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

Kehan Xu , Kun Zhang , Jingyuan Li * , Wei Huang , Yuanzhuo Wang *

Posted Date: 21 November 2024

doi: 10.20944/preprints202411.1648.v1

Keywords: Retrieval-Augmented Generation (RAG); Large Language Models (LLMs); reasoning graph; complex thought transformation; knowledge utilization

 doi:10.20944/preprints202411.1648.v1

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, China
[2] School of Information Science and Engineering, Yanshan University, Qinhuangdao, China
[3] Tencent WeChat AI - Pattern Recognition Center Tencent Inc, Beijing, China
[4] Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 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. Experimental results show that CRP-RAG significantly outperforms state-of-the-art LLMs and RAG baselines in three reasoning and question answering tasks, demonstrating superior factual consistency and robustness compared to existing RAG methods.

Keywords: Retrieval-Augmented Generation (RAG); Large Language Models (LLMs); reasoning graph; complex thought transformation; knowledge utilization

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. 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 the significant success of the RAG framework in open-domain question answering tasks, they still face two major challenges. Challenge 1: Interference from irrelevant retrieval results in the reasoning process. While the retrieval results of RAG are related to the topic of the queries, they

 doi:10.20944/preprints202411.1648.v1

often include documents that are not pertinent to the reasoning process [10]. As shown in Figure 1, a paragraph describing the color of horses owned by Joséphine de Beauharnais (Napoleon's first wife) might be retrieved in response to the query "What was the color of Napoleon's horse?", but it cannot support the RAG in deriving an answer through reasoning. These irrelevant documents act as information noise in the reasoning process, ultimately leading to hallucinations in the RAG framework's output. Challenge 2: Inability to plan knowledge utilization. RAG overlooks the complex relationships among the knowledge in retrieval results, failing to plan strategies such as the occasions and scope of knowledge use during the reasoning process. This results in misuse of retrieval results and reduced efficiency in knowledge utilization, ultimately leading to erroneous generation behavior by the RAG framework. To address Challenge 1, existing RAG methods advocate introducing special reasoning structures [11,12] to decompose and expand user queries into several sub-queries [13,14], aiming to provide a more detailed representation of the retrieval information needs based on the reasoning process. To tackle Challenge 2, current research designs and incorporates special knowledge structures during knowledge base construction to pre-establish relationships among knowledge [15,16]; on the other hand, it designs and integrates new RAG frameworks to dynamically plan and decide on retrieval and generation strategies [17,18].

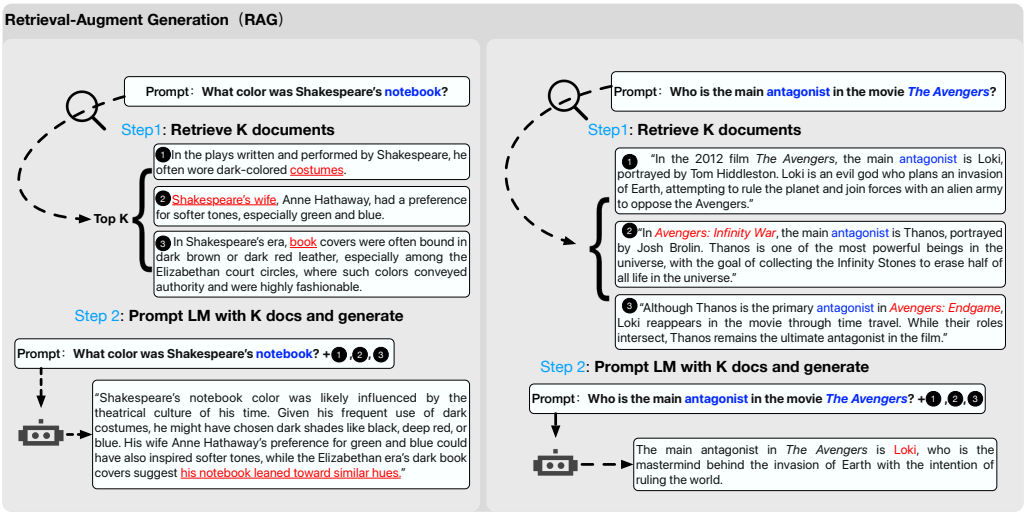

Figure 1. Two Challenges Faced by the RAG Framework: i) Left: The inference process is disturbed by irrelevant knowledge in the retrieved results. ii) Right: The complex associations among the knowledge in the retrieved results cannot be analyzed and understood.

However, current research on RAG methods still faces two issues. Issue 1: At the reasoning level, existing reasoning structures do not support modeling the reasoning processes and thought transformations for complex queries. Current studies on decomposing user queries based on reasoning structures often explain retrieval intent and construct complex associations between knowledge retrieval and reasoning processes by decomposing the reasoning process of complex queries into several independent linear reasoning paths. However, the reasoning for complex queries is often accomplished collaboratively by multiple available reasoning paths, involving complex thought transformations such as switching and backtracking among these paths [19]. The modeling of reasoning processes using independent linear reasoning paths is incomplete, which limits the ability of the RAG framework to dynamically adjust reasoning strategies when knowledge is insufficient, leading to factual hallucinations in LLMs. Issue 2: At the knowledge level, the processes of knowledge retrieval, utilization, and answer generation are not aligned with the reasoning process. Current research on planning knowledge usage within the RAG framework often relies on pre-constructed knowledge structures or trained strategy evaluation modules to plan and adjust strategies for knowledge retrieval, utilization, and generation. Pre-constructed knowledge structures and training data cannot be

 doi:10.20944/preprints202411.1648.v1

dynamically adjusted according to different reasoning processes, resulting in the independence of knowledge retrieval, utilization, and answer generation from the reasoning process, makes it difficult to effectively support the reasoning process, ultimately leading to factual hallucinations of LLMs caused by misuse of knowledge during the reasoning process.

Addressing the aforementioned issues, our work aims to achieve the following three objectives: 1) Design and utilize a complex reasoning structure based on the RAG method to model the reasoning process for complex queries in RAG. 2) Guide the knowledge retrieval and utilization of the RAG method based on this complex reasoning structure. 3) Evaluate the effectiveness of knowledge within the reasoning process based on the complex reasoning structure, and dynamically adjust the reasoning strategy of the RAG method to select reasoning paths supported by valid knowledge. Consequently, we propose the CRP-RAG framework, which supports complex reasoning and knowledge planning. This framework models the complex reasoning process of queries through a reasoning graph and utilizes this graph to guide knowledge retrieval, utilization planning, and reasoning strategy adjustment in the RAG method. The CRP-RAG framework consists of three modules: Reasoning Graph Construction (GC), Knowledge Retrieval and Aggregation (KRA), and Answer Generation (AG). Firstly, the GC module constructs a reasoning graph to represent the relationships between reasoning paths more comprehensively and flexibly, supporting complex thought transformations during the reasoning process. 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. Finally, the AG module evaluates the effectiveness of knowledge in the reasoning process at the level of reasoning graph nodes and selects reasoning paths with valid knowledge to guide LLMs in reasoning and answer generation. Experimental results demonstrate that our method significantly outperforms current strong baselines for tasks such as open-domain question answering, multi-hop reasoning question answering, and fact verification. Additionally, our method exhibits excellent noise resistance and factual consistency.

We summarize our contributions as follows:

(1) We propose, for the first time, a comprehensive modeling of the RAG reasoning process based on a reasoning graph, to support more sophisticated thought transformations.

(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, empirically validating the guiding significance of the reasoning graph in the knowledge retrieval, knowledge aggregation, and answer generation processes within the RAG framework.

2. Related Work

2.1. Retrieval Augmented Generation Based on 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 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

 doi:10.20944/preprints202411.1648.v1

knowledge. Text templates distill and summarize knowledge collections through 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]. Some work further fine-tuning models based on text-template instructions to 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 [24–26]. Additionally, some studies design specific knowledge structures tailored to specific generation task goals to improve the performance of RAG in those tasks. CANDLE [27] extends existing knowledge bases by constructing concepts and instances of existing knowledge and establishes 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 [28] and ARM-RAG [29] 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 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 by 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 improper use of knowledge by LLMs.

2.2. Retrieval Augmented Generation Based on Question 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, question 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 question 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 [14,30]. 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 [31,32] or refine reconstructive results through iterative question reformulation [33]. On the other hand, expansive decomposition decomposes queries into several sub-queries to expand the retrieval solution space based on specific decomposition rules [34,35] or structures [36]. 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 question decomposition methods often rely on LLMs to perform decomposition and reconstruction based on prompts. The implicit reasoning of LLMs may pose two issues: 1) The decomposition and reconstruction of queries by LLMs are independent of the reasoning process,

 doi:10.20944/preprints202411.1648.v1

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 [39]. Therefore, planning for knowledge retrieval and utilization, 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 [17,18,37,38] and the effectiveness of retrieval results [39,40], 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 factual consistency [18,40] and accuracy [41,42] 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 will 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 [43] 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) [11,44], CoT-SC [45] 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) [12] constructs the reasoning process as a tree, combining linear reasoning paths into a traceable multi-linear reasoning structure, further improving the representation ability of

 doi:10.20944/preprints202411.1648.v1

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 1.

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 in Equation 1.

$$G = GC(q) \tag{1}$$

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 in Equation 2.

$$K = KRA(V, q) \tag{2}$$

$k_i \in K$ represents the relevant knowledge obtained after knowledge retrieval and aggregation for the corresponding reasoning graph node v_i, and serves as 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.

$$a = AG(K, V, q) \tag{3}$$

 doi:10.20944/preprints202411.1648.v1

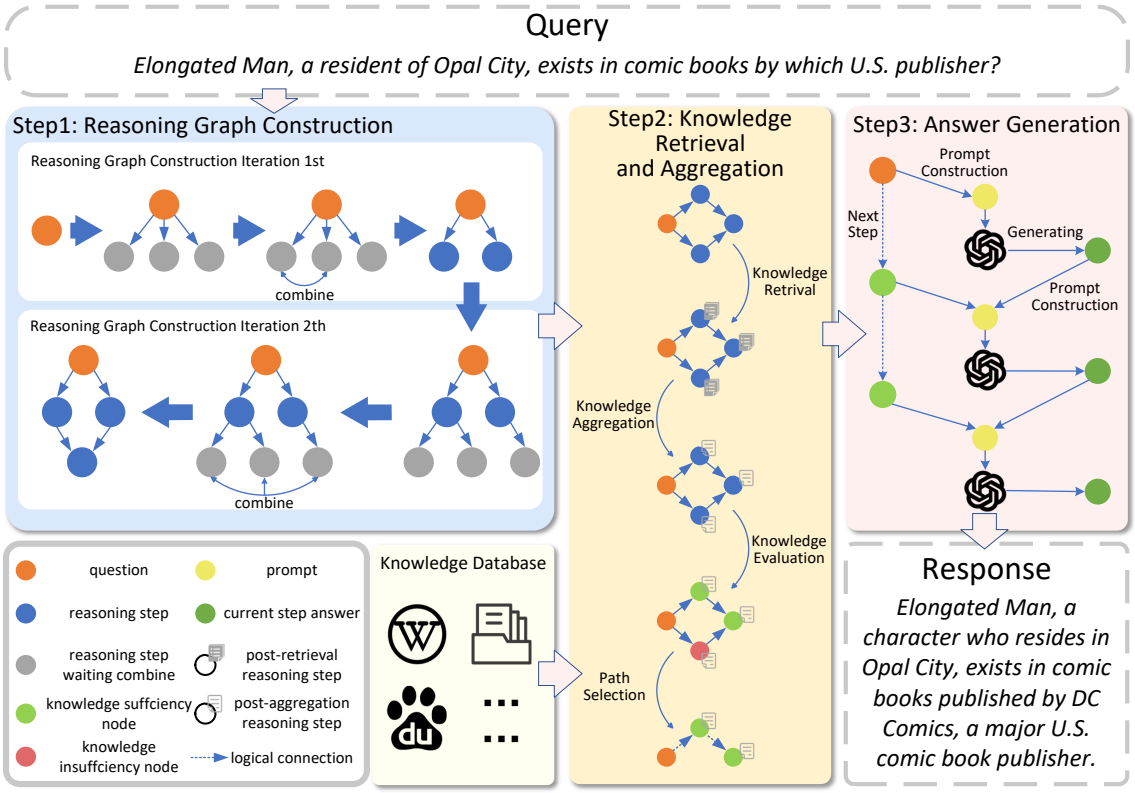

Figure 2. Overview of CRP-RAG framework. The CRP-RAG consists of three modules: i) The GC module constructs the reasoning graph based on the query. ii) The KRA module performs knowledge retrieval and aggregation based on the nodes of the reasoning graph. iii) The AG module generates a query-based answer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

3.2. 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 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 in Equation 4.

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

$V_n^{new}(i)$ 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, $V_n^{new} = [V_n^{new}(1), V_n^{new}(2), ..., V_n^{new}(k)]$, where the length of the 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.

 doi:10.20944/preprints202411.1648.v1

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 $V_n^{new} = [v_n^{new}(1), v_n^{new}(2), ..., v_n^{new}(m)]$, 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).

$$similarity = enc\left(v_n^{new}\left(i\right)\right) \bigodot enc\left(v_n^{new}\left(j\right)\right) \tag{5}$$

$$v_n^{combine}\left(i\right) = LLM\left(prompt_{combine}, v_n^{new}\left(i\right), v_n^{new}\left(j\right), q\right) \; if \; similarity > thereshold_{combine} \tag{6}$$

The $enc()$ function semantically encodes the new nodes based on a language model. The similarity 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. $v_n^{combine}\left(i\right)$ is the merged node resulting from the combination of $v_n^{new}\left(i\right)$ and $v_n^{new}\left(j\right)$, 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 $thereshold_{combine}$ 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 corresponds 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 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 $D = [d_1, d_2, ..., d_n]$. 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.

$$d_i = top_k\left(similarity\left(v_i, r_j\right)\right) \; i \in 0, 1, 2, \ldots, |V| - 1 \; , j \in 0, 1, 2, \ldots, |R| - 1 \tag{7}$$

Here, $similarity()$ is the semantic similarity calculation function defined in Equation 5. The function $\text{top}_k()$ 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 will perform knowledge refinement and aggregation on all retrieval result sets in D, forming an aggregated knowledge set $K = [k_1, k_2, ..., k_n]$. Each $k_i \in K$ is obtained from d_i through knowledge aggregation. Knowledge refinement and aggregation are achieved by LLMs that

 doi:10.20944/preprints202411.1648.v1

generate knowledge summaries for the relevant documents in d_i. The formulation for this process is shown in Equation 8.

$$k_i = LLM\left(prompt_{intergration}, d_i, v_i\right) \tag{8}$$

$prompt_{integration}$ 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 $C = [c_1, c_2, \ldots, c_p]$ composed of knowledge-sufficient nodes for reasoning and answer generation. Each $c_i = [s_1, s_2, \ldots, s_p]$ 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 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 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.

$$score = perplexity\left(LLM\left(prompt_{reasoning}, v_i, k_i\right)\right) \tag{9}$$

$$v_i \in \begin{cases} V_{sufficient}, score < thereshold_{perplexity} \\ V_{insufficient}, score \geq thereshold_{perplexity} \end{cases} \tag{10}$$

$thereshold_{perplexity}$ is a hyperparameter that represents the threshold for perplexity.

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 $c_i = [s_1, s_2, \ldots, s_p]$ in the set satisfy three conditions: 1) s_1 is a source node in G; 2) s_p is a sink node in G; 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.

$$result_n = LLM\left(condition_n, s_n, k_n\right) \tag{11}$$

$$condition_{n+1} = concate\left(condition_n, result_n\right) \tag{12}$$

The $concate()$ 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

 doi:10.20944/preprints202411.1648.v1

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.

$$answer = LLM\left(prompt_{abstract}, q, A\right) \tag{13}$$

$A = a_1, a_2, ..., a_m$ 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 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 three typical datasets: 1) Natural Questions (NQ) [46], 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 [47] 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) [48], 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 two typical datasets: 1) HotpotQA [49] introduces the concept of cross-document information integration for complex questions 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[50] 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 [51], 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

 doi:10.20944/preprints202411.1648.v1

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.

RALMs 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 RALMs frameworks. **1) The Vanilla RALMs Framework** aligns with the RAG method but replaces the generative language model with LLMs to enhance reasoning and knowledge planning. **2) The Query Decomposition RALMs Framework** decomposes queries into sub-queries before knowledge retrieval to better represent the retrieval needs of user queries. This study chooses IRCoT [52] and ITER-REGEN [53] as baselines for question decomposition-based RALMs, both of which use iterative retrieval to expand query information and improve the quality of retrieval results. **3) The Knowledge Structure RALMs Framework** models complex knowledge relationships in the knowledge base by designing special knowledge structures and prompts LLMs with knowledge associations between retrieval results through context. This study selects RAPTOR and GraphRAG 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 RALMs 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 RALMs framework for reasonable knowledge retrieval and answer generation. This study chooses Think-then-Act and Self-RAG 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 three QA evaluation metrics: **1) 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) Acc-LMs score** assesses 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. For the fact verification task, which resembles a classification format, we use the Acc-LMs score to compare the gold answer with the model's classification results, evaluating the correctness of the classification.

 doi:10.20944/preprints202411.1648.v1

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 [54] as the generative LLM for CRP-RAG and all the baselines, using BGE-large-en [55] as the retrieval model and the Wikipedia knowledge base dump from April 2024 [56]. Due to the instruction sensitivity of LLMs, all baselines supporting external knowledge retrieval adopted a prompt-based knowledge fusion approach, retrieving top-5 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. The remaining experimental settings for baseline methods were consistent with their original papers.

 doi:10.20944/preprints202411.1648.v1

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 Varifing
	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	85.0	25.3	37.3	48.3	34.0	47.8	57.2	28.1	40.5	53.6	68.4
RALMs 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 RALMs 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	86.0	84.4	60.2	75.8	81.2	45.8	61.1	73.4	36.0	47.4	58.5	71.5
Knowledge Structure RALMs 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 RALMs 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	76.3	80.1	79.3	58.2	69.0	77.4	67.4	80.1	86.0	57.6	69.4	79.1	80.8
Ours																
CRP-RAG	**63.2**	**71.1**	**82.3**	**79.7**	**86.4**	**87.0**	**62.5**	**75.6**	**85.2**	**81.0**	**87.6**	**87.4**	**69.3**	**77.9**	**81.0**	**85.0**

 doi:10.20944/preprints202411.1648.v1

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 RALMs 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 RALMs 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 RALMs 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 RALMs in achieving knowledge retrieval based on the reasoning process, thereby enhancing their knowledge retrieval performance in complex reasoning. Moreover, CRP-RAG outperforms RALMs 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 RALMs 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 RALMs framework.

5. Discussion

This section presents 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); and reasoning graph structure evaluation experiments to assess the rationality of the reasoning structure utilized in CRP-RAG's reasoning graphs (Section 5.4).

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 5.4. The ablation experimental groups include: 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

 doi:10.20944/preprints202411.1648.v1

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.

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 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 widespread knowledge insufficiency of reasoning paths.

 doi:10.20944/preprints202411.1648.v1

Table 2. Ablation study result.

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

Based on the experimental results, we conclude that CRP-RAG exhibits salient robustness, manifested in two aspects as illustrated in Figure 3. Firstly, in scenarios with lesser interference, CRP-RAG discards distracted reasoning paths and select 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.

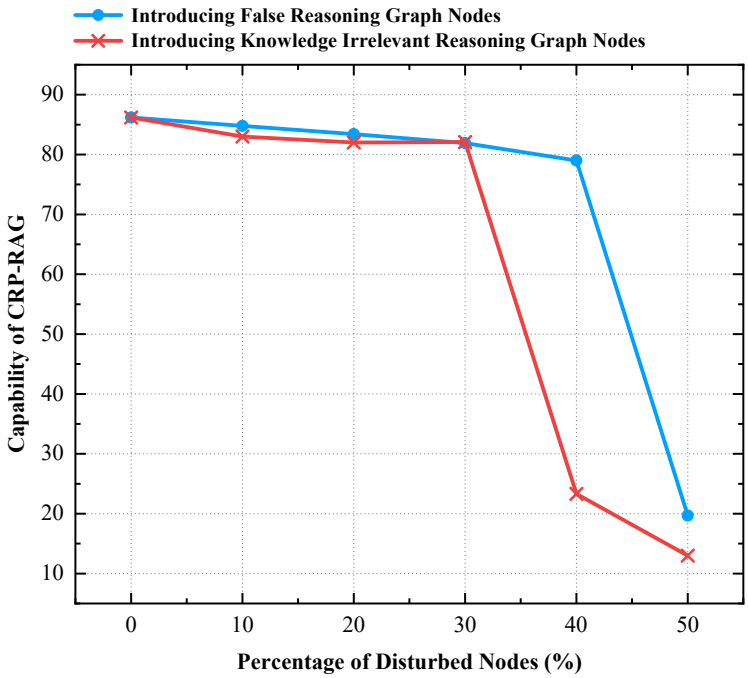

Figure 3. Robustness Analysis of CRP-RAG. The blue line represents the experimental results of introducing false reasoning graph nodes into CRP-RAG, while the red line indicates the experimental results of introducing knowledge-irrelevant reasoning graph nodes into CRP-RAG.

 doi:10.20944/preprints202411.1648.v1

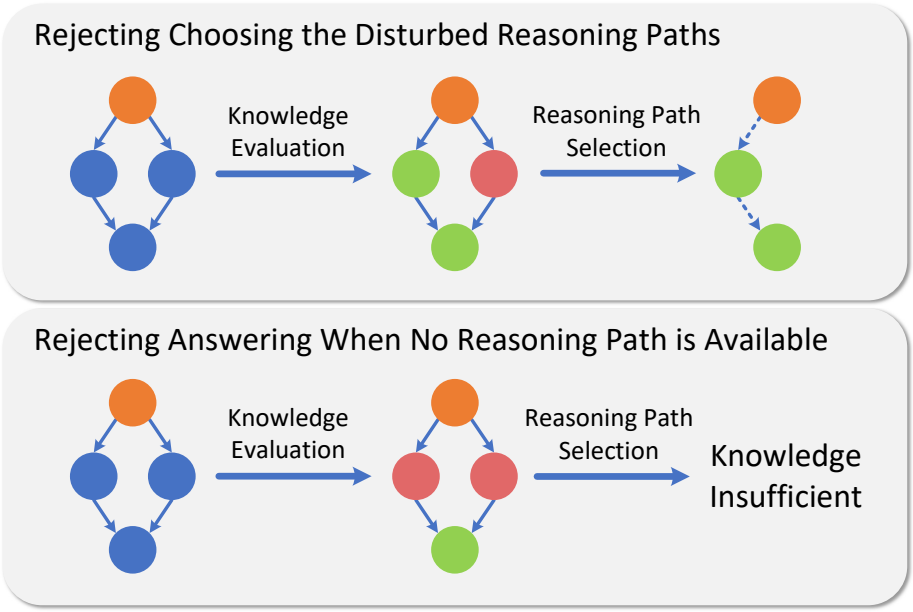

Figure 4. CRP-RAG discards distracted reasoning paths and abstains from answering when no valid reasoning path is available.

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 RALMs 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) phase. 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.

 doi:10.20944/preprints202411.1648.v1

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
	RALMs Framework	
RALMs	201.1	558.8
	Query Decomposition RALMs Framework	
IRCoT	208.2	608.6
ITER-RETGEN	593.9	1094.8
	Knowledge Structure RALMs Framework	
RAPTOR	124.0	477.5
GraphRAG	236.0	794.6
	Self-Planning RALMs Framework	
Think-then-Act	156.8	330.6
Self-RAG	112.0	116.5
	Ours	
CRP-RAG	**21.4**	**8.1**

5.4. 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 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-question-based text rewriting module, degrading CRP-RAG into a self-reflective RALMs framework relying on question rewriting and perplexity evaluation.

As shown in Table 5, CRP-RAG outperforms other reasoning structures on both datasets, with a more significant advantage when the reasoning structure is degraded to chains and texts.

 doi:10.20944/preprints202411.1648.v1

Table 4. Results of 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
	RALMs Framework	
RALMs	66.7	69.5
	Query Decomposition RALMs Framework	
IRCoT	72.1	75.5
ITER-RETGEN	74.2	75.6
	Knowledge Structure RALMs Framework	
RAPTOR	78.8	71.9
GraphRAG	79.0	73.4
	Self-Planning RALMs Framework	
Think-then-Act	81.9	80.7
Self-RAG	85.2	83.1
	Ours	
CRP-RAG	**92.5**	**91.8**

Table 5. 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

Analysis of generated samples reveals 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 questions, 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 RALMs 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.

 doi:10.20944/preprints202411.1648.v1

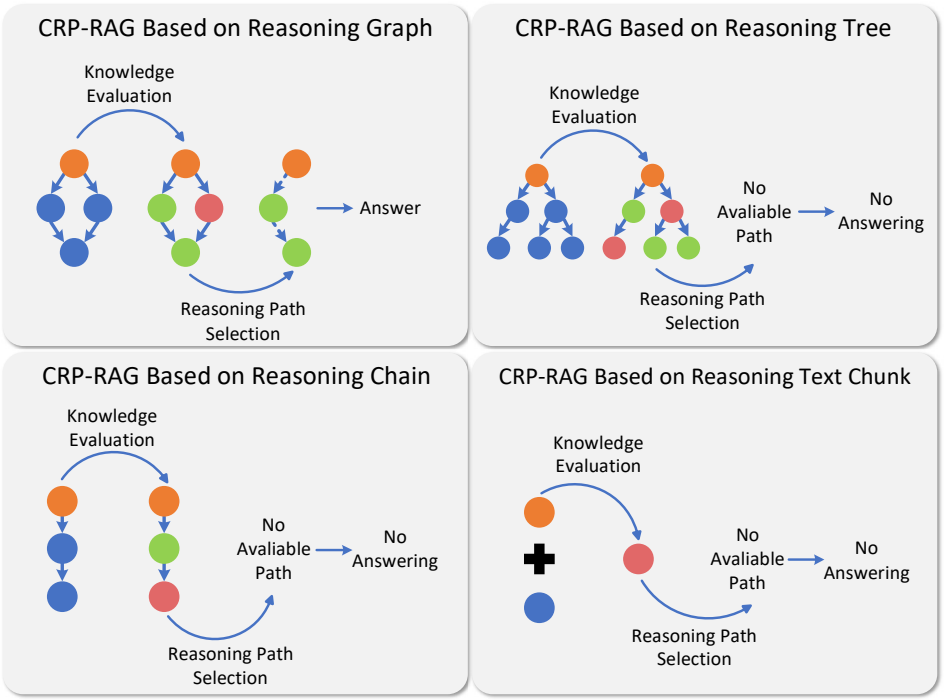

Figure 5. Impact of Different Reasoning Structures on CRP-RAG Behavior.

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.

Author Contributions: For research articles with several authors, a short paragraph specifying their individual contributions must be provided. The following statements should be used "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[46–51].

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

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. Requirements:

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.

 doi:10.20944/preprints202411.1648.v1

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 requirements for the content are:
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}

References

1. Brown, T.B. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165* **2020**. [Google Scholar].
2. Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H.W.; Sutton, C.; Gehrmann, S.; others. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research* **2023**, *24*, 1–113. [Google Scholar].

 doi:10.20944/preprints202411.1648.v1

3. Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; others. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* **2023**. [Google Scholar].
4. Bang, Y.; Cahyawijaya, S.; Lee, N.; Dai, W.; Su, D.; Wilie, B.; Lovenia, H.; Ji, Z.; Yu, T.; Chung, W.; others. A Multitask, Multilingual, Multimodal Evaluation of ChatGPT on Reasoning, Hallucination, and Interactivity. 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 (Volume 1: Long Papers), 2023, pp. 675–718. [Google Scholar].
5. Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; others. Training language models to follow instructions with human feedback. *Advances in neural information processing systems* **2022**, *35*, 27730–27744. [Google Scholar].
6. Huang, M.; Zhu, X.; Gao, J. Challenges in building intelligent open-domain dialog systems. *ACM Transactions on Information Systems (TOIS)* **2020**, *38*, 1–32. [Google Scholar].
7. Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.t.; Rocktäschel, T.; others. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems* **2020**, *33*, 9459–9474. [Google Scholar].
8. Guu, K.; Lee, K.; Tung, Z.; Pasupat, P.; Chang, M. Retrieval augmented language model pre-training. International conference on machine learning. PMLR, 2020, pp. 3929–3938. [Google Scholar].
9. Ram, O.; Levine, Y.; Dalmedigos, I.; Muhlgay, D.; Shashua, A.; Leyton-Brown, K.; Shoham, Y. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics* **2023**, *11*, 1316–1331. [Google Scholar].
10. 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. Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2024, pp. 719–729. [Google Scholar].
11. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q.V.; Zhou, D.; others. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems* **2022**, *35*, 24824–24837. [Google Scholar].
12. Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T.; Cao, Y.; Narasimhan, K. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems* **2024**, *36*. [Google Scholar].
13. Mao, Y.; He, P.; Liu, X.; Shen, Y.; Gao, J.; Han, J.; Chen, W. Generation-Augmented Retrieval for Open-Domain Question Answering. Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 2021, pp. 4089–4100. [Google Scholar].
14. Kim, M.; Park, C.; Baek, S. Augmenting Query and Passage for Retrieval-Augmented Generation using LLMs for Open-Domain Question Answering. *arXiv preprint arXiv:2406.14277* **2024**. [Google Scholar].
15. An, K.; Yang, F.; Li, L.; Lu, J.; Cheng, S.; Si, S.; Wang, L.; Zhao, P.; Cao, L.; Lin, Q.; others. Thread: A Logic-Based Data Organization Paradigm for How-To Question Answering with Retrieval Augmented Generation. *arXiv preprint arXiv:2406.13372* **2024**. [Google Scholar].
16. Sarthi, P.; Abdullah, S.; Tuli, A.; Khanna, S.; Goldie, A.; Manning, C.D. RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval. The Twelfth International Conference on Learning Representations. [Google Scholar].
17. Shen, Y.; Jiang, H.; Qu, H.; Zhao, J. Think-then-Act: A Dual-Angle Evaluated Retrieval-Augmented Generation. *arXiv preprint arXiv:2406.13050* **2024**. [Google Scholar].
18. Asai, A.; Wu, Z.; Wang, Y.; Sil, A.; Hajishirzi, H. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. The Twelfth International Conference on Learning Representations. [Google Scholar].
19. Besta, M.; Blach, N.; Kubicek, A.; Gerstenberger, R.; Podstawski, M.; Gianinazzi, L.; Gajda, J.; Lehmann, T.; Niewiadomski, H.; Nyczyk, P.; others. Graph of thoughts: Solving elaborate problems with large language models. Proceedings of the AAAI Conference on Artificial Intelligence, 2024, Vol. 38, pp. 17682–17690. [Google Scholar].
20. Xu, F.; Shi, W.; Choi, E. Recomp: Improving retrieval-augmented lms with compression and selective augmentation. *arXiv preprint arXiv:2310.04408* **2023**. [Google Scholar].

 doi:10.20944/preprints202411.1648.v1

21. Jin, J.; Zhu, Y.; Zhou, Y.; Dou, Z. BIDER: Bridging Knowledge Inconsistency for Efficient Retrieval-Augmented LLMs via Key Supporting Evidence. *arXiv preprint arXiv:2402.12174* **2024**. [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 preprint arXiv:2405.16420* **2024**. [Google Scholar].
23. Goel, K.; Chandak, M. HIRO: Hierarchical Information Retrieval Optimization. *arXiv preprint arXiv:2406.09979* **2024**. [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 preprint arXiv:2402.07630* **2024**. [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. Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2024, pp. 2905–2909. [Google Scholar].
26. 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 preprint arXiv:2408.15903* **2024**. [Google Scholar].
27. Wang, W.; Fang, T.; Li, C.; Shi, H.; Ding, W.; Xu, B.; Wang, Z.; Bai, J.; Liu, X.; Cheng, J.; others. CANDLE: iterative conceptualization and instantiation distillation from large language models for commonsense reasoning. *arXiv preprint arXiv:2401.07286* **2024**. [Google Scholar].
28. 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 preprint arXiv:2406.04271* **2024**. [Google Scholar].
29. Melz, E. Enhancing llm intelligence with arm-rag: Auxiliary rationale memory for retrieval augmented generation. *arXiv preprint arXiv:2311.04177* **2023**. [Google Scholar].
30. 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 preprint arXiv:2404.14043* **2024**. [Google Scholar].
31. 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 preprint arXiv:2402.03620* **2024**. [Google Scholar].
32. 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. *Computational Intelligence* **2024**, *40*, e12654. [Google Scholar].
33. 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 preprint arXiv:2305.07402* **2023**. [Google Scholar].
34. 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. Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2024, pp. 7339–7353. [Google Scholar].
35. Yoran, O.; Wolfson, T.; Ram, O.; Berant, J. Making Retrieval-Augmented Language Models Robust to Irrelevant Context. The Twelfth International Conference on Learning Representations. [Google Scholar].
36. 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. Proceedings of the AAAI Conference on Artificial Intelligence, 2024, Vol. 38, pp. 19560–19568. [Google Scholar].
37. 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 preprint arXiv:2402.10612* **2024**. [Google Scholar].
38. 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. *CoRR* **2024**. [Google Scholar].
39. Yan, S.Q.; Gu, J.C.; Zhu, Y.; Ling, Z.H. Corrective retrieval augmented generation. *arXiv preprint arXiv:2401.15884* **2024**. [Google Scholar].
40. 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 preprint arXiv:2403.06840* **2024**. [Google Scholar].

41. 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. The Twelfth International Conference on Learning Representations. [Google Scholar].
42. 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. Findings of the Association for Computational Linguistics: EMNLP 2024, 2024, pp. 10371–10393. [Google Scholar].
43. Kojima, T.; Gu, S.S.; Reid, M.; Matsuo, Y.; Iwasawa, Y. Large language models are zero-shot reasoners. *Advances in neural information processing systems* **2022**, *35*, 22199–22213. [Google Scholar].
44. Zhang, Z.; Zhang, A.; Li, M.; Smola, A. Automatic Chain of Thought Prompting in Large Language Models. The Eleventh International Conference on Learning Representations. [Google Scholar].
45. 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. The Eleventh International Conference on Learning Representations. [Google Scholar].
46. Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; others. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics* **2019**, *7*, 453–466. [Google Scholar].
47. Joshi, M.; Choi, E.; Weld, D.S.; Zettlemoyer, L. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2017, pp. 1601–1611. [Google Scholar].
48. Berant, J.; Chou, A.; Frostig, R.; Liang, P. Semantic parsing on freebase from question-answer pairs. Proceedings of the 2013 conference on empirical methods in natural language processing, 2013, pp. 1533–1544. [Google Scholar].
49. 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. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018, pp. 2369–2380. [Google Scholar].
50. Ho, X.; Nguyen, A.K.D.; Sugawara, S.; Aizawa, A. Constructing A Multi-hop QA Dataset for Comprehensive Evaluation of Reasoning Steps. Proceedings of the 28th International Conference on Computational Linguistics, 2020, pp. 6609–6625. [Google Scholar].
51. Thorne, J.; Vlachos, A.; Christodoulopoulos, C.; Mittal, A. FEVER: a Large-scale Dataset for Fact Extraction and VERification. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), 2018, pp. 809–819. [Google Scholar].
52. Trivedi, H.; Balasubramanian, N.; Khot, T.; Sabharwal, A. Interleaving Retrieval with Chain-of-Thought Reasoning for Knowledge-Intensive Multi-Step Questions. Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2023, pp. 10014–10037. [Google Scholar].
53. Shao, Z.; Gong, Y.; Shen, Y.; Huang, M.; Duan, N.; Chen, W. Enhancing Retrieval-Augmented Large Language Models with Iterative Retrieval-Generation Synergy. Findings of the Association for Computational Linguistics: EMNLP 2023, 2023, pp. 9248–9274. [Google Scholar].
54. GLM, T.; Zeng, A.; Xu, B.; Wang, B.; Zhang, C.; Yin, D.; Zhang, D.; Rojas, D.; Feng, G.; Zhao, H.; others. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793* **2024**. [Google Scholar].
55. Xiao, S.; Liu, Z.; Zhang, P.; Muennighof, N. C-pack: Packaged resources to advance general chinese embedding. *arXiv preprint arXiv:2309.07597* **2023**. [Google Scholar].
56. Merity, S.; Xiong, C.; Bradbury, J.; Socher, R. Pointer Sentinel Mixture Models. International Conference on Learning Representations, 2022. [Google Scholar].