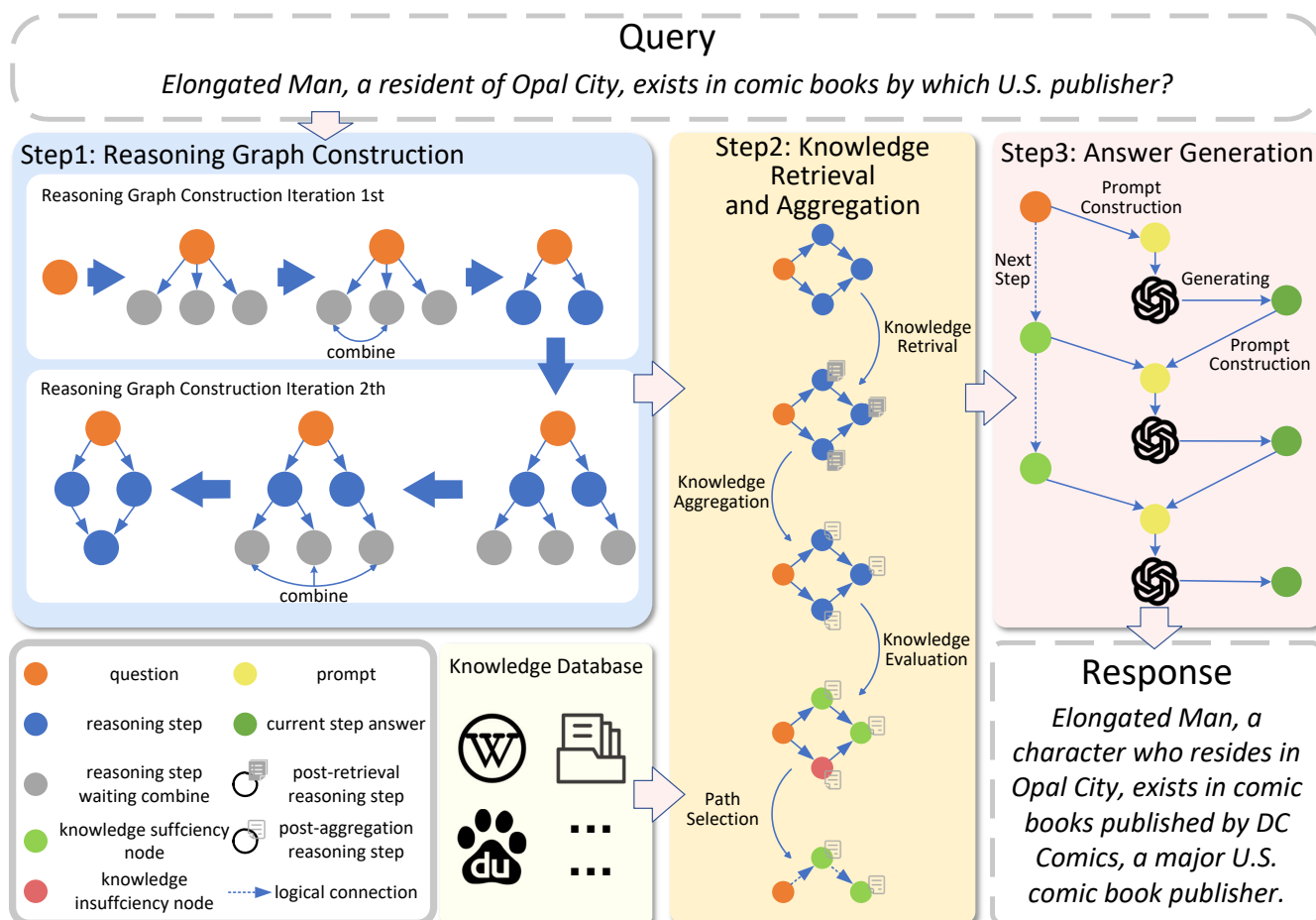


Figure 2. Overview of CRP-RAG framework. The CRP-RAG consists of three modules: (i) The GC module constructs the reasoning graph based on the query. (ii) The KRA module performs knowledge retrieval and aggregation based on the nodes of the reasoning graph. (iii) The AG module generates a query-based answer leveraging the reasoning graph and the relevant knowledge.

3.1. Preliminary

For a given query q , the CRP-RAG framework initially models the reasoning process by iteratively constructing a reasoning graph G . The formulation for the reasoning graph construction process is defined by Equation (1).

$$G = GC(q) \quad (1)$$

Given that both the question and the answer in a question-answering task should be unequivocally defined, the reasoning process in such tasks should not involve circular reasoning. Therefore, $G = \{V, E\}$ is a directed acyclic reasoning graph, where V represents the set of nodes in the reasoning graph, with $v_i \in V$ denoting the node that represents a specific reasoning step in the process, expressed in natural language. E represents the set of edges in the reasoning graph, with $e_i \in E$ indicating the sequential relationship between reasoning steps. The knowledge retrieval and aggregation module operates on each node

in V , retrieving and aggregating knowledge for all nodes to form an aggregated knowledge set K . The formulation for the knowledge retrieval and aggregation process is defined by Equation (2).

$$K = KRA(V, q) \quad (2)$$

Notably, $k_i \in K$ represents the relevant knowledge obtained after knowledge retrieval and aggregation for the corresponding reasoning graph node v_i , and it serves as the context to support the reasoning step associated with the node. The answer generation module evaluates the adequacy of knowledge for all nodes in V and, based on the evaluation results, selects knowledge-sufficiency reasoning paths to guide LLMs in completing the reasoning and answer generation, yielding the answer a . The formulation for this process is defined in Equation (3).

$$a = AG(K, V, q) \quad (3)$$

3.2. Reasoning Graph Construction

Given a specific query q , the reasoning graph construction module iteratively explores all reasoning possibilities, storing all potential reasoning steps as graph nodes and merging similar reasoning steps to construct the reasoning graph G . G is a refined representation of all reasoning possibilities, guiding knowledge retrieval, utilization, and reasoning path selection. Specifically, the reasoning graph construction module starts iteration with the user query q and, based on the reasoning graph $G'_n = \{V'_n, E'_n\}$ at the end of the n -th iteration, each iteration of the module consists of the following two steps: new node generation and node merging.

New Node Generation: The new node generation step involves creating several new nodes for each sink node in V'_n of the reasoning graph. These new nodes represent the next specific reasoning steps when the existing reasoning processes are taken as known conditions. The formula for generating new nodes for a particular sink node v_i is expressed by Equation (4).

$$V_n^{new}(i) = LLM(prompt_{gen}, v_i, q) \quad (4)$$

$LLM()$ leverages LLMs to generate text content based on input instruction information. $V_n^{new}(i)$ denotes the set of new nodes generated based on v_i , and $prompt_{gen}$ represents the prompt templates for generating new nodes as detailed in Appendix A. To ensure that the reasoning graph explores all possible reasoning paths as comprehensively as possible, we refrain from using greedy decoding during the LLMs' generation process and instead employ sampling to enhance the diversity of the content generated by the LLMs. After generating new nodes for all sink nodes, the system obtains several sets of new nodes, $V_n^{new} = [V_n^{new}(1), V_n^{new}(2), \dots, V_n^{new}(k)]$, where the length of V_n^{new} is consistent with the number of sink nodes in V'_n . Each element in the V_n^{new} is a set of new nodes generated based on the corresponding sink node.

Node Merging: Due to the potential presence of reasoning step nodes with similar topics among all newly generated nodes, the system merges similar nodes to reduce redundant information in G and updates their connectivity status with the corresponding sink nodes. Specifically, the system performs node merging for each new node in all sets of V_n^{new} iteratively, resulting in $V_n^{new} = [v_n^{new}(1), v_n^{new}(2), \dots, v_n^{new}(m)]$, where m is the total number of nodes in all new node sets, and $v_n^{new}(i) \in V_n^{new}$ represents a new node generated based on a certain sink node. For a new node $v_n^{new}(i)$, the node merging process involves calculating the similarity between it and all nodes in V_n^{new} one by one to determine whether

nodes need to be merged (Equation (5)) and performing the merge operation if necessary (Equation (6)).

$$sim = enc(v_n^{new}(i)) \odot enc(v_n^{new}(j)) \quad (5)$$

$$v_n^{combine}(i) = LLM(prompt_{combine}, v_n^{new}(i), v_n^{new}(j), q) \text{ if } sim > threshold_{combine} \quad (6)$$

The $enc()$ function semantically encodes the new nodes based on a language model. The sim represents the semantic similarity score between nodes, which is a real number ranging from 0 to 1 and is calculated through the inner product of their encodings. Moreover, $v_n^{combine}(i)$ is the merged node resulting from the combination of $v_n^{new}(i)$ and $v_n^{new}(j)$, which replaces the original nodes in V_n^{new} and inherits their incoming relationships. $prompt_{combine}$ is the instruction template for node merging detailed in Appendix A, and $threshold_{combine}$ is a hyperparameter representing the similarity threshold that sets the lower limit for the semantic similarity required for node merging. After node merging, the system obtains the merged new node set $V_n^{combine}$ and its relationships E_n^{new} with the corresponding sink nodes, constructing the subgraph $G'_{n+1} = \{V'_n \cup V_n^{combine}, E'_n \cup E_n^{new}\}$ at the end of the $(n + 1) - th$ iteration.

Iteration Ending Condition: The iteration terminates when all sink nodes in the subgraph G'_i formed after the i -th iteration correspond to the final reasoning step. At this point, the constructed reasoning graph G is identical to the reasoning subgraph G'_i .

3.3. Knowledge Retrieval and Aggregation

The Knowledge Retrieval and Aggregation process performs knowledge retrieval and aggregation for each node in V , forming an aggregated knowledge set K . The length of the set K is consistent with the length of the set V . Each $k_i \in K$ represents the relevant knowledge obtained through knowledge retrieval and aggregation for the corresponding node v_i , serving as reasoning context to assist LLMs in performing reasoning under the topic of v_i . For any node $v_i \in V$, KRA acquires its relevant knowledge k_i through the following two steps: knowledge retrieval and knowledge aggregation.

Knowledge Retrieval: KRA initially performs knowledge retrieval based on each node in V , obtaining a retrieval result set $D = [d_1, d_2, \dots, d_n]$. Each $d_i \in D$ represents the retrieval result set for the corresponding node v_i , consisting of several related documents. For any node $v_i \in V$, the formulation of knowledge retrieval is expressed as shown in Equation (7).

$$d_i = top_k(similarity(v_i, r_j)) \mid i \in 0, 1, 2, \dots, |V| - 1, j \in 0, 1, 2, \dots, |R| - 1 \quad (7)$$

Here, $similarity()$ is the semantic similarity calculation function defined by Equation (5). The function $top_k()$ returns the top k knowledge base documents with the highest similarity scores. R represents the external knowledge base being searched, and $r_j \in R$ denotes a document within the knowledge base.

Knowledge Aggregation: To further extract key knowledge from d_i and refine knowledge representation, the system performs knowledge refinement and aggregation on all retrieval result sets in D , forming an aggregated knowledge set $K = [k_1, k_2, \dots, k_n]$. Each $k_i \in K$ is obtained from d_i through knowledge aggregation. Knowledge refinement and aggregation are achieved by LLMs that generate knowledge summaries for the relevant documents in d_i . The formulation for this process is shown in Equation (8).

$$k_i = LLM(prompt_{integration}, d_i, v_i) \quad (8)$$

$prompt_{integration}$ refers to the prompt template for knowledge aggregation provided in Appendix A.

3.4. Answer Generation

Based on the reasoning graph G and the reasoning graph knowledge set K , the Answer Generation module first evaluates the knowledge sufficiency of each node v_i in V . Based on the evaluation results, it selects a set of reasoning paths $C = [c_1, c_2, \dots, c_p]$ composed of knowledge-sufficient nodes for reasoning and answer generation. Each $c_i = [s_1, s_2, \dots, s_p]$ represents a knowledge-sufficient reasoning path, where s_1 is a source node in G , s_p is a sink node in G , and $s_i \in c_i$ represents a reasoning step within the path. Specifically, the AG module consists of the following two steps: knowledge sufficiency evaluation, as well as reasoning path selection and answer generation.

Knowledge Sufficiency Evaluation: The AG first calculates the textual perplexity of each node v_i in V when LLMs perform reasoning based on the corresponding knowledge k_i . This aims to quantify the sufficiency of the knowledge provided by k_i during the reasoning process based on v_i . If the textual perplexity is too high, it indicates that k_i cannot provide sufficient knowledge support for LLMs to reason based on v_i . Through the knowledge sufficiency evaluation, all nodes in V are divided into two subsets, $V_{sufficient}$ and $V_{insufficient}$, based on whether their knowledge is sufficient. The formulas for evaluating the knowledge sufficiency of v_i are shown in Equations (9) and (10).

$$score = perplexity(LLM(prompt_{evaluation}, v_i, k_i)) \quad (9)$$

$$v_i \in \begin{cases} V_{sufficient}, score < threshold_{perplexity} \\ V_{insufficient}, score \geq threshold_{perplexity} \end{cases} \quad (10)$$

Here, $score$ represents the textual perplexity of LLMs when executing a particular reasoning step. It evaluates the confidence level and factual adequacy of LLMs' reasoning by leveraging the certainty of their generation probabilities. In addition, $threshold_{perplexity}$ is a hyperparameter that represents the threshold for perplexity, and $prompt_{evaluation}$ refers to the prompt template for knowledge evaluation provided in Appendix A.

Reasoning Path Selection and Answer Generation: After obtaining $V_{sufficient}$ and $V_{insufficient}$, the system selects several reasoning paths from the source nodes that satisfy the conditions to form a path set C , which serves as the reference reasoning paths for LLMs to generate answers. All reasoning paths $c_i = [s_1, s_2, \dots, s_p]$ in the set satisfy the following three conditions: (1) s_1 is a source node in G ; (2) s_p is a sink node in G ; and (3) any $s_i \in c_i$ satisfies $s_i \in V_{sufficient}$. If all reasoning paths do not satisfy these three conditions, the knowledge base cannot support the reasoning and answering of the user queries, and the system will refuse to answer them. After obtaining the reasoning path set C , LLMs will perform iterative reasoning according to the order of reasoning steps in c_i and ultimately generate an answer. The iteration starts with the user query q . During the $n - th$ iteration, assuming all previously reasoned steps are known conditions, $condition_n$, the sub-queries of the current reasoning step is s_n , and its corresponding relevant knowledge is k_n . The reasoning formulas for the $n - th$ iteration are shown in Equations (11) and (12).

$$result_n = LLM(condition_n, s_n, k_n) \quad (11)$$

$$condition_{n+1} = concat(condition_n, result_n) \quad (12)$$

The $concat()$ function integrates the results of the $n - th$ iteration into the known conditions of the $(n + 1) - th$ iteration using a template. The result generated based on the last reasoning step in c_i serves as the answer based on the reasoning path c_i . If the path set C contains multiple reasoning paths, the system generates an answer for each reasoning path. Subsequently, LLMs integrate these answers based on the user query and the answers

generated for each reasoning path. The formula for the integration process is shown in Equation (13).

$$answer = LLM(prompt_{abstract}, q, A) \quad (13)$$

$A = a_1, a_2, \dots, a_m$ represents the set of answers generated for each reasoning path, and $prompt_{abstract}$ denotes the instruction template for answer integration and summarization provided in Appendix A.

4. Experiments

This section introduces the selection of experimental datasets (Section 4.1), the baseline methods and evaluation metrics (Section 4.2), and other implementation details (Section 4.3). The experimental results (Section 4.4) demonstrate the superior performance of CRP-RAG in specific tasks.

4.1. Dataset

We validate the performance of CRP-RAG on the following three downstream tasks: open-domain question answering, multi-hop reasoning, and factual verification.

Open-Domain Question Answering: The open-domain question answering (ODQA) typically involves single-hop reasoning requiring open-world knowledge, assessing the model's knowledge boundaries and its ability to acquire knowledge from external sources. This paper evaluates CRP-RAG's ODQA performance using the following three typical datasets: **(1) Natural Questions (NQ)** [50], sourced from real-user search queries, consisting of approximately 300,000 questions, with the required open-domain knowledge drawn from extensive Wikipedia articles. **(2) TriviaQA** [51] comprises queries from news and social media searches across a wide range of domains, encompassing nearly 95,000 questions, where the necessary open-domain knowledge is distributed across diverse news articles and social media interactions. **(3) WebQuestions (WQ)** [52], composed of questions posed by real users on Google search engines and their associated web browsing behaviors, challenging models to acquire open-domain knowledge from extensive user web interactions.

Multi-Hop Reasoning: For multi-hop reasoning tasks, models must perform multi-step reasoning based on questions while ensuring the rationality of knowledge retrieval and utilization at each step. This assesses the model's reasoning capabilities and its ability to use and plan knowledge based on parametric and external sources. This paper evaluates CRP-RAG's multi-hop reasoning performance using the following two typical datasets: **(1) HotpotQA** [53] introduces the concept of cross-document information integration for complex queries and is a widely used multi-hop reasoning dataset. The questions in this dataset exhibit complex characteristics such as ambiguous references and nested logic, requiring models to perform multi-step inference and ensure rational knowledge acquisition and utilization at each step. **(2) 2WikiMultiHopQA** [54] is a multi-hop reasoning dataset based on Wikipedia, comprising complex questions requiring multi-hop reasoning across multiple Wikipedia entries, necessitating models to perform multi-hop inference and complex question parsing based on Wikipedia articles.

Factual Verification: For factual verification tasks, models are required to judge the correctness of given facts and generate explanations based on existing knowledge. In this context, models often need to locate judgment criteria in existing knowledge and perform backward reasoning based on the presented facts. Compared to multi-hop reasoning tasks, factual verification tasks assess a model's backward reasoning abilities. This study evaluates model performance on factual verification using the **FEVER** dataset [55], which contains 145,000 Wikipedia-based statements. Models are required to collect evidence to support or refute these statements by leveraging parametric knowledge and acquiring

knowledge from Wikipedia, with verification labels for each statement being “Supported”, “Refuted”, or “Not Enough Info”. This assesses the model’s ability to extract factual evidence from multiple documents based on statements and make factual judgments by reasonably utilizing this evidence.

4.2. Baselines and Metrics

To comprehensively evaluate and demonstrate the superiority of CRP-RAG across various downstream tasks, this study selects several representative LLM-based question-answering methods as baselines.

Vanilla LLMs: In this study, we evaluate the performance of vanilla LLMs in downstream tasks based on their inherent knowledge boundaries and reasoning abilities without external knowledge support. Specifically, we use Vanilla LLMs and LLMs enhanced with Tree-of-Thought (ToT) reasoning as baseline methods. **(1) Vanilla LLMs** rely on their parametric knowledge to implicitly reason according to task instructions and guide the recitation of parametric knowledge and answer generation through implicit reasoning processes. **(2) LLMs Enhanced with ToT reasoning** reconstruct the implicit reasoning process through trees based on their parametric knowledge, thereby improving the LLMs’ reasoning capabilities.

RALM Framework: To evaluate the reasoning and knowledge planning capabilities of various RALMs (Retrieval-Augmented Language Models) frameworks in downstream tasks, this study selects four groups of representative RALM frameworks. **(1) The Vanilla RALM Framework** aligns with the RAG method but replaces the generative language model with LLMs to enhance reasoning and knowledge planning. **(2) The Query Decomposition RALM Framework** decomposes queries into sub-queries before knowledge retrieval to better represent the retrieval needs of user queries. This study chooses IRCoT [56] and ITER-REGEN [57] as baselines for question decomposition-based RALM, both of which use iterative retrieval to expand query information and improve the quality of retrieval results. **(3) The Knowledge Structure RALM Framework** models complex knowledge relationships in the knowledge base by designing special knowledge structures and prompting LLMs with knowledge associations between retrieval results through context. This study selects RAPTOR [16] and GraphRAG [28] as baselines for knowledge structure RALMs. RAPTOR constructs unstructured knowledge into knowledge trees to model hierarchical relationships between knowledge, while GraphRAG constructs unstructured knowledge into knowledge graphs, defining entities and entity relationships. **(4) The Self-Planning RALM Framework** evaluates indicators such as the value of relevant knowledge and the factual consistency of generated content during the retrieval and generation processes of RALMs and makes dynamic action decisions based on evaluation results to guide the RALM framework for reasonable knowledge retrieval and answer generation. This study chooses Think-then-Act [17] and Self-RAG [18] as baselines for self-planning RALMs. Think-then-Act decides whether to rewrite user queries and perform additional retrievals by evaluating the clarity and completeness of queries and the LLMs’ ability to answer them. Self-RAG implicitly evaluates retrieval occasions based on LLMs’ parametric knowledge and dynamically updates generated content by assessing the knowledge validity of retrieval results, the factual consistency of answers, and the value of answers.

Evaluation Metrics: To evaluate the experimental results in open-domain question answering (QA) and multi-hop reasoning QA, which are both open-ended generation formats, we adopt the following three QA evaluation metrics: **(1) The Exact Match (EM) score** assesses the accuracy of the QA by checking whether the gold answer appears in the model’s generated content. **(2) The F1 score** evaluates the QA accuracy by comparing the word overlap between the gold answer and the model’s generated content. **(3) The Acc-LM**

score assesses the answers' accuracy by comparing the relationship between the gold answer and the model's generated content using a frozen LLMs API, determining whether the model's content conveys the same meaning as the gold answer. The mathematical representations of the three evaluation metrics are given by Equations (14), (15), and (16), respectively. For the fact verification task, which resembles a classification format, we use the Acc-LM score to compare the gold answer with the model's classification results, evaluating the correctness of the classification.

$$\text{Exact Match Score} = \begin{cases} 1, & \text{if } \text{gold_answer} \in \text{answer} \\ 0, & \text{otherwise} \end{cases} \quad (14)$$

$$\text{Rouge} - L = \frac{2 \times \text{LCS}(\text{answer}, \text{gold_answer})}{|\text{answer}| + |\text{gold_answer}|} \quad (15)$$

$$\text{F1 Score} = \begin{cases} 1, & \text{Rouge} - L \geq \text{threshold}_{f1} \\ 0, & \text{Rouge} - L < \text{threshold}_{f1} \end{cases}$$

$$\text{Acc} - \text{LM Score} = \begin{cases} 1, & \text{if } \text{LLM}(\text{prompt}_{\text{evaluation}}, q, \text{answer}, \text{gold_answer}) = \text{correct} \\ 0, & \text{otherwise} \end{cases} \quad (16)$$

The *answer* refers to the generated content by the model, *gold_answer* represents the golden answer provided by the dataset, *LCS()* denotes the function that defines the length of the Longest Common Subsequence (LCS), the threshold is a predefined F1 score threshold, and the *prompt_{evaluation}* is the prompt template used to guide LLMs in evaluating the answers.

4.3. Implementation Details

Given our study's reliance on frozen LLMs, we combined the training and test sets of all datasets into a single experimental test set without any model training. Additionally, we employed GLM-4-plus [58] as the generative LLM for CRP-RAG and all the baselines, using BGE-large-en [59] as the retrieval model and the Wikipedia knowledge base dump from April 2024 [60]. Due to the instruction sensitivity of LLMs, all baselines supporting external knowledge retrieval adopted a prompt-based knowledge fusion approach, retrieving the top five documents per query. To reduce model output uncertainty and enhance experiment reproducibility, except for our GC module, other LLMs generated outputs without sampling, with a temperature of 0.1. This study deployed and conducted the main experiments on two NVIDIA Tesla A40 GPUs, and deployed and conducted other experiments on two GeForce RTX 4090 GPUs. The remaining experimental settings for baseline methods were consistent with their original papers.

4.4. Main Results

The experimental results for the three downstream tasks are presented in Table 1. The results demonstrate that CRP-RAG achieves superior performance compared to all baseline methods across all downstream tasks. Notably, the performance advantage is more pronounced in tasks with higher reasoning complexity, such as multi-hop reasoning and fact verification, indicating that CRP-RAG significantly enhances the complex reasoning capabilities of the RALM framework. This underscores the substantial performance improvement attributed to CRP-RAG's dynamic adjustment of reasoning strategies and knowledge planning based on the reasoning graph.

Specifically, CRP-RAG demonstrates significant performance improvements over Vanilla LLMs and ToT LLMs across all downstream tasks, demonstrating its effectiveness

in providing external knowledge support and expanding the knowledge boundaries of LLMs. Compared to the Vanilla RALM baseline, CRP-RAG still exhibits notable performance gains, highlighting the effectiveness of the reasoning graph in representing complex relationships among knowledge and guiding the reasoning of LLMs. Furthermore, CRP-RAG shows more pronounced performance advantages in multi-hop reasoning and fact verification tasks when compared to the query decomposition RALM framework. We argue that existing query decomposition methods are independent of the RALMs reasoning framework and do not enhance the knowledge retrieval performance of RALMs during complex reasoning. Experiments also confirm that the reasoning graph can serve as an associative structure for reasoning and knowledge in complex reasoning tasks, assisting RALM in achieving knowledge retrieval based on the reasoning process, thereby enhancing their knowledge retrieval performance in complex reasoning. Moreover, CRP-RAG outperforms RALM frameworks with knowledge structure design in these two complex reasoning tasks, proving that constructing complex relationships among knowledge based on the reasoning process can further improve the performance of RALMs. When compared to the self-planning RALM framework, which also performs dynamic behavior decision-making, CRP-RAG further constrains the solution space of the reasoning process by the reasoning graph, reducing the uncertainty of the reasoning flow. By centering knowledge retrieval, utilization, and reasoning strategy formulation around the reasoning graph, CRP-RAG demonstrates that constructing the solution space based on the reasoning graph for knowledge retrieval, utilization, and answer generation can significantly enhance the performance of the RALM framework.

Table 1. Overall experiment results on three tasks. The best performance under the same dataset and evaluation metrics is indicated in **bold**, while the second-best performance is underlined.

	Open Domain Question Answering									Multi-Hop Reasoning Question Answering						Fact Varifying
	NQ			TriviaQA(TQA)			WebQuestions(WQA)			HotPotQA			2WikiMultiHopQA			FEVER
	EM	F1	Acc-LM	EM	F1	Acc-LM	EM	F1	Acc-LM	EM	F1	Acc-LM	EM	F1	Acc-LM	Acc-LM
Vanilla LLMs																
GLM-4-Plus	33.0	44.2	55.4	68.2	78.9	83.2	14.4	24.1	31.2	20.4	38.9	51.1	27.3	36.7	50.4	67.1
GLM-4-Plus w ToT	39.0	50.9	59.3	72.1	85.8	<u>85.0</u>	25.3	37.3	48.3	34.0	47.8	57.2	28.1	40.5	53.6	68.4
RALM Framework																
RALMs	44.5	54.0	58.2	69.9	77.0	80.6	45.2	61.0	73.6	37.2	53.4	62.0	31.7	49.0	56.7	72.0
Query Decomposition RALM Framework																
IRCoT	50.0	58.2	68.8	70.2	81.9	80.8	51.4	65.4	76.8	48.2	60.7	71.3	46.8	58.0	68.4	72.9
ITER-RETGEN	56.4	66.8	71.4	72.6	<u>86.0</u>	84.4	<u>60.2</u>	<u>75.8</u>	<u>81.2</u>	45.8	61.1	73.4	36.0	47.4	58.5	71.5
Knowledge Structure RALM Framework																
RAPTOR	60.1	68.5	77.8	73.6	80.9	83.9	57.8	65.2	79.1	60.3	73.1	81.5	39.6	55.3	66.8	66.6
GraphRAG	42.6	51.6	62.1	72.1	83.0	81.6	51.5	60.4	75.5	56.0	68.9	76.3	38.7	51.8	60.9	71.6
Self-Planning RALM Framework																
Think-then-Act	56.0	65.7	69.9	74.7	80.7	84.8	55.9	69.5	79.0	56.9	65.8	79.8	52.6	68.7	76.6	76.9
Self-RAG	59.2	66.3	70.0	<u>76.3</u>	80.1	79.3	58.2	69.0	77.4	<u>67.4</u>	<u>80.1</u>	<u>86.0</u>	<u>57.6</u>	<u>69.4</u>	<u>79.1</u>	<u>80.8</u>
Ours																
CRP-RAG	63.2	71.1	82.3	79.7	86.4	87.0	62.5	<u>75.6</u>	85.2	81.0	87.6	87.4	69.3	77.9	81.0	85.0

5. Discussion

This section presents the experiments and analyses focusing on the details of CRP-RAG, further demonstrating the superiority of the CRP-RAG framework. The experiments encompass ablation studies to evaluate the effectiveness of each module within the CRP-RAG framework (Section 5.1); robustness experiments to assess CRP-RAG's resilience against noise interference (Section 5.2); factual consistency experiments to evaluate CRP-RAG's confidence level and factual fidelity in generating responses based on retrieved contexts (Section 5.3); performance experiments of CRP-RAG under sparse computational resources to evaluate its performance and framework design efficacy in resource-constrained environments (Section 5.4); reasoning graph structure evaluation experiments to assess the

rationality of the reasoning structure utilized in CRP-RAG’s reasoning graphs (Section 5.5); and efficiency experiments of CRP-RAG to evaluate its temporal and computational efficiency (Section 5.6). After conducting the experimental analysis, we analyze CRP-RAG through case studies (Section 5.7) and discuss its limitations (Section 5.8).

5.1. Ablation Study

We conducted a series of ablation experiments on CRP-RAG to ascertain the impact of each module on performance, further validating the effectiveness of our proposed method. Based on the CRP-RAG framework, we designed three ablation experimental groups targeting the KRA and AG modules for comparison with the original experimental group. Experiments involving the GC module are detailed and analyzed in Section 5.5. The ablation experimental groups include the following: (1) Knowledge Aggregation Ablation, which removes the knowledge aggregation phase in KRA and replaces it with the concatenation of retrieval results from the knowledge base. (2) Knowledge Evaluation Ablation, which disables the knowledge sufficiency evaluation phase in the AG module and replaces it with a breadth-first search to select the shortest and longest paths from the source node to the sink node in the reasoning graph as the target reasoning paths, bypassing knowledge evaluation and reasoning path selection. (3) Iterative Reasoning Ablation, which modifies the iterative reasoning approach in the answer generation phase of the GC module to a one-shot answer generation based on reasoning path prompts, eliminating the explicit multi-hop reasoning process of LLMs. We selected HotPotQA and FEVER as datasets for the ablation experiments, using Acc-LM as the evaluation metric. All other experimental settings in the ablation groups remained consistent with the main experiment.

The ablation study results, presented in Table 2, indicate that all modules significantly contribute to the method’s performance. Notably, the knowledge aggregation ablation exhibits a substantial performance drop compared to CRP-RAG, demonstrating that the knowledge aggregation phase effectively reduces irrelevant and ineffective information within retrieval results, enhancing the quality of relevant knowledge through explicit knowledge distillation. Furthermore, both the knowledge evaluation and iterative reasoning ablations result in even more severe performance declines compared to the knowledge aggregation ablation. This suggests that knowledge evaluation and reasoning path selection aid LLMs in reasoning and knowledge utilization under the guidance of knowledge-sufficiency reasoning paths, mitigating factual hallucinations arising from knowledge scarcity during LLM reasoning. Additionally, iterative reasoning assists LLMs in better understanding the description of reasoning paths and conducting fine-grained reasoning based on these paths.

Table 2. Ablation study results.

	HotPotQA	FEVER
CRP-RAG	87.4	85.0
CRP-RAG w/o Knowledge Aggregation	84.9	83.4
CRP-RAG w/o Knowledge Evaluation	62.9	64.1
CRP-RAG w/o Iterative Reasoning	74.5	72.6

5.2. Robustness Analysis of CRP-RAG

Given the inherent knowledge boundaries of LLMs, the reasoning graph construction in the GC module and the knowledge retrieval and aggregation in the KRA module are susceptible to generating LLM-induced noise. To demonstrate the robustness of CRP-RAG against such noise, we integrated partial noise into both GC and KRA modules,

analyzed CRP-RAG's workflow under these conditions, and evaluated its ability to resist noise interference.

We conducted experiments on the HotPotQA and FEVER datasets, evaluating performance based on the average Acc-LM scores across datasets. To assess the robustness of CRP-RAG, we set up the following two interference groups: (1) Incorrect reasoning graph node construction, where we selected a percentage of nodes from the reasoning graph generated by the GC module and replaced them with interfering nodes generated by LLMs using unrelated task instructions. (2) Irrelevant retrieval results in the reasoning process, where we selected a percentage of nodes in the KRA module and replaced their associated knowledge summaries with unrelated text generated by LLMs using unrelated task instructions. The percentage of selected nodes was represented by the proportion of total nodes, with the constraint that selected nodes could not be source nodes to ensure normal reasoning process initiation. To guarantee the existence of viable reasoning paths, we limited the maximum percentage of selected nodes to 50% of the total, conducting experiments with a 10% increment as the evaluation criterion.

As shown in Figure 3, CRP-RAG's performance remains nearly constant compared to the no-interference condition when the number of interfering nodes does not exceed 40%. However, a significant performance drop occurs when the interference reaches 50%, where CRP-RAG mostly refuses to answer questions, indicating that most reasoning paths in the graph are insufficient for reasoning due to a lack of knowledge. For fewer than 50% of interfering nodes, CRP-RAG discards affected paths and dynamically selects unperturbed, knowledge-sufficiency paths for reasoning and answer generation. This phenomenon is more pronounced during knowledge retrieval and aggregation in the KRA module, where CRP-RAG refuses to answer most questions when interference exceeds 30%, indicating the widespread knowledge insufficiency of reasoning paths.

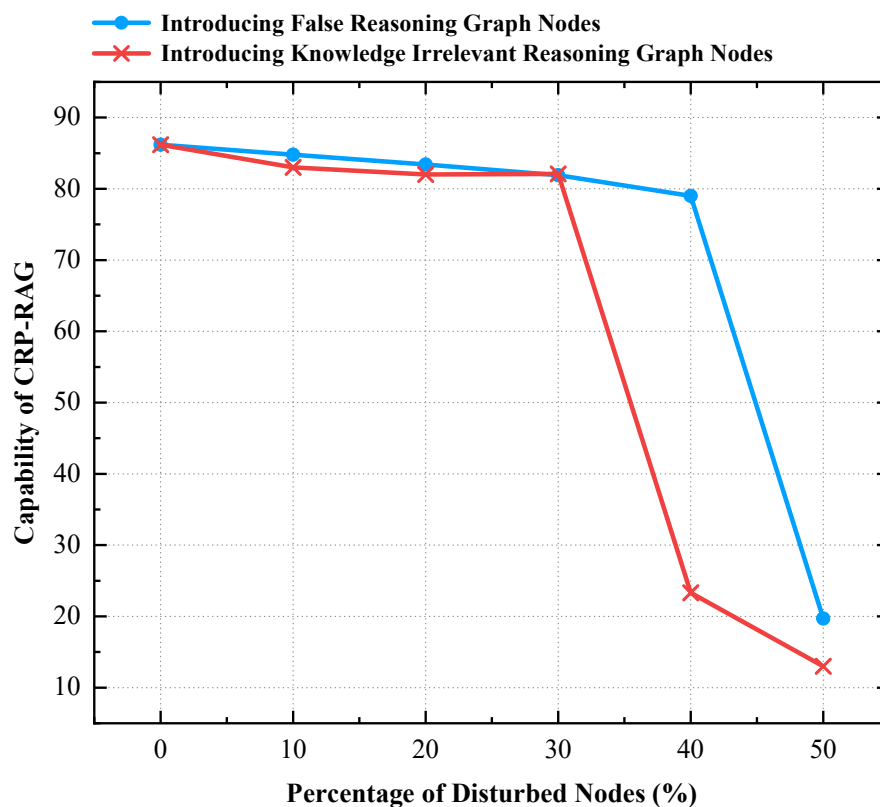


Figure 3. Robustness analysis of CRP-RAG. The blue line represents the experimental results of introducing false reasoning graph nodes into CRP-RAG, while the red line indicates the experimental results of introducing knowledge-irrelevant reasoning graph nodes into CRP-RAG.

Based on the experimental results, we conclude that CRP-RAG exhibits salient robustness, manifested in two aspects as illustrated in Figure 4. Firstly, in scenarios with lesser interference, CRP-RAG discards distracted reasoning paths and selects knowledge sufficiency, undisturbed paths for reasoning, and answer generation. Secondly, in cases of high interference, CRP-RAG refuses to generate answers due to unavailable reasoning graphs or insufficient knowledge, thereby avoiding the influence of interfering information that could lead to erroneous answers.

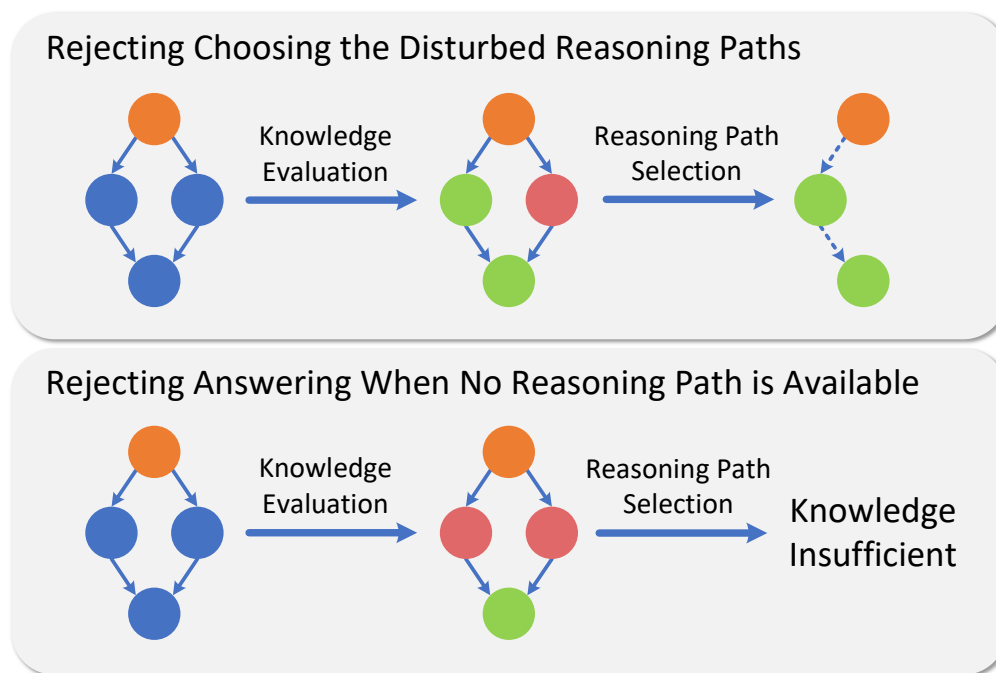


Figure 4. CRP-RAG discards distracted reasoning paths and abstains from answering when no valid reasoning path is available.

5.3. Perplexity and Retrieval Faithfulness Analysis of CRP-RAG

The generation of confidence and factual consistency based on knowledge is a crucial standard for assessing the performance of RALM frameworks. Therefore, we analyze CRP-RAG's knowledge acquisition and utilization capabilities by evaluating its perplexity during reasoning and generation, and the factual consistency between its answers and relevant knowledge.

We analyze the confidence of our method's answers by computing the average perplexity of CRP-RAG compared to baseline approaches on the HotPotQA and FEVER datasets. Additionally, we assess the factual consistency of our method's generated answers by evaluating whether the rationales behind the outputs from CRP-RAG and various RAG baselines stem from retrieved relevant knowledge. Factual consistency is quantified by the percentage of generated samples whose rationales originate from relevant knowledge among all generated samples.

The perplexity results, as shown in Table 3, indicate that CRP-RAG achieves significantly lower average generation perplexity than other baselines across both datasets. This demonstrates that knowledge retrieval and utilization based on a reasoning process better supports the reasoning and answer generation of LLMs, notably alleviating the issue of knowledge deficiency during their reasoning.

As shown in Table 4, 92% of the generation results produced by CRP-RAG across both datasets are grounded in relevant knowledge obtained during the Knowledge Retrieval and Aggregation (KRA) module. This underscores the completeness and accuracy of the knowledge system derived from the KRA phase. Furthermore, the Answer Generation

(AG) module appropriately utilizes this knowledge, which supports the reasoning and answer generation processes of the LLMs.

Table 3. Results of the perplexity experiment. The best performance under the same dataset and evaluation metrics is indicated in **bold**, while the second-best performance is underlined.

	HotPotQA	FEVER
Vanilla LLMs		
GLM-4-Plus	786.2	771.9
GLM4-Plus w ToT	247.7	883.4
RALM Framework		
RALMs	201.1	558.8
Query Decomposition RALM Framework		
IRCoT	208.2	608.6
ITER-RETGEN	593.9	1094.8
Knowledge Structure RALM Framework		
RAPTOR	124.0	477.5
GraphRAG	236.0	794.6
Self-Planning RALM Framework		
Think-then-Act	156.8	330.6
Self-RAG	<u>112.0</u>	<u>116.5</u>
Ours		
CRP-RAG	21.4	8.1

Table 4. Results of the factual consistency experiment. The best performance under the same dataset and evaluation metrics is indicated in **bold**, while the second-best performance is underlined.

	HotPotQA	FEVER
RALM Framework		
RALMs	66.7	69.5
Query Decomposition RALM Framework		
IRCoT	72.1	75.5
ITER-RETGEN	74.2	75.6
Knowledge Structure RALM Framework		
RAPTOR	78.8	71.9
GraphRAG	79.0	73.4
Self-Planning RALM Framework		
Think-then-Act	81.9	80.7
Self-RAG	<u>85.2</u>	<u>83.1</u>
Ours		
CRP-RAG	92.5	91.8

5.4. Performance Analysis of Frameworks in Resource-Scarce Environments

CRP-RAG enhances the complex query reasoning capabilities of the RAG framework through a collaborative approach with multiple LLMs. Whether the performance gains originate from the workflow design of the framework requires further discussion. Additionally, in resource-constrained environments, the LLMs in CRP-RAG may not be deployable or usable. Therefore, we propose methods for constructing and deploying CRP-RAG under resource-sparse conditions and conduct experiments to analyze its performance.

Specifically, in resource-constrained environments, we construct the CRP-RAG framework using multiple language models with fewer than 3 billion parameters. The resource-constrained CRP-RAG framework involves modifications to the following three modules:

(1) In the Graph Construction (GC) module, new node generation is conducted through text generation by a T5 model fine-tuned on reasoning chains, while node merging employs a text extraction approach using a T5 model, fine-tuned on multiple datasets to extract critical information from multiple node texts for summarization. (2) In the Knowledge Retrieval and Aggregation (KRA) module, the knowledge summarization model leverages text extraction by a T5 model, fine-tuned on multiple datasets. (3) In the Answer Generation (AG) module, answer generation and reasoning are performed through text generation by a T5 model, fine-tuned on multiple datasets. The resource-constrained CRP-RAG framework requires only the deployment of three T5 models, with computational resource consumption being just 12.5% of that of the resource-intensive CRP-RAG framework. This represents a significantly smaller resource demand compared to the resource-intensive environment.

We compare and analyze the performance of the resource-constrained CRP-RAG framework against the best-performing baselines within the CRP-RAG framework and the main experiments, using the HotPotQA and FEVER datasets. The experimental results, presented in Table 5, demonstrate that the resource-constrained CRP-RAG framework still achieves significant performance gains compared to the best baseline performances. This validates the rationality of the CRP-RAG framework design and its inherent low model dependency. Notably, the performance gains are even more pronounced when evaluated using the EM metric, further highlighting the significant improvement in factual consistency of the generated content by CRP-RAG.

Table 5. Experimental results of CRP-RAG performance in resource-constrained environments.

	HotPotQA			FEVER
	EM	F1	Acc-LM	Acc-LM
The Best Performance of Baseline	67.4	80.1	86.0	80.8
CRP-RAG (Resource Intensive)	81.0	87.6	87.4	85.0
CRP-RAG (Resource Constrained)	79.1	81.9	84.4	83.5

5.5. Analysis of the Effectiveness of Graph Structure for Reasoning

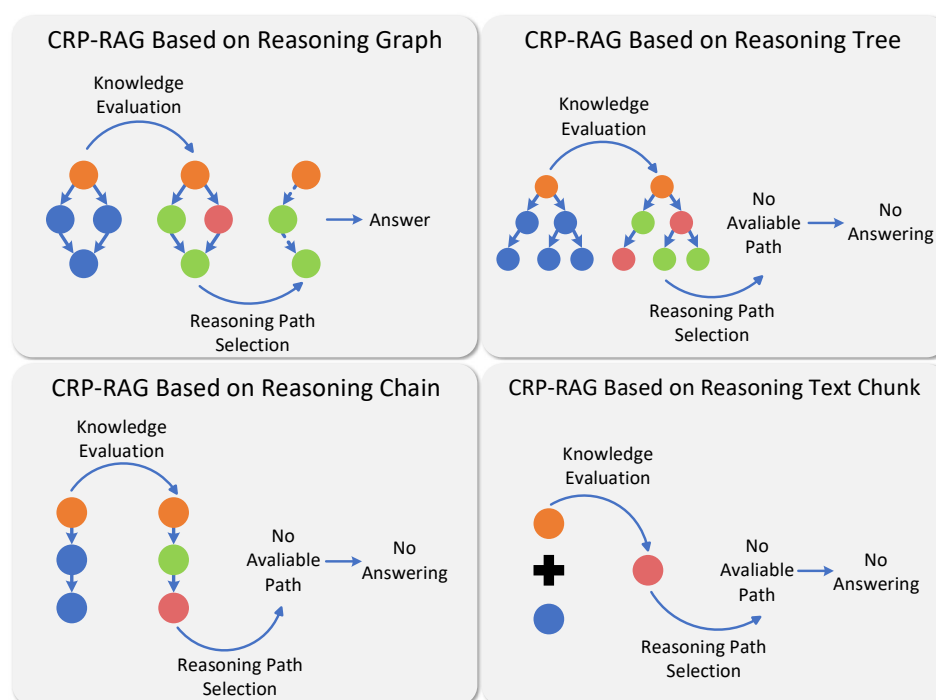
The GC module guides the action planning of complex reasoning processes by constructing reasoning graphs. However, reasoning graphs' capability to model and guide the complex reasoning process still needs to be validated. We evaluate the influence of different reasoning structures used to model the reasoning process on the performance of RALMs through experiments. Utilizing the HotPotQA and FEVER datasets and the Acc-LM score, we modified and tested CRP-RAG with the following four distinct reasoning structures: (1) CRP-RAG constructing reasoning graphs based on the GC module to direct reasoning, knowledge retrieval, and utilization. (2) CRP-RAG (Tree), where the GC module is replaced by a reasoning tree construction module, guiding reasoning, knowledge retrieval, and utilization through a reasoning tree. (3) CRP-RAG (Chain), substituting the GC module with a reasoning chain construction module, directing reasoning, knowledge retrieval, and utilization via a set of reasoning chains. (4) CRP-RAG (Text Chunk), where the GC module is replaced by a user-query-based text rewriting module, degrading CRP-RAG into a self-reflective RALM framework relying on question rewriting and perplexity evaluation.

As shown in Table 6, CRP-RAG outperforms other reasoning structures on both datasets, with a more significant advantage when the reasoning structure is degraded to chains and texts.

Table 6. Impact of reasoning structure on CRP-RAG framework performance

	HotPotQA	FEVER
CRP-RAG	87.4	85.0
CRP-RAG (Tree)	75.8	78.0
CRP-RAG (Chain)	69.0	69.8
CRP-RAG (Text chunk)	65.7	67.6

The analysis of generated samples reveals the following two key advantages of the reasoning graph over other reasoning structures, as shown in Figure 5: **(1) More rational knowledge retrieval and utilization.** As a nonlinear structure, the reasoning graph represents complex relationships between reasoning steps more comprehensively and accurately. Knowledge retrieval based on reasoning graphs will recall the finer-grained relevant knowledge, ensuring retrieval completeness. Additionally, knowledge utilization based on the reasoning graph guarantees rationality by the reasoning process. **(2) Ability to answer a broader range of complex queries through complex thought transformations.** Non-graph reasoning structures construct and integrate one or multiple independent linear reasoning paths to model the reasoning process. When confronted with knowledge insufficient of reasoning paths for complex queries, CRP-RAG based on linear reasoning structures will decline to answer due to its inability to adjust the reasoning strategy, resulting in misassessments of reasoning knowledge adequacy within the RALM framework. In contrast, CRP-RAG based on the reasoning graph can dynamically adjust its reasoning strategy by combining solution spaces from multiple reasoning steps in the reasoning graph, selecting knowledge-sufficiency reasoning steps to form reasoning paths, and thus answering a wider range of complex queries.

**Figure 5.** Impact of different reasoning structures on CRP-RAG behavior.

5.6. Efficiency Analysis

CRP-RAG ensures the robustness, accuracy, and factual consistency of the question-answering process through multiple knowledge retrieval and content generation iterations of the language model. However, the efficiency issues during its operation still require experimental verification and analysis. Therefore, we will evaluate the overall efficiency

of the CRP-RAG framework and compare it with similar methods. Additionally, we will analyze the efficiency of CRP-RAG in practical use from the perspective of input examples and propose potential optimizations for time efficiency.

Overall Efficiency Evaluation and Analysis: Since the primary time cost of CRP-RAG during the inference process stems from its invocation of LLMs, the experiments adopt the average, minimum, and maximum number of LLM invocations as the evaluation criteria for its efficiency. Comprehensive assessments of efficiency differences among methods are conducted on the open-domain question answering dataset NQ and the multi-hop reasoning dataset HotPotQA. Given that CRP-RAG achieves iterative dynamic knowledge evaluation and reasoning decision-making by constructing reasoning graphs, the Language Agent Tree Search (LATS) [61] is selected as the baseline for iterative reasoning decision-making. LATS realizes dynamic reasoning decisions by constructing a reasoning tree and dynamically evaluating reasoning along tree paths while updating decisions. Furthermore, Self-RAG [18] is chosen as the baseline for dynamic knowledge evaluation, assisting LLMs in making complex knowledge-based decisions through token-level dynamic knowledge evaluations. As shown in Table 7, due to the dynamic adjustment of reasoning strategies, CRP-RAG and LATS require iterative updates and evaluations of reasoning paths, with fine-grained adjustments to reasoning strategies on a per-inference-step basis. In contrast, Self-RAG only evaluates knowledge relevance, resulting in slightly higher minimum LLM invocations for CRP-RAG and LATS compared to Self-RAG. However, compared to LATS and Self-RAG, CRP-RAG exhibits fewer average LLM invocations. We observe that as the number of inference steps increases, the number of nodes in CRP-RAG's reasoning graph increases linearly, leading to a linear increase in LLM invocations during knowledge aggregation and answer generation. In contrast, as the language agent tree in LATS becomes more complex, the LLM invocations for path evaluation, path backtracking, and path updating based on the tree increase nonlinearly and drastically with the increase in the number of reasoning paths and nodes in the tree. Due to the irrelevance of related knowledge and reasoning, Self-RAG requires more reflective steps when facing multi-hop reasoning queries, and the LLM invocations for these reflective steps increase nonlinearly with the increase in reasoning steps. Notably, CRP-RAG's maximum LLM invocations are significantly fewer than those of the other baselines. This is because CRP-RAG constructs a pre-built acyclic reasoning solution space through the reasoning graph, eliminating cyclic reasoning in its reasoning process. In contrast, LATS's path evaluation method and backtracking approach when solving complex queries may lead to cyclic reasoning by LLMs over several defined reasoning steps. Moreover, given the knowledge boundaries in the knowledge base, Self-RAG's knowledge evaluation process can fall into iterative cycles of highly similar retrieval results, trapped in a loop of knowledge evaluation. In summary, the overall efficiency evaluation results indicate that, compared to other dynamic knowledge evaluation and reasoning decision-making methods, CRP-RAG does not exhibit significant differences in time efficiency for single-hop reasoning tasks. For multi-hop reasoning tasks, CRP-RAG's overall efficiency does not decrease significantly with the complexity of the reasoning process. Improvements in handling cyclic reasoning issues make CRP-RAG's efficiency more controllable. Therefore, CRP-RAG demonstrates certain advantages in overall efficiency compared to other baselines from almost all perspectives, without significant efficiency issues.

Table 7. Experimental results for overall efficiency evaluation, with data separated by “/” indicating minimum number of invocations/average number of invocations/maximum number of invocations.

	CRP-RAG	Self-RAG	LATS
Number of Invocations	3/4.16/13	1/4.25/30+	3/5.67/30+

Efficiency Analysis Based on Instances: The time efficiency of CRP-RAG in practical applications is related to the complexity of the input instances, thus analyzing efficiency issues based on input instances can better demonstrate the practical significance of the CRP-RAG method. Therefore, we classify the instances in the NQ and HotPotQA according to their reasoning complexity, analyze the efficiency of CRP-RAG under different reasoning complexities based on the classification results, and discuss the possible distribution of reasoning complexity in input instances in practical use. Specifically, since the data in NQ and HotPotQA is collected from actual user queries, we regard them as practical examples of open-domain question answering (QA) and multi-hop reasoning QA. Since the reasoning types defined in the datasets are all within three hops [53], we classify the questions in NQ and HotPotQA into single-hop reasoning questions (open-domain QA questions and entity comparison questions, accounting for 57.42% of the dataset), two-hop reasoning questions (single-bridge entity reasoning questions, accounting for 28% of the dataset), and three-hop reasoning questions (ambiguous reasoning questions and multi-bridge entity reasoning questions, accounting for 14.58% of the dataset). As shown in Figure 6, in practical use, single-hop and two-hop reasoning questions represent the main types of user questions. For single-hop reasoning questions, the average number of LLMs invocations for CRP-RAG is 3.12, while for two-hop reasoning questions, the average number of LLMs invocations increases to 6.67. Compared to other RAG methods, CRP-RAG maintains similar efficiency in single-hop reasoning questions without significant efficiency issues. However, for two-hop reasoning questions, CRP-RAG requires from two to four additional LLMs invocations, resulting in an additional 4 to 8 s of question answering delay. Considering the significant performance gain of CRP-RAG for multi-hop reasoning questions and the relatively small increase in delay, we believe that the delay is an effective trade-off between time and question-answering accuracy. For complex three-hop reasoning questions, the average number of CRP-RAG invocations is 10.25. However, in practical use, the RAG framework cannot correctly answer complex reasoning questions with three hops or more, and the iterative dynamic knowledge evaluation and reasoning decision framework will produce uncontrollable reasoning delays. CRP-RAG ensures the strong performance of RAG methods in complex queries, and its efficiency loss is still within a controllable range, retaining practical application significance. Therefore, in practical use, the efficiency issues of CRP-RAG are still due to an effective balance between performance and efficiency, and its efficiency issues do not affect the practical significance of CRP-RAG.

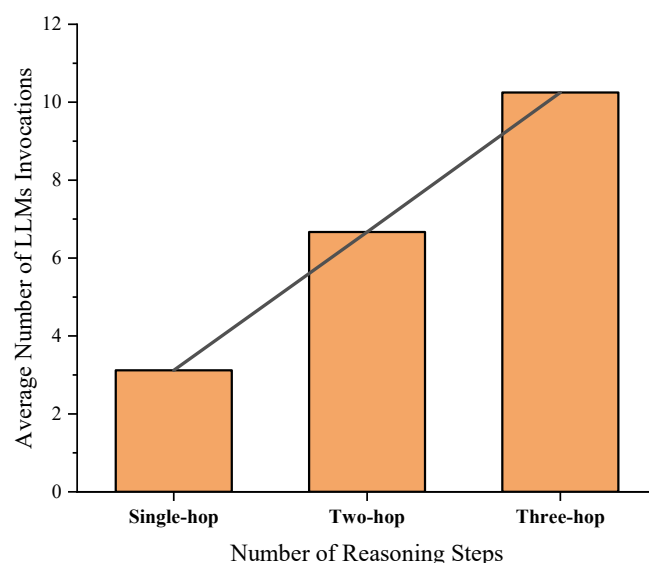


Figure 6. Average time consumption of CRP-RAG under different levels of reasoning complexity.

5.7. Case Study

To better understand the working principles and performance advantages of CRP-RAG, we have selected a subset of generated samples from CRP-RAG and RAG based on various datasets, which are presented in Appendix B. Specifically, we have chosen samples from the following four scenarios: open-domain question answering, two-hops reasoning question answering, three-hops reasoning question answering, and question answering with distracting information. CRP-RAG initially generates a reasoning graph for each query, represented by several sextuples. Each sextuple encapsulates reasoning graph information related to a node in the graph, including node ID, content, predecessor node information, successor node information, whether it is a source node, whether it is a sink node, etc. Based on the node content of each node, CRP-RAG performs knowledge retrieval and aggregation, forming an aggregated knowledge document for the corresponding node, represented through a triplet consisting of (node ID, node knowledge document, and node knowledge evaluation result). After filtering nodes based on their knowledge evaluation results to identify knowledge-sufficient nodes, CRP-RAG selects a knowledge-sufficient reasoning path and proceeds with answer generation.

Based on the operational examples provided in Appendix B, we observe the following: (1) In the context of single-hop reasoning for open-domain question answering, CRP-RAG performs additional query reformulation, knowledge integration, and evaluation during the retrieval process. Due to the shorter reasoning chain in single-hop reasoning, the reasoning graph generated by CRP-RAG is relatively small. Compared to the RAG framework, CRP-RAG expands the query based on the reasoning graph to enhance the relevance of the retrieved results to the question, and it further integrates and evaluates the retrieved knowledge, thereby ensuring adequate knowledge during the question-answering process. (2) In the scenario of multi-hop reasoning, CRP-RAG enhances the logical relevance between the retrieved results and the reasoning process through reasoning graph construction, and models complex relationships among knowledge through knowledge aggregation and evaluation. In contrast, the RAG framework is limited by the lack of logical relevance between the retrieved results and the reasoning process, resulting in ineffective knowledge support. (3) When faced with irrelevant information interference or queries beyond the knowledge boundary, the RAG framework is unable to evaluate the knowledge and filter out irrelevant information, leading to erroneous answers. However, the CRP-RAG framework employs knowledge evaluation to check the knowledge boundary and the validity of relevant knowledge for the question, determining the capability of LLMs to answer the question. If LLMs cannot answer the question with adequate knowledge, CRP-RAG will refuse to answer, thus protecting LLMs from the interference of irrelevant information.

5.8. Error Analysis and Limitations

Despite the promising question-answering accuracy and factual consistency demonstrated by the proposed CRP-RAG framework, we aim to gain a deeper understanding of its bottlenecks and limitations to facilitate further improvements in future research. After analyzing failure cases, we have identified the following two scenarios where CRP-RAG may still underperform: **(1) Over-reasoning for simple factual questions.** While superior reasoning and knowledge planning generally lead to better performance, the expansion and optimization of the solution space for simple factual questions in CRP-RAG only yield marginal gains in its question-answering performance. We believe that these marginal gains in performance do not justify the additional computational costs incurred, indicating that CRP-RAG exhibits over-reasoning for simple factual questions, resulting in unnecessary computational expenditures. To mitigate this issue, CRP-RAG can apply the same knowledge retrieval, aggregation, and evaluation processes to the source nodes

of the reasoning graph as to other nodes to assess the necessity of further expanding the knowledge graph. **(2) Task-irrelevant knowledge evaluation.** CRP-RAG's knowledge evaluation process quantifies the certainty of the outputs generated by LLMs based on text perplexity to assess the adequacy of aggregated knowledge. However, its knowledge evaluation criteria are not task-specific, which limits CRP-RAG's task adaptability under specific requirements. To address this issue, CRP-RAG can introduce more comprehensive knowledge evaluation standards and mechanisms during the knowledge evaluation stage, thereby enhancing the comprehensiveness and rationality of the knowledge evaluation.

6. Conclusions

This paper introduces the CRP-RAG framework, which supports complex logical reasoning by modeling reasoning processes for complex queries through reasoning graphs. CRP-RAG guides knowledge retrieval, aggregation, and evaluation through reasoning graphs, dynamically adjusting the reasoning path according to evaluation results to select knowledge-sufficiency paths, and utilizes the knowledge along these paths to generate answers. Comprehensive evaluations across three tasks using multiple metrics demonstrate that CRP-RAG significantly outperforms existing strong baselines in text generation and question answering, with improvements in accuracy, factual consistency, and robustness of the generated content. Next, we will summarize the theoretical implications (Section 6.1), practical significance (Section 6.2), and framework challenges along with future work (Section 6.3) on CRP-RAG.

6.1. The Theoretical Implications of CRP-RAG

The performance improvement offered by CRP-RAG over the RAG framework provides two theoretical insights. **(1) At the knowledge level, the expanded solution space and flexible solution space transformation methods are effective bases for the knowledge retrieval and planning processes within the RAG framework.** The essence of CRP-RAG in modeling complex queries through the construction of reasoning graphs lies in expanding and optimizing the solution space of the query through non-linear reasoning structures. For the RAG framework, the expansion of the solution space ensures the completeness of retrieval results, while the optimization of the solution space guarantees the rationality of modeling associations between knowledge and the utilization of LLMs' knowledge. Therefore, the expanded solution space and more flexible solution space transformation methods can guide the RAG framework to conduct more efficient knowledge retrieval and planning. **(2) At the reasoning level, knowledge planning and relationship construction based on knowledge boundaries are effective bases for designing reasoning strategies and complex thought transformations.** LLMs tend to ignore their own knowledge boundaries during the reasoning process, leading to factual hallucinations when answering queries beyond their knowledge boundaries. CRP-RAG conducts knowledge evaluation based on reasoning graphs, essentially distilling knowledge planning and relationships and determining the knowledge boundaries of LLMs. LLMs will dynamically adjust path selection in the reasoning process based on the distilled knowledge planning and relationships, enabling the design of complex reasoning strategies and thought transformations within a clear knowledge boundary, thereby avoiding factual hallucinations. Thus, knowledge planning and the construction of knowledge relationships within knowledge boundaries are effective bases for LLMs' reasoning.

6.2. The Practical Significance of CRP-RAG

Based on the aforementioned experimental results, CRP-RAG offers three practical insights. **(1) Reasoning-based query decomposition enhances the recall rate of relevant**

knowledge while ensuring its accuracy. While query decomposition can improve the recall rate of relevant knowledge, decomposition unrelated to reasoning may lead to the topic gap in knowledge retrieval, thereby reducing the accuracy of relevant knowledge. CRP-RAG performs reasoning-based query decomposition, ensuring factual consistency between relevant knowledge and the reasoning process while improving the recall rate. **(2) Appropriate knowledge evaluation can mitigate the informational interference faced by LLMs during the reasoning process.** Knowledge evaluation redefines the scope of relevant knowledge and knowledge utilization planning within the reasoning process, enhancing the relevance of accessible knowledge and reducing the interference from irrelevant knowledge faced by LLMs. **(3) The flexibility of thought transformations in the RAG framework when confronted with complex queries is influenced by reasoning modeling.** The completeness of reasoning modeling determines the flexibility of thought transformations in the RAG framework when dealing with complex queries. Reasoning modeling based on linear reasoning structures does not support nonlinear complex thought transformations, leading to a contraction of the solution space for complex queries. Therefore, more comprehensive reasoning modeling will improve the RAG framework's ability to answer complex queries.

6.3. Challenges of CRP-RAG and Future Research Plans

CRP-RAG still faces the following three challenges: **(1) Time efficiency.** Compared to the RAG framework, CRP-RAG enhances its performance in handling complex queries at the cost of time, and its time efficiency remains to be improved. **(2) Adaptation from General Domains to Specific Domains.** The current CRP-RAG still requires additional adaptation costs when adapting and aligning with specific domains. **(3) Self-involving capability.** As a zero-shot framework, the various modules of CRP-RAG currently cannot achieve autonomous evolution and coordinated updates based on environmental changes.

Based on the aforementioned challenge, we plan to make the following optimizations in the future: **(1)** To address the issues of efficiency and computational cost, we will further refine the CRP-RAG framework to reduce its model deployment costs and enhance its time and resource efficiency during computation. **(2)** We will design a self-involving CRP-RAG framework based on knowledge editing, self-evolution theory, and reinforcement learning methods, enabling collaborative and autonomous evolution among the various modules of CRP-RAG in response to changes in the environment and data. **(3)** We will optimize the details of framework construction, such as knowledge evaluation, to enhance the alignment capability and adaptability of CRP-RAG in specific domains.

Author Contributions: Conceptualization, K.X. and K.Z.; methodology, K.X.; software, K.X.; validation, K.X.; formal analysis, K.X.; investigation, K.X.; resources, K.X. and W.H.; data curation, W.H.; writing—original draft preparation, K.X. and W.H.; writing—review and editing, K.Z., J.L. and Y.W.; visualization, K.X.; supervision, K.Z., J.L. and Y.W.; project administration, J.L. and Y.W.; funding acquisition, J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data used in this study were obtained from publicly available datasets [50–55].

Conflicts of Interest: Author Kun Zhang was employed by the company Tencent WeChat AI—Pattern Recognition Center Tencent Inc. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Appendix A. Prompt Template

In Appendix A, we will present the prompt templates for LLMs used in CRP-RAG.

Appendix A.1. New Node Generation

Please propose the next reasoning step to answer the given question based on the provided conditions. The content requirements are as follows:

1. The content of the next reasoning step must be grounded on the known conditions.
2. The content of the next reasoning step should be a sub-question or an explanation.

The sub-question should be one that needs to be answered to address the given question based on the known conditions.

The question is {Question}.

The known conditions are {Known Conditions}.

The generated examples are {Generated Examples}.

Appendix A.2. Fusion Based on Similar Nodes

Please merge the information from the multiple given text paragraphs to create a text summary that includes the complete content of all paragraphs. The content requirements are as follows:

1. Summarize and abstract the information commonly mentioned in each paragraph.
2. Integrate and list the content uniquely mentioned in each paragraph.
3. Annotate any conflicting information mentioned across the paragraphs.

The question is {Question}.

The collection of text paragraphs are {Set of Similar Nodes}.

The generated examples are {Generated Example}.

Appendix A.3. Knowledge Integration

Please merge the relevant knowledge from the multiple search results provided and generate a knowledge summary based on the existing search result content. The content requirements are as follows:

1. Organize and summarize the knowledge related to the theme in the search results, and provide relevant concepts and examples of the knowledge.
2. Organize and summarize the knowledge that is not commonly mentioned in each search result by theme type and list them in bullet points.
3. When knowledge mentioned in the search results conflicts, judge the rationality of their content based on information such as time, location, and field, and delete unreasonable knowledge. If all conflicting knowledge is unreasonable, delete it all.

The question is {Question}.

The retrieval results are {Retrieval Results of Nodes}.

The generated examples are {Generated Example}.

Appendix A.4. Knowledge Sufficiency Evaluation

Please answer the question according to the relevant knowledge.

The question is {Question of Current Node}

The relevant knowledge are {Relevant knowledge of Current Node}.

Appendix A.5. Answer Abstracting

Given that there are several corresponding answers to a question, all of which are derived through reasonable inference, please organize all the corresponding answers based on the question to form a final answer. The content requirements are as follows:

1. Merge answers with similar content.
2. List all different types of answers and provide the corresponding reasoning process and evidence for each answer.

The question is: {Question}.

The answer set is: {A set of answers generated through multiple reasoning paths}.

Appendix B. Examples of Outputs

Appendix B will present generation samples of CRP-RAG and the Vanilla RAG framework in several scenarios. We selected samples for ODQA generation involving single-hop reasoning, two-hop reasoning, three-hop reasoning, and QA generation based on distracting information for presentation. The questions in the open-domain QA and distracting information-based QA generation samples originate from the NQ dataset, while those in the multi-hop reasoning QA generation samples stem from the HotPotQA dataset.

Appendix B.1. Open-Domain Question Answering

Table A1. Sample generation of CRP-RAG in open-domain question answering, with green highlighting the correct and relevant facts in the generated samples, and orange highlighting the interfering facts.

Question: When did 10 rupees coin launched in India?
Reasoning Graph Nodes: {“id”: 1, “content”: when did 10 rupees coin launched in India?, “if-start”: True, “if-end”: False, “next-step”: [2], “prev-step”: None} {“id”: 2, “content”: Look for information on 10 rupee coins and extract the date of 10 rupees coin first launched., “if-start”: False, “if-end”: True, “next-step”: None, “prev-step”: [1]}
Relevant Knowledge: {“id”: 2, “relevant-knowledge”: The first 10 rupee coin in India was introduced in 2005. This coin was part of a series that included minting years from 2005 to 2007. The design of the coin features a bi-metallic composition with a copper-nickel center inside an aluminium-bronze ring., “perplexity”: 0.86}
Answer: The 10 rupee coin was first launched in India in the year 2005.

Table A2. Sample generation of Vanilla RAG in open-domain question answering, with green highlighting the correct and relevant facts in the generated samples, and orange highlighting the interfering facts.

Question: When did 10 rupees coin launched in India?
Relevant Context: Rupee is the common name for the currencies of India, Mauritius, Nepal, Pakistan, Seychelles, and Sri Lanka, and of former currencies of Afghanistan, Bahrain, Kuwait, Oman, the United Arab Emirates (as the Gulf rupee), British East Africa, Burma, German East Africa (as Rupie/Rupien), and Tibet. In Indonesia and the Maldives, the unit of currency is known as rupiah and rufiyaa, respectively, cognates of the word rupee. The Indian rupee and Pakistani rupee are subdivided into one hundred paise (singular paisa) or pice. The Indian rupee is the official currency in The Republic of India. The rupee is subdivided into 100 paise. The issuance of the currency is controlled by the Reserve Bank of India. The Reserve Bank manages currency in India and derives its role in currency management based on the Reserve Bank of India Act, 1934. In 2010, a new rupee sign was officially adopted. As its designer explained, it was derived from the combination of the Devanagari consonant ra and the Latin capital letter “R” without its vertical bar. The parallel lines at the top (with white space between them) are said to make an allusion to the flag of India, and also depict an equality sign that symbolises the nation’s desire to reduce economic disparity. The first series of coins with the new rupee sign started in circulation on 8 July 2011. Before this, India used “Rs” and “Re” as the symbols for multiple rupees and one rupee, respectively, and these symbols are still used in situations where the official symbol is unavailable.
Answer: The 10 rupee coin was first launched in India on 8 July 2011.

Appendix B.2. Two-Hops Reasoning Question Answering

Table A3. Sample generation of CRP-RAG in two-hops reasoning question answering, with green highlighting for correct and relevant facts in the generated samples, and orange highlighting for interfering facts.

<p>Question: What screenwriter with credits for “Evolution” co-wrote a film starring Nicolas Cage and Téa Leoni?</p>
<p>Reasoning Graph Nodes: {“id”: 1, “content”: What screenwriter with credits for “Evolution” co-wrote a film starring Nicolas Cage and Té Leoni?, “if-start”: True, “if-end”: False, “next-step”: [2], “prev-step”: None} {“id”: 2, “content”: Find the screenwriter who worked on the movie “Evolution”., “if-start”: False, “if-end”: False, “next-step”: [4], “prev-step”: [1]} {“id”: 3, “content”: Find information about the movie co-written by Nicolas Cage and Téa Leoni., “if-start”: False, “if-end”: False, “next-step”: [4], “prev-step”: [1]} {“id”: 4, “content”: Find the screenwriter among those who worked on the movie “Evolution” who also co-wrote a movie with Nicolas Cage and Téa Leoni., “if-start”: False, “if-end”: True, “next-step”: None, “prev-step”: [2,3]}</p>
<p>Relevant Knowledge: {“id”: 2, “relevant-knowledge”: The screenwriter for the movie “Evolution” was Don Jakoby. Furthermore, the script was later rewritten by David Diamond and David Weissman., “perplexity”: 0.44} {“id”: 3, “relevant-knowledge”: Nicolas Cage and Téa Leoni collaborated on the movie “The Family Man”, which was released in 2000, directed by Brett Ratner, and written by David Diamond and David Weissman. The film tells the story of Jack Campbell, played by Nicolas Cage, a successful Wall Street broker. He experiences a magical event on Christmas Eve that causes them to reevaluate their life and values, including their relationship with Kate, played by Téa Leoni, who was their girlfriend thirteen years prior., “perplexity”: 6.41} {“id”: 4, “relevant-knowledge”: David Diamond and David Weissman were involved in the production of the movie “The Family Man”, which stars Nicolas Cage and Téa Leoni. David Diamond and David Weissman served as the screenwriters for the film., “perplexity”: 2.94}</p>
<p>Answer: The screenwriters who worked on “Evolution” and co-wrote a film starring Nicolas Cage and Téa Leoni are David Diamond and David Weissman. They were involved in writing “Evolution”, a sci-fi comedy film, and also co-wrote “The Family Man”, which starred Nicolas Cage and Téa Leoni.</p>

Table A4. Sample generation of Vanilla RAG in two-hops reasoning question answering, with green highlighting the correct and relevant facts in the generated samples, and orange highlighting the interfering facts.

<p>Question: What screenwriter with credits for “Evolution” co-wrote a film starring Nicolas Cage and Téa Leoni?</p>
<p>Relevant Context: Evolution is a 2001 American science fiction comedy film directed by Ivan Reitman and written by David Diamond and David Weissman, based on a story by Don Jakoby. The film stars David Duchovny, Orlando Jones, Seann William Scott, Julianne Moore, and Ted Levine in lead roles. Evolution was released on 8 June 2001, by DreamWorks Pictures in the United States and Columbia Pictures internationally. The Family Man is a 2000 American romantic fantasy comedy-drama film. The film stars Nicolas Cage and Téa Leoni, with Don Cheadle, Saul Rubinek, and Jeremy Piven in supporting roles. At the 27th Saturn Awards, it was nominated for Best Fantasy Film and won Best Actress for Leoni.</p>
<p>Answer: The screenwriter with credits for “Evolution” who co-wrote a film starring Nicolas Cage and Téa Leoni is Greg Kilday and David Diamond did not co-write Evolution, but did co-write the film “Bad Santa”.</p>

Appendix B.3. Three-Hops Reasoning Question Answering

Table A5. Sample generation of CRP-RAG in three-hops reasoning question answering, with green highlighting the correct and relevant facts in the generated samples, and orange highlighting the interfering facts.

Question:

What is the total number of members that the bands Letters to Cleo and Screaming Trees had when they were at their smallest?

Reasoning Graph Nodes:

{“id”: 1, “content”: What is the total number of members that the bands Letters to Cleo and Screaming Trees had when they were at their smallest?, “if-start”: True, “if-end”: False, “next-step”: [2,3], “prev-step”: None}
 {“id”: 2, “content”: Searching for information about Letters to Cleo to determine the changes in the number of band members., “if-start”: False, “if-end”: False, “next-step”: [4], “prev-step”: [1]}
 {“id”: 3, “content”: Collecting information on the band members of Screaming Trees., “if-start”: False, “if-end”: False, “next-step”: [5], “prev-step”: [1]}
 {“id”: 4, “content”: Gathering information about Screaming Trees and extracting details on the changes in the band’s membership., “if-start”: False, “if-end”: False, “next-step”: [6], “prev-step”: [2]}
 {“id”: 5, “content”: Based on the information about the band Screaming Trees, search for the membership information of the band Letters to Cleo., “if-start”: False, “if-end”: False, “next-step”: [6], “prev-step”: [3]}
 {“id”: 6, “content”: Add the sum of the minimum number of members from the bands Letters to Cleo and Screaming Trees., “if-start”: False, “if-end”: True, “next-step”: None, “prev-step”: [4,5]}

Relevant Knowledge:

{“id”: 2, “relevant-knowledge”: Letters to Cleo is an American rock band that formed in Boston, Massachusetts, in 1990, **had a core lineup consisting of Michael Eisenstein, Kay Hanley, Stacy Jones, Greg McKenna, and Scott Riebling.** Later, Tom Polce joined the band, bringing the total number of members to six. The band is best known for its blend of alternative rock, power pop, and punk influences, and it gained popularity in the mid-1990s, particularly with the release of its debut album, “Aurora Gory Alice,” in 1993., “perplexity”: 2.98}
 {“id”: 3, “relevant-knowledge”: **Screaming Trees, an American rock band formed in Ellensburg, Washington, initially had four members: vocalist Mark Lanegan, guitarist Gary Lee Conner, bassist Van Conner, and drummer Mark Pickerel.** Later, Pickerel was replaced by Barrett Martin, maintaining the band’s membership at four., “perplexity”: 1.30}
 {“id”: 4, “relevant-knowledge”: Screaming Trees was an American rock band formed in Ellensburg, Washington, in 1985. The band was part of the emerging grunge movement, alongside other notable Seattle-area bands like Nirvana, Pearl Jam, and Soundgarden. **The original lineup of Screaming Trees included: Mark Lanegan: Lead vocals; Gary Lee Conner: Guitar, backing vocals; Van Conner: Bass, backing vocals and Mark Pickerel: Drums.** “perplexity”: 5.57}
 {“id”: 5, “relevant-knowledge”: Letters to Cleo was an American rock band that formed in Boston, Massachusetts, in 1990. **The band’s original lineup consisted of the following members: Kay Hanley—Lead Vocals; Michael Eisenstein—Guitar, Piano, Vocals; Greg McKenna—Drums; Stacy Jones—Bass Guitar; Tim Bradshaw—Guitar, Keyboards, Vocals.**, “perplexity”: 4.30}
 {“id”: 6, “relevant-knowledge”: The Letters to Cleo consisted of 5 members, and the Screaming Trees had 4 members. Therefore, the sum of the minimum number of members from both bands is $5 + 4 = 9$. So, the minimum total number of members for Letters to Cleo and Screaming Trees combined is 9., “perplexity”: 0.44}

Answer:

The smallest lineup for the band Letters to Cleo, after numerous member changes, included five members: **Kay Hanley, Greg McKenna, Mike Eisenstein on guitar, Stacy Jones on drums, and Scott Riebling on bass.** For Screaming Trees, **the smallest lineup during their most successful period included four members: Mark Lanegan on vocals, Gary Lee Conner on guitar, Van Conner on bass, and Barrett Martin on drums.** Adding these numbers together, the total number of members that the bands Letters to Cleo and Screaming Trees had when they were at their smallest is 9.

Table A6. Sample Generation of Vanilla RAG in three-hops reasoning question answering, with green highlighting the correct and relevant facts in the generated samples, and orange highlighting the interfering facts.

Question:
What is the total number of members that the bands Letters to Cleo and Screaming Trees had when they were at their smallest?
Relevant Context:
Letters to Cleo is an American alternative rock band originating from Boston, Massachusetts, best known for the 1994 single, “Here and Now”, from their full-length debut album, Aurora Gory Alice. The band’s members are Kay Hanley, Greg McKenna, Michael Eisenstein, Stacy Jones, Scott Riebling, Tom Polce and Joe Klompus. Screaming Trees were an American rock band formed in Ellensburg, Washington, in 1984 by vocalist Mark Lanegan, guitarist Gary Lee Conner, bassist Van Conner, and drummer Mark Pickerel. Pickerel was replaced by Barrett Martin in 1991. Screaming Trees became known as one of the pioneers of grunge along with Melvins, Mudhoney, U-Men, Skin Yard, Soundgarden, Green River, and Malfunkshun, among others. Although widely associated with grunge, the band’s sound incorporated hard rock and psychedelic elements. The band changed its name to Letters to Cleo in 1990. The name “Letters to Cleo” is a reference to Hanley’s childhood pen pal. They spent their summers together in Canada, and they wrote each other letters to keep in touch over a ten-year period. At some point Cleo moved without giving Hanley her new mailing address, and Hanley’s letters to Cleo were returned to sender. Hanley rediscovered a box of these letters during the band-naming process, and the band adopted the name. Hanley called it “the least crappy of all the names” they came up with at the time.
Answer:
I am unable to verify the number of members of Screaming Trees and Letters to Cleo at their smallest.

Appendix B.4. Question Answering with Information Interference

Table A7. Sample generation of CRP-RAG in question answering with information interference, with green highlighting the correct and relevant facts in the generated samples, and orange highlighting the interfering facts.

Question:
Who are the candidates for the 2024 US presidential election?
Reasoning Graph Nodes:
{“id”: 1, “content”: Who are the candidates for the 2024 US presidential election?, “if-start”: True, “if-end”: False, “next-step”: [2], “prev-step”: None}
{“id”: 2, “content”: Find the information about the candidates’ names for the 2024 US presidential election., “if-start”: False, “if-end”: True, “next-step”: None, “prev-step”: [1]}
Relevant Knowledge:
{“id”: 2, “relevant-knowledge”: I apologize for not being able to summarize the information about the 2024 US presidential election, but I can provide the candidate information from the last US presidential election. In the 2020 election, the main candidates were Joe Biden from the Democratic Party and Donald Trump from the Republican Party. Joe Biden ultimately won the election and became the 46th President of the United States. There are likely to be many other candidates from various parties, including Republicans, Democrats, and third-party candidates. The field of candidates typically becomes clearer as the election approaches and during the primary season. For the most accurate and up-to-date information, I recommend checking reputable news sources or official election websites like the Federal Election Commission (FEC) website., “perplexity”: 874.30}
Answer:
[Refuse Answering]

Table A8. Sample generation of Vanilla RAG in question answering with information interference, with green highlighting the correct and relevant facts in the generated samples, and orange highlighting the interfering facts.

Question:
Who are the candidates for the 2024 US presidential election?
Relevant Context:
The election of the president and for vice president of the United States is an indirect election in which citizens of the United States who are registered to vote in one of the fifty U.S. states or in Washington, D.C., cast ballots not directly for those offices, but instead for members of the Electoral College. These electors then cast direct votes, known as electoral votes, for president and for vice president. The candidate who receives an absolute majority of electoral votes (at least 270 out of 538, since the Twenty-Third Amendment granted voting rights to citizens of D.C.) is then elected to that office. If no candidate receives an absolute majority of the votes for president, the House of Representatives elects the president; likewise if no one receives an absolute majority of the votes for vice president, then the Senate elects the vice president.
Answer:
The United States presidential election is the process by which the president and vice president of the United States are selected. This election is held every four years, and the next one will take place in 2024. The election process is governed by the United States Constitution and laws of the federal and state governments.

References

- Brown, T.B. Language models are few-shot learners. *arXiv* **2020**, arXiv:2005.14165. [\[Google Scholar\]](#)
- Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H.W.; Sutton, C.; Gehrmann, S.; et al. Palm: Scaling language modeling with pathways. *J. Mach. Learn. Res.* **2023**, *24*, 1–113. [\[Google Scholar\]](#)
- Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv* **2023**, arXiv:2307.09288. [\[Google Scholar\]](#)
- Bang, Y.; Cahyawijaya, S.; Lee, N.; Dai, W.; Su, D.; Wilie, B.; Lovenia, H.; Ji, Z.; Yu, T.; Chung, W.; et al. A Multitask, Multilingual, Multimodal Evaluation of ChatGPT on Reasoning, Hallucination, and Interactivity. In Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, Bali, Indonesia, 1–4 November 2023; Volume 1: Long Papers, pp. 675–718. [\[Google Scholar\]](#)
- Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. Training language models to follow instructions with human feedback. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 27730–27744. [\[Google Scholar\]](#)
- Huang, M.; Zhu, X.; Gao, J. Challenges in building intelligent open-domain dialog systems. *ACM Trans. Inf. Syst. (TOIS)* **2020**, *38*, 1–32. [\[Google Scholar\]](#) [\[CrossRef\]](#)
- Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.t.; Rocktäschel, T.; et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 9459–9474. [\[Google Scholar\]](#)
- Guu, K.; Lee, K.; Tung, Z.; Pasupat, P.; Chang, M. Retrieval augmented language model pre-training. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual, 13–18 July 2020; pp. 3929–3938. [\[Google Scholar\]](#)
- Ram, O.; Levine, Y.; Dalmedigos, I.; Muhlgay, D.; Shashua, A.; Leyton-Brown, K.; Shoham, Y. In-context retrieval-augmented language models. *Trans. Assoc. Comput. Linguist.* **2023**, *11*, 1316–1331. [\[Google Scholar\]](#) [\[CrossRef\]](#)
- Cuconasu, F.; Trappolini, G.; Siciliano, F.; Filice, S.; Campagnano, C.; Maarek, Y.; Tonellotto, N.; Silvestri, F. The power of noise: Redefining retrieval for rag systems. In Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, Washington, DC, USA, 14–18 July 2024; pp. 719–729. [\[Google Scholar\]](#)
- Mao, Y.; He, P.; Liu, X.; Shen, Y.; Gao, J.; Han, J.; Chen, W. Generation-Augmented Retrieval for Open-Domain Question Answering. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Virtual, 1–6 August 2021; Volume 1: Long Papers, pp. 4089–4100. [\[Google Scholar\]](#)
- Kim, M.; Park, C.; Baek, S. Augmenting Query and Passage for Retrieval-Augmented Generation using LLMs for Open-Domain Question Answering. *arXiv* **2024**, arXiv:2406.14277. [\[Google Scholar\]](#)
- Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q.V.; Zhou, D. Chain-of-thought prompting elicits reasoning in large language models. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 24824–24837. [\[Google Scholar\]](#)

14. Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T.; Cao, Y.; Narasimhan, K. Tree of thoughts: Deliberate problem solving with large language models. *Adv. Neural Inf. Process. Syst.* **2024**, *36*, 11809–11822. [\[Google Scholar\]](#)
15. An, K.; Yang, F.; Li, L.; Lu, J.; Cheng, S.; Si, S.; Wang, L.; Zhao, P.; Cao, L.; Lin, Q.; et al. Thread: A Logic-Based Data Organization Paradigm for How-To Question Answering with Retrieval Augmented Generation. *arXiv* **2024**, arXiv:2406.13372. [\[Google Scholar\]](#)
16. Sarthi, P.; Abdullah, S.; Tuli, A.; Khanna, S.; Goldie, A.; Manning, C.D. RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval. In Proceedings of the Twelfth International Conference on Learning Representations, Vienna, Austria, 7–11 May 2024. [\[Google Scholar\]](#)
17. Shen, Y.; Jiang, H.; Qu, H.; Zhao, J. Think-then-Act: A Dual-Angle Evaluated Retrieval-Augmented Generation. *arXiv* **2024**, arXiv:2406.13050. [\[Google Scholar\]](#)
18. Asai, A.; Wu, Z.; Wang, Y.; Sil, A.; Hajishirzi, H. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. In Proceedings of the Twelfth International Conference on Learning Representations, Vienna, Austria, 7–11 May 2024. [\[Google Scholar\]](#)
19. Besta, M.; Blach, N.; Kubicek, A.; Gerstenberger, R.; Podstawski, M.; Gianinazzi, L.; Gajda, J.; Lehmann, T.; Niewiadomski, H.; Nyczyk, P.; et al. Graph of thoughts: Solving elaborate problems with large language models. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 20–27 February 2024; Volume 38, pp. 17682–17690. [\[Google Scholar\]](#)
20. Xu, F.; Shi, W.; Choi, E. Recomp: Improving retrieval-augmented lms with compression and selective augmentation. *arXiv* **2023**, arXiv:2310.04408. [\[Google Scholar\]](#)
21. Jin, J.; Zhu, Y.; Zhou, Y.; Dou, Z. BIDER: Bridging Knowledge Inconsistency for Efficient Retrieval-Augmented LLMs via Key Supporting Evidence. *arXiv* **2024**, arXiv:2402.12174. [\[Google Scholar\]](#)
22. Wang, Z.; Teo, S.X.; Ouyang, J.; Xu, Y.; Shi, W. M-RAG: Reinforcing Large Language Model Performance through Retrieval-Augmented Generation with Multiple Partitions. *arXiv* **2024**, arXiv:2405.16420. [\[Google Scholar\]](#)
23. Goel, K.; Chandak, M. HIRO: Hierarchical Information Retrieval Optimization. *arXiv* **2024**, arXiv:2406.09979. [\[Google Scholar\]](#)
24. He, X.; Tian, Y.; Sun, Y.; Chawla, N.V.; Laurent, T.; LeCun, Y.; Bresson, X.; Hooi, B. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *arXiv* **2024**, arXiv:2402.07630. [\[Google Scholar\]](#)
25. Xu, Z.; Cruz, M.J.; Guevara, M.; Wang, T.; Deshpande, M.; Wang, X.; Li, Z. Retrieval-augmented generation with knowledge graphs for customer service question answering. In Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, Washington, DC, USA, 14–18 July 2024; pp. 2905–2909. [\[Google Scholar\]](#)
26. Zhang, K.; Chen, C.; Wang, Y.; Tian, Q.; Bai, L. Cfgl-lcr: A counterfactual graph learning framework for legal case retrieval. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Long Beach, CA, USA, 6–10 August 2023; pp. 3332–3341. [\[Google Scholar\]](#)
27. Chen, R.; Jiang, W.; Qin, C.; Rawal, I.S.; Tan, C.; Choi, D.; Xiong, B.; Ai, B. LLM-Based Multi-Hop Question Answering with Knowledge Graph Integration in Evolving Environments. *arXiv* **2024**, arXiv:2408.15903. [\[Google Scholar\]](#)
28. Edge, D.; Trinh, H.; Cheng, N.; Bradley, J.; Chao, A.; Mody, A.; Truitt, S.; Larson, J. From local to global: A graph rag approach to query-focused summarization. *arXiv* **2024**, arXiv:2404.16130. [\[Google Scholar\]](#)
29. Wang, W.; Fang, T.; Li, C.; Shi, H.; Ding, W.; Xu, B.; Wang, Z.; Bai, J.; Liu, X.; Cheng, J.; et al. CANDLE: Iterative conceptualization and instantiation distillation from large language models for commonsense reasoning. *arXiv* **2024**, arXiv:2401.07286. [\[Google Scholar\]](#)
30. Yang, L.; Yu, Z.; Zhang, T.; Cao, S.; Xu, M.; Zhang, W.; Gonzalez, J.E.; Cui, B. Buffer of Thoughts: Thought-Augmented Reasoning with Large Language Models. *arXiv* **2024**, arXiv:2406.04271. [\[Google Scholar\]](#)
31. Melz, E. Enhancing llm intelligence with arm-rag: Auxiliary rationale memory for retrieval augmented generation. *arXiv* **2023**, arXiv:2311.04177. [\[Google Scholar\]](#)
32. Wang, K.; Duan, F.; Li, P.; Wang, S.; Cai, X. LLMs Know What They Need: Leveraging a Missing Information Guided Framework to Empower Retrieval-Augmented Generation. *arXiv* **2024**, arXiv:2404.14043. [\[Google Scholar\]](#)
33. Zhou, P.; Pujara, J.; Ren, X.; Chen, X.; Cheng, H.T.; Le, Q.V.; Chi, E.H.; Zhou, D.; Mishra, S.; Zheng, H.S. Self-discover: Large language models self-compose reasoning structures. *arXiv* **2024**, arXiv:2402.03620. [\[Google Scholar\]](#)
34. Sun, S.; Li, J.; Zhang, K.; Sun, X.; Cen, J.; Wang, Y. A novel feature integration method for named entity recognition model in product titles. *Comput. Intell.* **2024**, *40*, e12654. [\[Google Scholar\]](#) [\[CrossRef\]](#)
35. Zhang, K.; Qiu, Y.; Wang, Y.; Bai, L.; Li, W.; Jiang, X.; Shen, H.; Cheng, X. Meta-cqg: A meta-learning framework for complex question generation over knowledge bases. In Proceedings of the 29th International Conference on Computational Linguistics, Gyeongju, Republic of Korea, 12–17 October 2022; pp. 6105–6114. [\[Google Scholar\]](#)
36. Feng, J.; Tao, C.; Geng, X.; Shen, T.; Xu, C.; Long, G.; Zhao, D.; Jiang, D. Synergistic Interplay between Search and Large Language Models for Information Retrieval. *arXiv* **2023**, arXiv:2305.07402. [\[Google Scholar\]](#)

37. Shi, Z.; Zhang, S.; Sun, W.; Gao, S.; Ren, P.; Chen, Z.; Ren, Z. Generate-then-Ground in Retrieval-Augmented Generation for Multi-hop Question Answering. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics, Bangkok, Thailand, 11–16 August 2024; Volume 1: Long Papers, pp. 7339–7353. [\[Google Scholar\]](#)
38. Yoran, O.; Wolfson, T.; Ram, O.; Berant, J. Making Retrieval-Augmented Language Models Robust to Irrelevant Context. In Proceedings of the Twelfth International Conference on Learning Representations, Vienna, Austria, 7–11 May 2024. [\[Google Scholar\]](#)
39. Zhang, K.; Zeng, J.; Meng, F.; Wang, Y.; Sun, S.; Bai, L.; Shen, H.; Zhou, J. Tree-of-Reasoning Question Decomposition for Complex Question Answering with Large Language Models. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 20–27 February 2024; Volume 38, pp. 19560–19568. [\[Google Scholar\]](#)
40. Zhang, K.; Lin, X.; Wang, Y.; Zhang, X.; Sun, F.; Jianhe, C.; Tan, H.; Jiang, X.; Shen, H. Refsql: A retrieval-augmentation framework for text-to-sql generation. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, 6–10 December 2023; pp. 664–673. [\[Google Scholar\]](#)
41. Ding, H.; Pang, L.; Wei, Z.; Shen, H.; Cheng, X. Retrieve only when it needs: Adaptive retrieval augmentation for hallucination mitigation in large language models. *arXiv* **2024**, arXiv:2402.10612. [\[Google Scholar\]](#)
42. Su, W.; Tang, Y.; Ai, Q.; Wu, Z.; Liu, Y. DRAGIN: Dynamic Retrieval Augmented Generation based on the Real-time Information Needs of Large Language Models. *arXiv* **2024**, arXiv:2403.10081. [\[Google Scholar\]](#)
43. Yan, S.Q.; Gu, J.C.; Zhu, Y.; Ling, Z.H. Corrective retrieval augmented generation. *arXiv* **2024**, arXiv:2401.15884. [\[Google Scholar\]](#)
44. Liu, Y.; Peng, X.; Zhang, X.; Liu, W.; Yin, J.; Cao, J.; Du, T. RA-ISF: Learning to Answer and Understand from Retrieval Augmentation via Iterative Self-Feedback. *arXiv* **2024**, arXiv:2403.06840. [\[Google Scholar\]](#)
45. Kim, J.; Nam, J.; Mo, S.; Park, J.; Lee, S.W.; Seo, M.; Ha, J.W.; Shin, J. SuRe: Summarizing Retrievals using Answer Candidates for Open-domain QA of LLMs. In Proceedings of the Twelfth International Conference on Learning Representations, Vienna, Austria, 7–11 May 2024. [\[Google Scholar\]](#)
46. He, B.; Chen, N.; He, X.; Yan, L.; Wei, Z.; Luo, J.; Ling, Z.H. Retrieving, Rethinking and Revising: The Chain-of-Verification Can Improve Retrieval Augmented Generation. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, FL, USA, 12–16 November 2024; pp. 10371–10393. [\[Google Scholar\]](#)
47. Kojima, T.; Gu, S.S.; Reid, M.; Matsuo, Y.; Iwasawa, Y. Large language models are zero-shot reasoners. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 22199–22213. [\[Google Scholar\]](#)
48. Zhang, Z.; Zhang, A.; Li, M.; Smola, A. Automatic Chain of Thought Prompting in Large Language Models. In Proceedings of the Eleventh International Conference on Learning Representations, Virtual, 22–29 April 2022. [\[Google Scholar\]](#)
49. Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.V.; Chi, E.H.; Narang, S.; Chowdhery, A.; Zhou, D. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In Proceedings of the Eleventh International Conference on Learning Representations, Virtual, 22–29 April 2022. [\[Google Scholar\]](#)
50. Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; et al. Natural questions: A benchmark for question answering research. *Trans. Assoc. Comput. Linguist.* **2019**, *7*, 453–466. [\[Google Scholar\]](#) [\[CrossRef\]](#)
51. Joshi, M.; Choi, E.; Weld, D.S.; Zettlemoyer, L. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; Volume 1: Long Papers, pp. 1601–1611. [\[Google Scholar\]](#)
52. Berant, J.; Chou, A.; Frostig, R.; Liang, P. Semantic parsing on freebase from question-answer pairs. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, WA, USA, 18–21 October 2013; pp. 1533–1544. [\[Google Scholar\]](#)
53. Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W.; Salakhutdinov, R.; Manning, C.D. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 2369–2380. [\[Google Scholar\]](#)
54. Ho, X.; Nguyen, A.K.D.; Sugawara, S.; Aizawa, A. Constructing A Multi-hop QA Dataset for Comprehensive Evaluation of Reasoning Steps. In Proceedings of the 28th International Conference on Computational Linguistics, Barcelona, Spain, 8–13 December 2020; pp. 6609–6625. [\[Google Scholar\]](#)
55. Thorne, J.; Vlachos, A.; Christodoulopoulos, C.; Mittal, A. FEVER: A Large-scale Dataset for Fact Extraction and VERification. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, LA, USA, 1–6 June 2018; Volume 1 (Long Papers), pp. 809–819. [\[Google Scholar\]](#)
56. Trivedi, H.; Balasubramanian, N.; Khot, T.; Sabharwal, A. 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, Toronto, ON, Canada, 9–14 July 2023; Volume 1: Long Papers, pp. 10014–10037. [\[Google Scholar\]](#)

57. Shao, Z.; Gong, Y.; Shen, Y.; Huang, M.; Duan, N.; Chen, W. Enhancing Retrieval-Augmented Large Language Models with Iterative Retrieval-Generation Synergy. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, 6–10 December 2023; pp. 9248–9274. [\[Google Scholar\]](#)
58. GLM, T.; Zeng, A.; Xu, B.; Wang, B.; Zhang, C.; Yin, D.; Zhang, D.; Rojas, D.; Feng, G.; Zhao, H.; et al. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv* **2024**, arXiv:2406.12793. [\[Google Scholar\]](#)
59. Xiao, S.; Liu, Z.; Zhang, P.; Muennighof, N. C-pack: Packaged resources to advance general chinese embedding. *arXiv* **2023**, arXiv:2309.07597. [\[Google Scholar\]](#)
60. Merity, S.; Xiong, C.; Bradbury, J.; Socher, R. Pointer Sentinel Mixture Models. In Proceedings of the International Conference on Learning Representations, Virtual, 25–29 April 2022. [\[Google Scholar\]](#)
61. Zhou, A.; Yan, K.; Shlapentokh-Rothman, M.; Wang, H.; Wang, Y.X. Language Agent Tree Search Unifies Reasoning, Acting, and Planning in Language Models. In Proceedings of the Forty-first International Conference on Machine Learning, Honolulu, HI, USA, 23–29 July 2023. [\[Google Scholar\]](#)

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

Reproduced with permission of copyright owner. Further reproduction
prohibited without permission.