Role-of-Thought: Enhancing Reasoning Capability Through Implicit Knowledge Extraction via Role-Oriented Multi-Agent Collaboration

Daojian Zeng[a,b], Penghao Li[b], Tianle Li[b], Shanshan Li[b], Tingjiao Huang[b], Lincheng Jiang[†c]

[a] *Institute of AI and Targeted International, Communication, Hunan Normal University, Changsha, 410081, China*
[b] *Hunan Provincial Key Laboratory of Intelligent Computing and Language Information Processing, Hunan Normal University, Changsha, 410081, China*
[c] *College of Advanced Interdisciplinary Studies, National University of Defense Technology, Changsha, 410073, China*

Abstract

Large Language Models (LLMs) have shown remarkable performance in Natural Language Processing (NLP) tasks, but they still face significant challenges in complex reasoning tasks due to their limitations in deeply analyzing implicit knowledge and logic. However, existing Multi-Agent System (MAS) may produce redundant or conflicting information during parallel reasoning, reducing response quality and masking key insights. To address these issues, we propose Role-of-Thought (RoT), an adaptive MAS that dynamically generates domain-specific expert agents to enhance reasoning capabilities. RoT divides complex tasks into multi-stage sub-tasks, assigns them to relevant expert agents, and integrates the reasoning processes into a coherent reasoning chain. By leveraging the strengths of multiple expert agents and multi-stage reasoning, RoT effectively extracts and applies implicit knowledge, improving the accuracy and adaptability of reasoning. Experimental results demonstrate that RoT significantly outperforms baseline reasoning methods, showing enhanced reasoning performance in various tasks.

Keywords: Large Language Models; Prompting Engineering; Multi-Agent System; Reasoning Ability; Implicit Knowledge Extraction

1. Introduction

Large Language Models (LLMs) [1] have performed exceptionally well in the traditional field of Natural Language Processing (NLP) [2, 3, 4] through pre-training on vast general corpora. However, LLMs are more inclined to capture superficial language patterns, which restricts their capacity to deeply analyze implicit knowledge, understood as unarticulated and context-dependent associations or domain regularities. Such knowledge often does not directly surface in the text and is closely intertwined with the underlying logic [5]. These limitations impair LLMs' reasoning abilities and cause hallucinations during reasoning tasks [6]. To address this limitation, research is now focusing on ways to improve LLMs' capacities for reasoning.

Researchers have explored two main approaches to enhance the reasoning capabilities of LLMs. (1) Post-training [7, 8] for additional training of LLMs on specific tasks or datasets.

[†]Corresponding author: Lincheng Jiang (Email: linchengjiang@nudt.edu.cn; ORCID: 0009-0005-7825-0258)

Method like Generalized Reinforcement Policy Optimization (GRPO) [9] utilize data from areas with high reasoning requirements, such as code and mathematics, to refine pre-trained LLMs. However, post-training typically requires substantial computer resources, and obtaining effective training data is difficult. (2) Prompt engineering can maximize the potential of LLMs by designing appropriate input prompts and generating output that meets the requirements. Following Chain of Thought (CoT) paradigm [10], some methods [11, 12] encourage the model to generate a sequential reasoning chain, thereby enhancing the reliability of its reasoning process. Compared to post-training, prompt engineering can flexibly stimulate the model's reasoning potential without requiring additional training data or computational resources.

Building on prompt engineering, agents based on LLM can deeply consider reasoning tasks [13, 14]. ExpertPrompting [15] identifies LLM as a task-related expert agent and activates the agent's identity to respond. However, a single agent role may lead to bias and limit exploration of other perspectives [16]. To overcome this constraint, Multi-Agent System (MAS) [17] provide diverse viewpoints by assigning distinct role identities to each agent [18, 19]. These viewpoints aim to broaden the reasoning capabilities of LLMs. Multi-Agent Debate (MAD) [20] and CoT-Rerailer [21] leverage multiple agents to review and revise reasoning chains from various perspectives. Furthermore, Iteration-of-Thought [22] and Multi-Expert Prompting [16] promote collaborative reasoning through a structured process that supports task allocation among agents. However, when all agents engage in the reasoning process at the same time, important knowledge is frequently obscured by unimportant details [23]. This phenomenon not only reduces response quality but also leads MAS to prioritize consensus-driven responses over accurate ones.

To address these challenges, we propose Role-of-Thought (RoT), a role-oriented MAS. The core idea of RoT is to dynamically generate expert agent roles with multiple identities related to specific reasoning tasks. To facilitate effective information exchange, these roles are assigned to corresponding sub-tasks and participate in multi-stage reasoning. RoT consists of four modules: (1) **Expert Agent Generation**, (2) **Opinion Expression**, (3) **Task Allocation**, and (4) **Multi-Stage Reasoning**. First, RoT flexibly analyzes reasoning tasks from multiple perspectives by dynamically generating expert agents in different fields. Next, these agents selectively retrieve relevant task-related knowledge and express opinions aligned corresponding to their identity. Based on the opinions of expert agents, the reasoning process is divided into interconnected multi-stage sub-tasks, each handled by the expert agent in the corresponding field. Finally, upon completing these sub-tasks, expert agents synthesize their contributions to generate a comprehensive reasoning chain. By dynamically generating expert agents, reasoning tasks can be analyzed from multiple perspectives, producing responses with relevant information. These responses often contain implicit knowledge not reflected in the task. Task Allocation and Multi-Stage Reasoning module ensure that implicit knowledge is not overshadowed by irrelevant information and that consensus-driven responses do not occur. In summary, our main contributions are as follows.

1. We propose RoT, a MAS that simulates experts in various fields to enhance LLMs' performance in reasoning tasks by mining implicit knowledge.
2. We introduce a Multi-Stage Reasoning module, which decomposes complex reasoning tasks into sub-tasks, ensuring that RoT does not produce consensus-driven responses during the reasoning process.
3. Through extensive experiments, we demonstrate that RoT significantly outperforms baseline reasoning methods.

2. Related Work

In this section, we discuss the foundational concepts related to our work, including post-training techniques, chain-of-thought reasoning, and multi-agent system for reasoning.

2.1. Post-training

Post-training methodologies aim to enhance the reasoning capabilities of pre-trained LLMs by fine-tuning their parameters and architectures to better suit specific tasks. For example, Fine-tuning [11] involves training LLMs on task-specific labeled data to learn relevant patterns and rules. Studies by Luo et al. [24] and Trung et al. [25] demonstrate that fine-tuning on mathematical domain datasets substantially improves LLMs' problem-solving abilities in mathematics. However, as noted by Kalajdzievski et al. [26] and Liu et al. [27], fine-tuning may result in catastrophic forgetting, which can reduce the models' capacity for generalization. Reinforcement Learning from Human Feedback (RLHF) [8] has become a prominent post-training approach that aligns LLMs with human preferences and values. Nevertheless, RLHF is confronted with challenges such as the complexity of training reward models and instability during reinforcement learning. Direct Preference Optimization (DPO) [7] addresses some of these issues by incorporating preference data directly into the loss function, enabling more efficient learning of reasoning paths aligned with human expectations. Similarly, DeepSeek-R1 [28] utilizes Generalized Reinforcement Policy Optimization (GRPO) [9] to sample outputs from real-time policy models, significantly enhancing reasoning performance while reducing memory consumption. Despite these advancements, post-training methods often require substantial computational resources and large volumes of high-quality labeled data, which remain significant challenges for their widespread application.

2.2. Chain-of-Thought Reasoning

Prompt engineering guides LLMs to perform specific tasks without modifying their parameters. Wei et al. [10] introduced chain-of-thought (CoT) reasoning, which highlights the importance of multi-step logical processes to reach definitive answers. Building on this, Kojima et al. [29] discovered that adding the phrase "let's think step by step" to prompts enables LLMs to perform zero-shot logical reasoning without human-provided examples. Self-consistency (SC) [11] improves reasoning by sampling multiple reasoning paths and selecting the most consistent answer. Tree of Thought (ToT) [14] builds on CoT by integrating diverse reasoning paths to enhance decision-making. Re-reading [30] further deepens model understanding during reasoning by encouraging iterative comprehension. Additionally, methods such as self-refine [31] and self-verification [32] enable LLMs to iteratively critique and improve their outputs. Beyond prompt-based techniques, fine-tuning LLMS on datasets containing CoT responses presents another path to enhance CoT reasoning abilities. OpenAI-o1 [33] model further advances CoT methods by feeding both reasoning steps and predictions back into the model, enabling multiple iterative processing rounds. Overall, CoT-based reasoning techniques guide models to generate more detailed intermediate steps, thereby improving overall performance. However, when tackling reasoning tasks requiring broad knowledge, a single CoT approach may struggle to address diverse perspectives, and exhaustive exploration of reasoning paths can lead to inefficient use of computational resources.

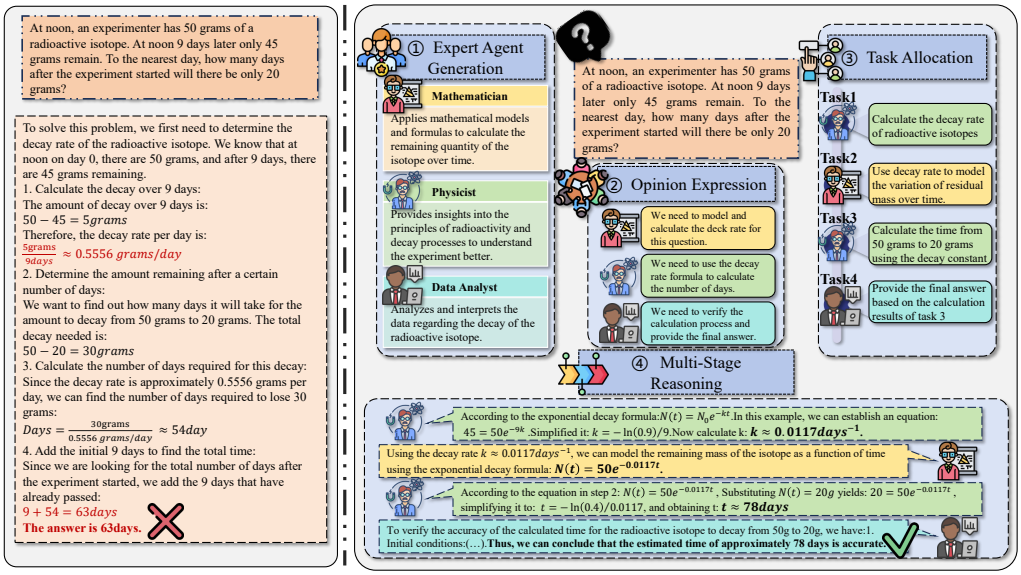


Figure 1. The left figure shows a single thought chain incorrectly applying implicit knowledge in the reasoning process. The right figure shows the overall framework of Role of Thought, including four major modules: **(1) Expert Agent Generation**; **(2) Opinion Expression**; **(3) Task Allocation**; **(4) Multi-Stage Reasoning**. These modules work in concert to achieve precise solutions for reasoning tasks.

2.3. Multi-agent System with Reasoning

MAS utilizes multiple LLMs as distinct agents that collaboratively reason about a given task, thereby enhancing both the depth and breadth of the reasoning process. Multi-Agent Debate (MAD) [20] improves the stability of LLMs outputs by structuring agents into proponents and opponents to engage in organized debates. Building on MAD, Liu et al. [34] introduced group debates among intelligent agents embodying different professional roles. Their findings suggest that assigning distinct identities to agents in MAS enhances the diversity of perspectives among agents. Such role-playing mechanisms improve LLMs' reasoning performance by encouraging agents to adopt specialized viewpoints. Wang et al. [18] proposed Solo Performance Prompting (SPP), a method that adaptively generates diverse roles to boost the LLMs multi-turn collaborative reasoning capabilities. Similarly, the multi-expert prompt [16] generates multiple expert agents tailored to the task, leveraging their similarities and differences to steer the reasoning process. However, these approaches often overlook the fact that different professions may hold divergent perspectives on the same task. At the same time, agents in the collaboration process often tend to converge on a common understanding, leading to unique but correct viewpoints being overlooked. To solve the reasoning task, MAS must accurately mine implicit knowledge and reduce the influence of incorrect information during cooperation.

3. Methodology

To effectively solve complex reasoning tasks, it is necessary to correctly invoke relevant knowledge and apply it at the appropriate time. As shown in Figure 1, we propose RoT, a MAS for solving reasoning tasks, which consists of four modules: (1) **Expert Agent Generation.**

Multiple expert agents related to the query are generated based on the reasoning query; (2) **Opinion Expression.** Each expert agent expresses unique opinions based on the query, laying the foundation for subsequent task decomposition; (3) **Task Allocation.** The query is divided into multi-stage sub-tasks and allocated to the appropriate expert agent based on their opinions; (4) **Multi-Stage Reasoning.** Each expert agent solves the corresponding sub-task, and the reasoning processes of all sub-tasks are ultimately integrated to form the final reasoning path. Throughout this process, LLMs extracts implicit knowledge and applies it appropriately, thereby more effectively exploring and solving the task. The complete RoT process is shown in the pseudo-code Algorithm 1 below.

Algorithm 1 Role-of-Thought (RoT)

Input: query Q, base LLMs L
Output: answer A
agents ← Expert(Q, L) ▷ Generating expert agent team according to query
opinions ← expression(agents, Q, L) ▷ Expert agents express their opinions
sub-tasks ← Allocate(agents, Q, opinions, L) ▷ Allocate sub-tasks
n ← len(sub-tasks) ▷ Calculate the number of sub-tasks
Initialize process
for $i = 1$ **to** n **do**
    if $i == 1$ **then**
        step ← Reason(sub-tasks[i], L) ▷ Reason the first sub-task
    else
        step ← Reason(sub-tasks[i], process, L) ▷ Combine the previous reasoning process to reason the sub-tasks
    end if
    process ← process + step ▷ Add each stage to the reasoning process
end for
answer ← summary(process, L) ▷ Get final answer
return answer

3.1. Expert Agent Generation

When confronted with a complex reasoning task, people typically analyze tasks based on related fields and attempt to use corresponding knowledge. This module analyzes the domain involved in the query and generates and assigns a unique expert identity to each agent. These expert agents' identities are dynamically tailored to task-specific domains, enabling flexible adjustments in Opinion Expression and sub-task completion.

Generating the identities of expert agents is a crucial part of this module. We summarize it into three aspects: discernibility, informativeness, and automation. First, the description must be tailored according to specific instructions to ensure that the conceived expert agent has the most relevant background and experience for the task domain. Second, the description should be detailed and comprehensive, covering all necessary information about the expert agent to guide LLM's behavior as expected. Third, considering the inefficiency and impracticality of manual annotation, the generation of descriptions must be automated. RoT can accurately match the analytical perspective required by the task and stimulate LLM's related field knowledge. This

module makes the subsequent reasoning more targeted and avoids inference deviation caused by vague or inaccurate expert agents' identity.

To make the description of expert agents more accurate, we divide the Expert Agent Generation module into two parts: (1) generating the identities *id* of M expert agents; (2) generating the capabilities a of the expert agents. The formulas are defined as follows.

$$A(Q) = \{A_i \mid i = 1, 2, \ldots, M\} \tag{1}$$

$$A_i = \{id_i, a_i\} \tag{2}$$

$A(Q)$ represents the team of expert agents generated based on the query Q, A_i represents the i-th expert agent, and each expert agent is a tuple $\{id_i, a_i\}$. id_i is the identity of the expert agent A_i, and a_i is the capability description of the expert agent A_i, which includes the agent's analytical ability, knowledge base, and reasoning skills in a specific domain. Both id_i and a_i are determined based on the query Q. Prompts used in this module are detailed in Appendix B.1.

3.2. Opinion Expression

After the generation of expert agents, each agent must express a unique opinion based on its expertise domain regarding the task. These opinions can be either insights that facilitate reasoning by explicitly guiding subsequent steps or relevant implicit knowledge that indirectly supports the task by filling in unstated premises or background assumptions. They not only provide a multi-perspective view for the initial understanding of the task, but also lay the foundation for subsequent task decomposition and allocation. This module ensures that reasoning benefits from multiple professional perspectives, thereby avoiding the limitations of a single viewpoint.

We define the process of the Opinion Expression module as follows.

$$O(Q, A) = \{O_i \mid i = 1, 2, \ldots, M\} \tag{3}$$

$$O_i = P_{\text{opinion}}(Q, A_i) \tag{4}$$

$O(Q, A)$ represents the set of opinions expressed by the expert agent set A on the query Q, O_i represents the opinion expressed by the i-th expert agent A_i, and P_{opinion} denotes the prompt for generating opinions based on the query Q and expert agent A_i.

Compared with directly assigning tasks, this intermediate step allows agents to offer prioritized concerns, context-specific interpretations, or knowledge cues that can guide task decomposition. These preliminary perspectives reduce the likelihood of sub-task misallocation and improve the chances of correctly using implicit knowledge. Prompts used in this module are detailed in Appendix B.2.

3.3. Task Allocation

Following the expert agents' opinions, the Task Allocation module allocates the task to the appropriate agents in a reasonable manner after breaking it up into a number of related sub-tasks based on the agents' knowledge and opinions. The key to task allocation lies in matching sub-tasks to agents based on both their expertise and the contextual insights from Opinion Expression module, thereby improving the efficiency and accuracy of reasoning. Rational task allocation enables agents to leverage their domain-specific strengths and collaboratively solve complex tasks. At the same time, this module also avoids consensus-driven responses and effectively prevents key knowledge from being overshadowed by irrelevant information.

The Task Allocation module process can be represented as:

$$S = P_{\text{allocate}}(Q, A, O) = \{\text{step}_j \mid j = 1, 2, \ldots, N\} \tag{5}$$

$$\text{step}_j = \{\text{sub}_j, A_i\} \tag{6}$$

S represents the set of N allocated steps, and P_{allocate} denotes the prompt for Task Allocation based on the query Q, the team of expert agents A, and their opinions O. step_j represents the specific task allocated in the j-th stage, and each specific task is a tuple $\{sub_j, A_i\}$, where sub_j and A_i are the sub-task to be reasoned and the assigned expert agent for completing $step_j$, respectively. Prompts used in this module are detailed in Appendix B.3.

3.4. Multi-Stage Reasoning

In the Multi-Stage Reasoning module, each expert agent builds upon the reasoning content from the previous stages to conduct reasoning for the sub-task they are responsible for. After multiple stages of reasoning, the results from each stage are summarized to construct a complete reasoning path, thereby providing a comprehensive and accurate basis for solving the task. Summarizing multi-stage reasoning allows for verification and correction of the reasoning results from each sub-task, further enhancing the reliability and accuracy of the reasoning process.

The Multi-Stage Reasoning module can be formally represented as:

$$R_j = \sum_{j=1}^{N} RP_j \tag{7}$$

$$RP_j = P_{\text{reasoning}}(Q, R_{j-1}, \text{step}_j), \quad j = 1, 2, \ldots, N \tag{8}$$

$$\text{Answer} = P_{\text{summary}}(R_N) \tag{9}$$

R_j represents all the reasoning content up to the j-th stage, RP_j denotes the reasoning process for the j-th stage, $P_{\text{reasoning}}$ is the prompt for reasoning based on the query Q, the previous reasoning content R_{j-1}, and the specific task $step_j$ allocated for that stage, and P_{summary} is the prompt for summarizing all the reasoning content R_N to generate the final answer.

Summarizing the reasoning processes of each sub-task not only allows for verification and correction of the reasoning results at each stage but also constructs a complete reasoning path. A complete reasoning path clearly demonstrates the logic behind solving complex tasks, enhances the reliability and credibility of LLMs' reasoning, and makes the reasoning results more persuasive. Prompts used in this module are detailed in Appendix B.4 and B.5.

4. Experiments

4.1. Experiment Setting

In this study, we provide a comprehensive overview of the experiments conducted to demonstrate the effectiveness of RoT in handling reasoning tasks. This study introduces the experimental setup in detail, including the datasets used, evaluation metrics, and a thorough description of the baseline models. Through these experiments, we demonstrate that RoT can significantly enhance the ability of LLMs to solve complex reasoning tasks.

4.1.1. Dataset and Metrics

To demonstrate the accuracy and effectiveness of different methods, we use accuracy (ACC) as the evaluation metric. The experiments selected three representative knowledge reasoning tasks to assess our method: (1) **OpenBookQA** [35]. This dataset provides the core facts related to the query as a reference, but additional common sense knowledge is needed to answer the questions correctly. (2) **ARC-Challenge** [36]. Derived from real middle and high school science exams, this dataset requires the model to have stronger logical analysis and the ability to use relevant knowledge. (3) **MMLU** [37]. Covering 57 evaluation tasks ranging from elementary education to professional disciplines, this is currently the most comprehensive assessment set for general knowledge and higher-order cognitive abilities. In our experiments, we selected eight college tasks from MMLU that represent higher-order cognitive abilities: College computer science, College mathematics, College medicine, College physics, Computer security, Econometrics, Electrical engineering, and Formal logic. The settings of these three datasets cover the evaluation of LLMs' common sense reasoning, basic knowledge, and higher-order cognitive abilities, while all require the LLMs to acquire relevant knowledge and use it reasonably. The specific sample sizes for each dataset are provided in Appendix A.

4.1.2. Baseline

We selected five baseline models for comparison with RoT: (1) **Zero-shot**: This reflects the inherent reasoning capability of LLMs without any prompting. (2) **CoT** [10]: This demonstrates the logical reasoning and step-by-step reasoning ability of LLMs. (3) **CoT-SC** [11]: This method involves selecting the most consistent response from multiple responses generated by the LLMs. (4) **Self-refine** [31]: This incorporates the idea of self-reflection into reasoning to reduce the probability of intermediate errors. (5) **Multi-ExpertPrompt** [16]: This compares the ideas and viewpoints of multiple generated expert agents and integrates common opinions, differing opinions, and unique perspectives on this basis.

4.2. Main Result

To ensure comparability with the baselines, our experiments utilized both the open-source large model Mistral-7B-it v0.2 [38] and the closed-source large model GPT-3.5-turbo-0613 [39]. The number of expert agents in RoT was set to three. Table 1 presents the main results of our experiments.

Table 1. The ACC of Different Methods Across Three Datasets

Model	Method	OpenBookQA	ARC-Challenge	MMLU							
				College computer science	College mathematics	College medicine	College physics	Computer security	Econometrics	Electrical engineering	Formal logic
Mistral-7B-it v0.2	Zero-shot	28.80	56.91	33.00	23.00	48.83	20.79	49.00	35.20	29.20	40.28
	CoT	63.00	68.17	47.00	34.00	51.74	26.73	65.00	38.40	39.82	47.22
	CoT-SC	67.60	70.39	49.00	36.00	53.48	32.67	68.00	37.60	37.17	49.30
	Self-refine	32.80	57.25	36.00	23.00	41.86	24.75	52.00	30.40	32.74	40.97
	Multi-expert	51.40	53.77	34.00	34.00	45.46	24.75	53.00	36.40	27.43	37.50
	RoT(Our)	**74.20**	**75.60**	**65.00**	**37.00**	**61.27**	**50.98**	**76.00**	**43.85**	**62.76**	**53.97**
GPT-3.5-turbo-0613	Zero-shot	65.00	56.91	38.00	38.00	54.65	28.71	45.00	35.20	33.62	32.63
	CoT	79.20	68.17	48.00	33.00	62.79	37.62	77.00	34.40	41.59	55.55
	CoT-SC	78.00	70.39	50.00	37.00	63.95	35.64	76.00	39.20	41.59	56.25
	Self-refine	61.80	57.25	33.00	29.00	38.37	35.64	62.00	35.20	26.54	56.25
	Multi-expert	71.80	53.77	41.00	28.00	54.06	45.54	63.00	37.60	37.17	51.39
	RoT(Our)	**92.00**	**92.74**	**76.00**	**71.00**	**83.81**	**79.41**	**84.00**	**64.91**	**73.10**	**61.11**

4.3. Analysis

As shown in Table 1, the experimental results clearly demonstrate that RoT outperforms all baseline models on the OpenBookQA, ARC-Challenge, and MMLU datasets. Taking the GPT-3.5-turbo model as an example, compared to the best performing baselines, RoT achieved an increase of 12.80 percentage points in ACC on the OpenBookQA dataset and a 22.35 percentage point increase on the ARC-Challenge dataset. In addition to these overall improvements, RoT excels in higher-order cognitive abilities, particularly across the diverse academic fields covered by the MMLU dataset. Specifically, RoT saw a 19.86 percentage point increase in ACC on the "College Medicine" task, a 33.87 percentage point increase on the "College Physics" task, a 25.71 percentage point increase on the "Econometrics" task, and a 4.86 percentage point increase on the "Formal Logic" task.

Compared to other methods, RoT offers significant advantages. MAD assigns fixed roles to agents, which can lead to inflexibility and inaccuracies when agents operate outside their domain expertise. In contrast, RoT dynamically generates expert agents based on the specific requirements of each reasoning task. This dynamic generation ensures that the most suitable expert agents are involved, providing a more adaptable and precise approach to complex reasoning challenges. RoT's ability to dynamically instantiate roles enables better alignment with task-specific domain characteristics, resulting in more accurate and reliable outcomes.

Additionally, compared to Multi-Expert Prompting, RoT's Multi-Stage Reasoning module provides a significant edge. For instance, under the GPT-3.5-turbo model, RoT outperforms Multi-Expert Prompting by 20.20 percentage points on the OpenBookQA dataset and by 38.97 percentage points on the ARC-Challenge dataset. Multi-Expert Prompting often suffers from the issue of key knowledge being overshadowed by irrelevant information during the communication process among agents. This can result in lower response quality and a tendency towards consensus-driven rather than precise responses. RoT addresses this challenge by decomposing reasoning tasks into multiple stages and assigning them to expert agents with different domain expertise. This approach ensures that each sub-task is thoroughly analyzed in a targeted manner, effectively avoiding the dilution of critical information. Consequently, RoT enhances the stability and accuracy of the reasoning process, making it more robust and reliable.

RoT dynamically generates domain-specific expert agents with relevant professional backgrounds and capabilities based on the characteristics of each task. These agents then analyze the tasks from multiple perspectives, thereby significantly improving the accuracy and reliability of LLMs' knowledge application. Moreover, the flexible decomposition of reasoning tasks through the Multi-Stage Reasoning module is a key factor in its performance improvement. By decomposing reasoning tasks into multiple stages and assigning them to expert agents with different domain expertise, each sub-task can be thoroughly analyzed in a targeted manner. This effectively avoids knowledge gaps and logical discontinuities that may occur in a single LLM's reasoning process.

In summary, RoT demonstrates significant advantages and potential in complex reasoning tasks through its dynamic generation of domain-specific expert agents and the Multi-Stage Reasoning module. These innovations highlight RoT's critical value in improving reasoning accuracy, enhancing reasoning process transparency, and adapting to different scenarios.

5. Ablation Studies

We further explored the impact of several key factors on the knowledge reasoning capabilities of RoT, including the number of expert agents, whether discussion is needed before Task Alloca-

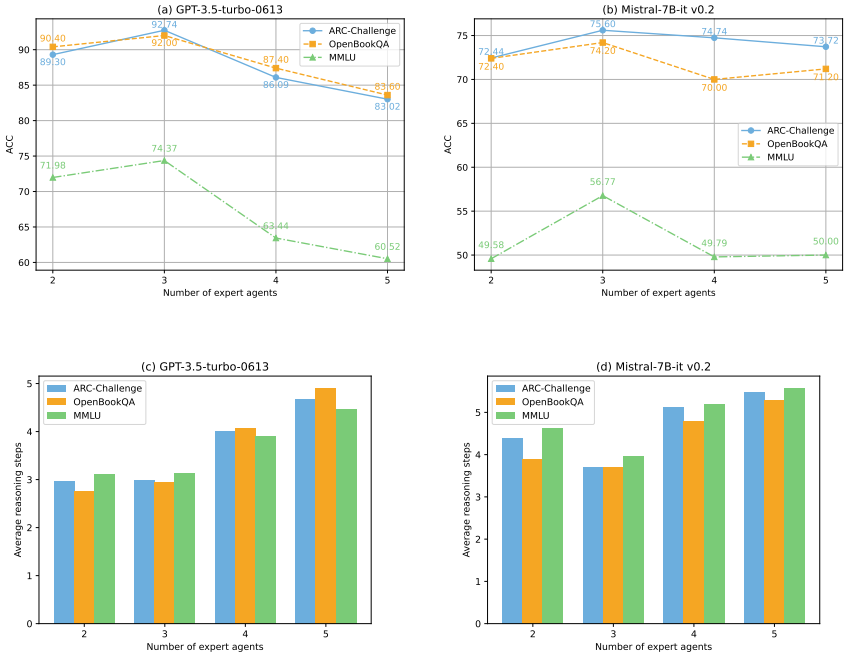


Figure 2. ACC and average reasoning steps of RoT with varying numbers of expert agents

tion, and the relevance of the reasoning process. These experiments aimed to provide a deeper understanding of the mechanisms underlying RoT.

5.1. Investigation of the Number of Expert Agents

In this experiment, we incrementally adjusted the number of expert agents involved in the reasoning process, ranging from two to five agents. We tested the framework on OpenBookQA, ARC-Challenge, and MMLU datasets, recording the ACC and average number of reasoning steps under different settings.

As shown in the figure 2, the relationship between the number of expert agents and performance in complex reasoning tasks is nonlinear. There exists an optimal threshold that balances the coverage of multiple knowledge domains and the avoidance of redundant reasoning, thereby preventing resource wastage. The experimental results indicate that while the RoT's reasoning performance initially improves as the number of expert agents increases, it eventually plateaus or even slightly declines. When the number of expert agents is set to three, RoT achieves the best reasoning performance across the datasets. By combining the analysis with the average number of reasoning steps, we observe that when the number of expert agents is two, RoT is unable to adequately cover the diverse domain knowledge required for complex reasoning tasks. Some sub-tasks are not assigned to the most suitable expert agents, and individual agents are required to handle tasks from multiple domains. This leads to a trial-and-error reasoning process, which negatively impacts both the accuracy and comprehensiveness of the reasoning. In contrast, an excessive number of expert agents can lead to redundant reasoning processes. The viewpoints and reasoning results from different expert agents may overlap significantly. This not only wastes

computational resources but also introduces noise during the integration phase of reasoning, potentially lowering the overall efficiency. Therefore, there is an optimal threshold for the number of expert agents that balances the need for diverse knowledge coverage and the avoidance of redundancy. This experiment demonstrates that using three expert agents strikes the optimal balance, effectively handling reasoning tasks without introducing unnecessary redundancy.

5.2. Necessity of the Opinion Expression Module

To assess whether the Opinion Expression module is necessary in RoT, we compared the performance of the full RoT with a simplified version w/o the Opinion Expression module. In settings, the number of expert agents was kept consistent at three. We conducted experiments on OpenBookQA, ARC-Challenge, and MMLU datasets and recorded and compared various performance metrics. Table 2 shows the experimental results regarding the necessity of the Opinion Expression module.

Table 2. Ablation Study of Necessity of the Opinion Expression Module

Model	Method	OpenBookQA	ARC-Challenge	MMLU
Mistral-7B-it v0.2	RoT(w/o Opinions)	70.60	74.06	50.21
	RoT	74.20	75.60	56.77
GPT-3.5-turbo-0613	RoT(w/o Opinions)	82.40	85.58	60.62
	RoT	92.00	92.74	74.37

The Opinion Expression module plays a crucial role in RoT by providing essential multi-perspective and multi-domain insights, along with semantic associations, which support accurate and well-reasoned task decomposition. The distinct viewpoints expressed by expert agents in this stage reflect their domain-specific understanding and focus, offering clear direction and priority cues for dividing tasks. Bypassing this module and proceeding directly to task allocation can cause mismatches between sub-tasks and the expertise of assigned agents. As a result, certain sub-tasks may lack targeted and in-depth reasoning, negatively impacting the coherence and accuracy of the entire reasoning process. Inappropriate task-agent pairing may lead to incomplete or flawed reasoning, ultimately compromising the overall correctness of the reasoning chain. This highlights the essential role of the Opinion Expression module as a foundational step in achieving efficient, precise, and context-aware reasoning within RoT.

6. Conclusion

In conclusion, Role-of-Thought (RoT) has made significant progress in enhancing the reasoning capabilities of LLMs. By dynamically generating domain-specific expert agents and employing a multi-stage reasoning process, RoT effectively extracts and utilizes implicit knowledge to improve reasoning accuracy and adaptability. The experimental results have shown that RoT outperforms existing baseline methods on multiple datasets, highlighting its strong reasoning capabilities and potential for application in various complex reasoning tasks. The dynamic generation of expert roles and multi-stage reasoning in RoT not only improve reasoning accuracy but also enhance the transparency and interpretability of the reasoning process. Future work will focus on exploring the application potential of RoT in additional domains and tasks to further advance the use of LLMs in NLP.

Author Contributions

This section explains the specific contributions made by each author to the manuscript. The following is a list of the typical roles and responsibilities.

Daojian Zeng (zengdj@hunnu.edu.cn): Conceptualization, Methodology, Software.
Penghao Li (lph1004@hunnu.edu.cn): Methodology, Software, Writing-Original draft.
Tianle Li (linjiadegou2@hunnu.edu.cn): Software, Investigation.
Shanshan Li (202470294365@hunnu.edu.cn): Software, Investigation.
Tingjiao Huang (huangtj@hunnu.edu.cn): Software, Investigation.
Lincheng Jiang (linchengjiang@nudt.edu.cn): Supervision, Writing-review, Editing.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No. 62276095 and 72204261) and the National Social Science Foundation of China(No.20ZD047).

References

[1] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.
[2] K. Shuster, J. Urbanek, E. Dinan, A. Szlam, and J. Weston, "Dialogue in the wild: Learning from a deployed role-playing game with humans and bots," in *Find. Assoc. Comput. Linguist. ACL-IJCNLP 2021*, 2021, pp. 611–624.
[3] S. Lin, J. Hilton, and O. Evans, "Truthfulqa: Measuring how models mimic human falsehoods," *arXiv preprint arXiv:2109.07958*, 2021.
[4] J. Ahn, R. Verma, R. Lou, D. Liu, R. Zhang, and W. Yin, "Large language models for mathematical reasoning: Progresses and challenges," *arXiv preprint arXiv:2402.00157*, 2024.
[5] P. H. Panawenna and S. Wickramanayake, "Understanding omitted facts in transformer-based abstractive summarization," in *2024 Moratuwa Eng. Res. Conf. (MERCon)*. IEEE, 2024, pp. 624–629.
[6] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung, "Survey of hallucination in natural language generation," *ACM Comput. Surv.*, vol. 55, no. 12, pp. 1–38, 2023.
[7] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn, "Direct preference optimization: Your language model is secretly a reward model," *Adv. Neural Inf. Process. Syst.*, vol. 36, pp. 53 728–53 741, 2023.
[8] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, "Training language models to follow instructions with human feedback," *Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 27 730–27 744, 2022.
[9] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. Li, Y. Wu *et al.*, "Deepseekmath: Pushing the limits of mathematical reasoning in open language models," *arXiv preprint arXiv:2402.03300*, 2024.
[10] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," *Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 24 824–24 837, 2022.
[11] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou, "Self-consistency improves chain of thought reasoning in language models," *arXiv preprint arXiv:2203.11171*, 2022.
[12] Y. Fu, H. Peng, A. Sabharwal, P. Clark, and T. Khot, "Complexity-based prompting for multi-step reasoning," *arXiv preprint arXiv:2210.00720*, 2022.
[13] M. Besta, N. Blach, A. Kubicek, R. Gerstenberger, M. Podstawski, L. Gianinazzi, J. Gajda, T. Lehmann, H. Niewiadomski, P. Nyczyk *et al.*, "Graph of thoughts: Solving elaborate problems with large language models," in *Proc. AAAI Conf. Artif. Intell.*, vol. 38, no. 16, 2024, pp. 17 682–17 690.
[14] S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," *Adv. Neural Inf. Process. Syst.*, vol. 36, pp. 11 809–11 822, 2023.
[15] B. Xu, A. Yang, J. Lin, Q. Wang, C. Zhou, Y. Zhang, and Z. Mao, "Expertprompting: Instructing large language models to be distinguished experts," *arXiv preprint arXiv:2305.14688*, 2023.
[16] D. X. Long, D. N. Yen, A. T. Luu, K. Kawaguchi, M. Y. Kan, and N. F. Chen, "Multi-expert prompting improves reliability, safety, and usefulness of large language models," *arXiv preprint arXiv:2411.00492*, 2024.
[17] Y. Shoham and K. Leyton-Brown, *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge, U.K.: Cambridge Univ. Press, 2008.

[18] Z. Wang, S. Mao, W. Wu, T. Ge, F. Wei, and H. Ji, "Unleashing the emergent cognitive synergy in large language models: A task-solving agent through multi-persona self-collaboration," *arXiv preprint arXiv:2307.05300*, 2023.
[19] Z. Hu, H. P. Chan, J. Li, and Y. Yin, "Unlocking varied perspectives: A persona-based multi-agent framework with debate-driven text planning for argument generation," *arXiv preprint arXiv:2406.12345*, 2024.
[20] T. Liang, Z. He, W. Jiao, X. Wang, Y. Wang, R. Wang, Y. Yang, S. Shi, and Z. Tu, "Encouraging divergent thinking in large language models through multi-agent debate," *arXiv preprint arXiv:2305.19118*, 2023.
[21] G. Wan, Y. Wu, J. Chen, and S. Li, "Cot rerailer: Enhancing the reliability of large language models in complex reasoning tasks through error detection and correction," *arXiv preprint arXiv:2408.13940*, 2024.
[22] S. K. Radha, Y. N. Jelyani, A. Ghukasyan, and O. Goktas, "Iteration of thought: Leveraging inner dialogue for autonomous large language model reasoning," *arXiv preprint arXiv:2409.12618*, 2024.
[23] Y. Du, S. Li, A. Torralba, J. B. Tenenbaum, and I. Mordatch, "Improving factuality and reasoning in language models through multiagent debate," in *41st Int. Conf. Mach. Learn. (ICML)*, 2023.
[24] H. Luo, Q. Sun, C. Xu, P. Zhao, J. Lou, C. Tao, X. Geng, Q. Lin, S. Chen, and D. Zhang, "Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct," *arXiv preprint arXiv:2308.09583*, 2023.
[25] L. Trung, X. Zhang, Z. Jie, P. Sun, X. Jin, and H. Li, "Reft: Reasoning with reinforced fine-tuning," in *Proc. 62nd Annu. Meet. Assoc. Comput. Linguist.*, 2024, pp. 7601–7614.
[26] D. Kalajdzievski, "Scaling laws for forgetting when fine-tuning large language models," *arXiv preprint arXiv:2401.05605*, 2024.
[27] S. Liu, Y. Yao, J. Jia, S. Casper, N. Baracaldo, P. Hase, Y. Yao, C. Y. Liu, X. Xu, H. Li *et al.*, "Rethinking machine unlearning for large language models," *Nature Mach. Intell.*, pp. 1–14, 2025.
[28] D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi *et al.*, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," *arXiv preprint arXiv:2501.12948*, 2025.
[29] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," *Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 22 199–22 213, 2022.
[30] X. Xu, C. Tao, T. Shen, C. Xu, H. Xu, G. Long, J. Lou, and S. Ma, "Re-reading improves reasoning in large language models," in *Proc. 2024 Conf. Empir. Methods Nat. Lang. Process. (EMNLP)*, 2024, pp. 15 549–15 575.
[31] A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao, S. Wiegreffe, U. Alon, N. Dziri, S. Prabhumoye, Y. Yang *et al.*, "Self-refine: Iterative refinement with self-feedback," *Adv. Neural Inf. Process. Syst.*, vol. 36, pp. 46 534–46 594, 2023.
[32] Y. Weng, M. Zhu, F. Xia, B. Li, S. He, S. Liu, B. Sun, K. Liu, and J. Zhao, "Large language models are better reasoners with self-verification," *arXiv preprint arXiv:2212.09561*, 2022.
[33] A. Jaech, A. Kalai, A. Lerer, A. Richardson, A. El-Kishky, A. Low, A. Helyar, A. Madry, A. Beutel, A. Carney *et al.*, "Openai o1 system card," *arXiv preprint arXiv:2412.16720*, 2024.
[34] T. Liu, X. Wang, W. Huang, W. Xu, Y. Zeng, L. Jiang, H. Yang, and J. Li, "Groupdebate: Enhancing the efficiency of multi-agent debate using group discussion," *arXiv preprint arXiv:2409.14051*, 2024.
[35] T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal, "Can a suit of armor conduct electricity? a new dataset for open book question answering," *arXiv preprint arXiv:1809.02789*, 2018.
[36] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord, "Think you have solved question answering? try arc, the ai2 reasoning challenge," *arXiv preprint arXiv:1803.05457*, 2018.
[37] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, "Measuring massive multitask language understanding," *arXiv preprint arXiv:2009.03300*, 2020.
[38] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de Las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier *et al.*, "Mistral 7b," *arXiv preprint arXiv:2310.06825*, 2023.
[39] OpenAI, "Introducing chatgpt," [Online], 2022, accessed: Nov. 30, 2022. [Online]. Available: https://openai.com/blog/chatgpt

APPENDICES

A Dataset Distribution

We listed the quantity distribution of each dataset.

Dataset		Quantity
OpenBookQA		500
ARC-Challenge		1172
MMLU	College computer science	100
	College mathematics	100
	College medicine	173
	College physics	102
	Computer security	100
	Econometrics	114
	Electrical engineering	145
	Formal logic	126
Total		2632

B Prompt Design

We presents the prompts utilized in the distinct modules of RoT. These prompts are crafted to guide expert agents in delivering outputs that align with their roles and capabilities.

B.1 Expert Agent Generation Module Prompt

This prompt is designed to generate various domain expert agent roles and brief descriptions for tackling specific tasks. By inputting a task question, the system outputs a dictionary - formatted answer containing expert agent roles and descriptions, setting the foundation for subsequent task allocation and reasoning processes.

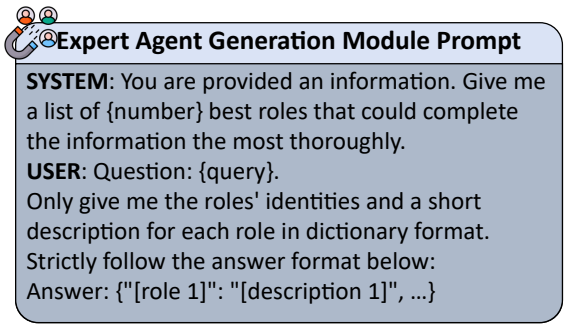


Figure 3. Prompt design template for Expert Agent Generation module

B.2 Opinion Expression Module Prompt

This prompt encourages expert agents to offer knowledge or insights relevant to their roles for solving a given problem. Each agent provides a concise viewpoint, enhancing task decomposition and allocation with diverse perspectives and professional information.

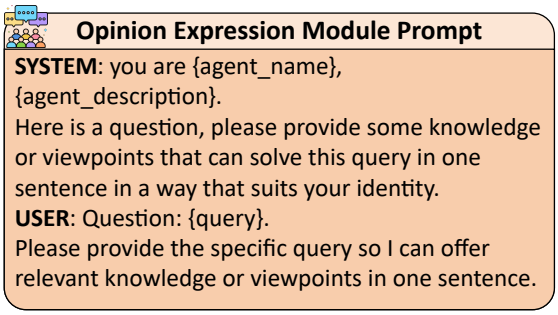


Figure 4. Prompt design template for Opinion Expression module

B.3 Task Allocation module Prompt

Based on the problem and expert agents' viewpoints, this prompt divides tasks into multiple stages and assigns them to suitable agents. It outputs a dictionary indicating each stage's agent and reasoning content, ensuring rational and targeted task distribution.

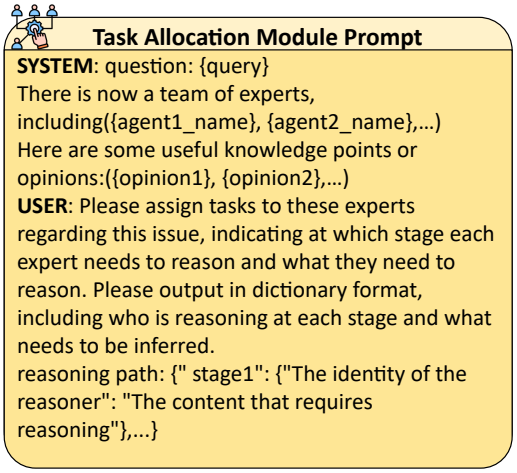


Figure 5. Prompt design template for Task Allocation module

B.4 Multi-Stage Reasoning module Prompt

This prompt guides expert agents through Multi-Stage Reasoning. Agents, building on prior reasoning, provide concise answers for their stage, contributing to a coherent and accurate reasoning chain.

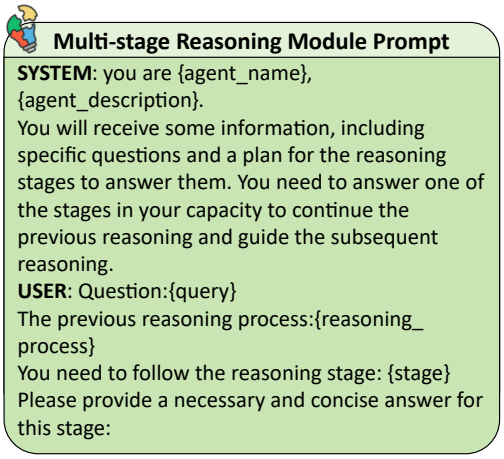


Figure 6. Prompt design template for Multi-Stage Reasoning module

B.5 Summary Prompt

This prompt requires the agent to summarize the final answer from the entire reasoning process. The agent outputs the sole correct answer in a specified format, ensuring a clear and precise conclusion.

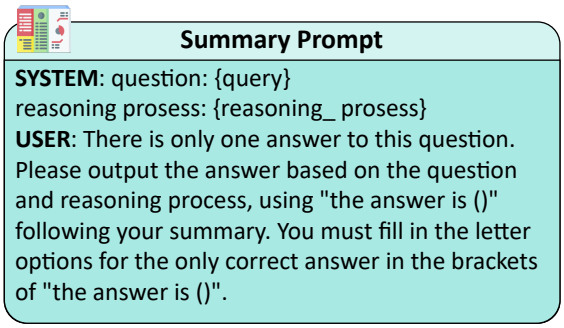


Figure 7. Prompt design template for summary