CP-Search: A Chain Progressive Search Training Framework Incentivizing the Cognitive Behaviors for Searching in LLMs

Zehua Wang[1,2], **Shipeng Li**[1], **Buzhou Tang**[1,2,3]*
[1]Department of Computer Science, Harbin Institue of Technology (Shenzhen), Shenzhen, China
[2]Guangdong Provincial Key Laboratory of Intelligent Information Processing
[3]Peng Cheng Laboratory, Shenzhen, China
wangzehua@stu.hit.edu.cn
tangbuzhou@gmail.com

Abstract

Retrieval-Augmented Generation (RAG) has been demonstrated to effectively mitigate the knowledge recency issue in Large Language Models (LLMs) while significantly reducing hallucinations. However, existing RAG methods exhibit insufficient capability in modeling reasoning paths for complex multi-hop reasoning tasks. While Reinforcement Learning (RL) has demonstrated success in enhancing model reasoning ability, Token-level RL frameworks exhibit inherent limitations in maintaining coherent reasoning trajectories. This approach remains susceptible to the compounding accumulation of contextual errors during the retrieval process, ultimately resulting in erroneous output generation. To address this challenge, we propose Chain Progressive Search (CP-Search), a novel two-stage training framework designed to enhance the model's retrieval capability in complex scenarios. This framework models the entire retrieval process as a Retrieval-level Markov Decision Process, systematically optimizing the model's retrieval behavior at each step of the chained retrieval. Specifically, CP-Search first constructs a retrieval-cognitive behavioral dataset and employs Supervised Fine-Tuning (SFT) to endow the model with cognitive behaviors for searching. More importantly, by introducing a dense progressive procedural reward in reinforcement learning training, CP-Search significantly improves the model's reasoning consistency and feedback correction ability in chained retrieval. Experiments conducted on multiple multi-hop datasets demonstrate that CP-Search significantly outperforms existing RAG methods in complex multi-hop reasoning tasks.

Code — https://github.com/kyre-99/CP-Search

1 Introduction

Large language models (LLMs) have demonstrated remarkable capabilities in semantic understanding and question answering (Jaech et al. 2024), yet their performance remains constrained by two challenges: knowledge obsolescence and hallucination phenomena (Ji et al. 2023; Zhang et al. 2025b), primarily due to the timeliness and coverage limitations of training data (Luo et al. 2023). The introduction of Retrieval-Augmented Generation (RAG) has significantly mitigated these limitations by providing high-quality contextual information to the models (Lewis et al. 2020; Shi et al. 2024; Wang et al. 2025).

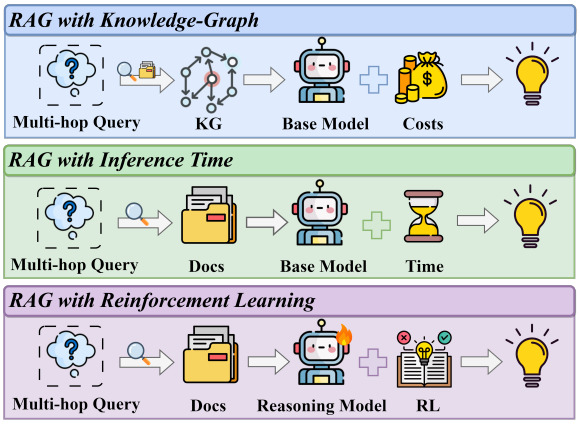

Figure 1: Comparison of Different RAG Methods in Complex Retrieval Scenarios.

Although RAG has been widely adopted in LLMs, their traditional pipeline—where the model initiates a single retrieval and obtains semantically relevant documents from a knowledge base—exhibits notable limitations when handling complex multi-hop reasoning tasks. Due to the lack of iterative retrieval capability, models often fail to construct complete evidence chains, leading to reasoning failures (Gao et al. 2023). To address this, a common approach involves pre-constructing high-quality Knowledge Graphs (KGs) to provide structured context (Sanmartin 2024; Guo et al. 2024), leveraging inter-node relationships to assist in multi-hop reasoning. However, KG construction relies on expert annotation, which is costly and difficult to scale (Team et al. 2025; Guo et al. 2025). Some other works attempt to increase the inference time of the model, such as using Chain of Thought (CoT) or multiple sampling, which also brings performance improvements (Press et al. 2022; Li et al. 2024; Trivedi et al. 2022a). Recent studies have shown that Reinforcement Learning (RL) can enhance LLM's reasoning ability . Some research has also applied this to the retrieval domain, demonstrating performance improvements (Jin et al. 2025a; Song et al. 2025). The char-

*Corresponding author

acteristics and performance of different methods can be referred to Figure 1 and Figure 2.

However, although RL can effectively enhance the reasoning ability of LLM, there are still two main limitations. First, recent research shows that reinforcement learning optimizes the token sampling process by increasing the probability of the model's existing cognitive behaviors (for example, increasing the probability of words like "wait") (Zeng et al. 2025; Hochlehnert et al. 2025), but for behaviors that did not exist before, such as dynamically adjusting the retrieval strategy and correcting the error detection process, they cannot be truly enabled by RL (Gandhi et al. 2025). In addition, relying solely on result rewards makes it difficult to deal with multi-hop scenarios. In such scenarios, each step of the process needs to be accurate. If early retrieval errors accumulate continuously, it will ultimately lead to reasoning biases (Chen et al. 2024). But, training effective procedural reward models for complex retrieval scenarios is difficult, and efficiently providing process-level feedback in such contexts remains an open problem.

To address these shortcomings, we propose the Chain Progressive Search framework (CP-Search), which reconstructs retrieval-based reasoning through a dual mechanism: (1) Model the RL optimization process as a Retrieval-level Markov Decision Process (Retrieval-Level MDP) for the retrieval task, and design six retrieval behaviors (such as verification, sub-goal decomposition) that the model should possess in chained retrieval. Based on this, construct a specialized dataset and explicitly model the retrieval decision-making process through supervised learning, enabling the model to have powerful retrieval reflection capabilities. (2) Introduce concise and effective progressive procedural rewards and decompose the result rewards into specific indicators such as retrieval accuracy and error penalties. This design enables the model not only to understand the semantic constraints of single-step retrieval but also to maintain and possess the dynamic error correction ability for process errors, ultimately forming a dynamically optimized reasoning retrieval chain.

In the evaluations of multiple multi-hop benchmarks, CP-Search has achieved significant improvements and outperformed various RAG methods, demonstrating the effectiveness of the method in this paper.

2 Related Work

2.1 Retrieval-Augmented Generation

Retrieval-Augmented Generation has been widely adopted in LLM applications. Significant progress has been made in building complex retrieval and re-ranking models (Zhang et al. 2025a), enabling LLMs to obtain highly relevant context information from retrieved content. Another research direction focuses on addressing complex retrieval challenges, especially for multi-hop chained retrieval. Graph-based methods such as LightRAG (Guo et al. 2024) and KG-RAG (Huang, Zhang, and Xiao 2025) solve this problem by converting knowledge bases into knowledge graphs, explicitly modeling the relationships between entities through graph structures to enhance the ability of LLMs to capture text relationships. Other methods focus on improving the LLM inference stage, such as using Chain of Thought (CoT) and multi-path exploration (Press et al. 2022; Li et al. 2024; Trivedi et al. 2022a). However, these methods usually incur significant computational costs in terms of resource costs and inference time.

2.2 Reinforcement Learning for Verification and Cognitive Enhancement

The release of DeepSeek-R1-zero and DeepSeek-R1 has demonstrated the effectiveness of using answer verification as a reward signal (Guo et al. 2025). Combining a rule-based reward model with an efficient reinforcement learning algorithm can significantly improve the performance of LLMs in complex reasoning tasks. Subsequent research, such as Search-R1 (Jin et al. 2025a) and R1-Searcher (Song et al. 2025), extended this framework to retrieval applications, adopting end-to-end reinforcement learning training to enhance the model's ability to interleave reasoning and retrieval operations. However, recent studies (Hochlehnert et al. 2025; Gandhi et al. 2025) have shown that the effectiveness of reinforcement learning in improving reasoning ability is closely related to the probability distribution of the model's internal cognitive behaviors. For example, under the same training process and with comparable model parameter scales, the Qwen family of models has demonstrated emergent reasoning capabilities, whereas the Llama series models have not shown a corresponding level of performance. This may be due to the model initially lacks reasoning behaviors. Therefore, there are still key challenges, how to precisely define optimizable behavioral indicators for the retrieval process, and how to develop a reinforcement learning-based self-improvement mechanism for the retrieval system. Solving these challenges will contribute to the wider application of RAG technology in the field of complex tasks.

3 Preliminaries

3.1 Retrieval-Level MDP

The basic framework for representing language generation as a reinforcement learning task is a Token-level Markov Decision Process (MDP) (Yue et al. 2025). Let the instruction be denoted as x and the output of LLM as y. Both x and y can be segmented into token sequences. Specifically, the input text x can be represented as $x = (x_0, \ldots, x_m)$, where each token x_i and y_i is from the predefined discrete vocabulary $\mathcal{A}$. We defined the Markov Decision Process as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$. The definitions of each component are as follows: The state space $\mathcal{S}$ contains all generation histories, with state $s_t = (x_0, \ldots, x_m, y_0, \ldots, y_t)$ representing the concatenation of instruction and generated tokens at step t. The action space $\mathcal{A}$ corresponds to the fixed token vocabulary, where each action $a_t \in \mathcal{A}$ selects the next output token y_{t+1}. The state transition function characterizes a token-to-token generation function. Formally, given the current state $s_t = (x_0, \ldots, x_m, y_0, \ldots, y_t)$, an action $a = y_{t+1}$ (i.e., the next generated token), and the subsequent state

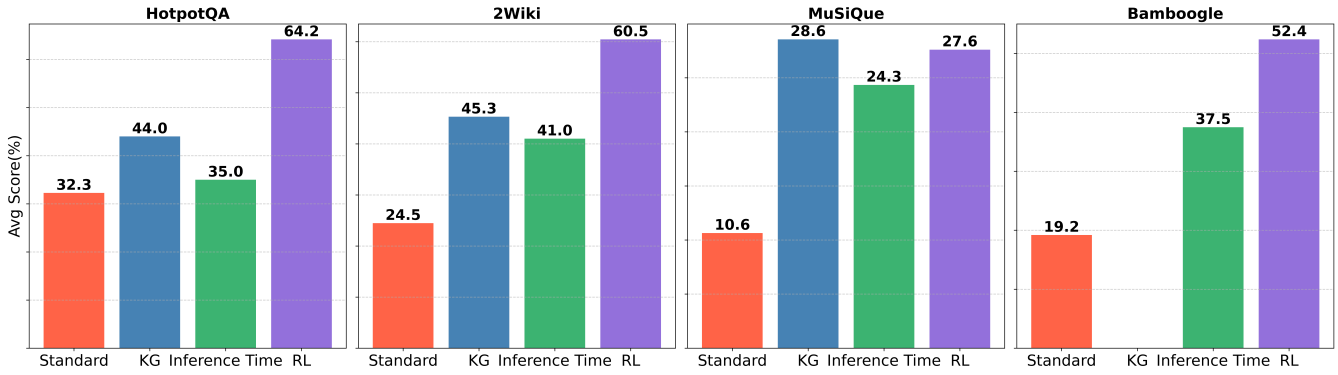

Figure 2: Comparison of results of different RAG methods on multiple multi-hop datasets. Details results and setups can be found in Table 1.The avg score is the average of the AVG of all methods under this type.

$s_{t+1} = (x_0, \ldots, x_m, y_0, \ldots, y_t, y_{t+1})$. The reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ provides scalar feedback through either learned human preferences (e.g., RLHF (Ouyang et al. 2022)) or task-specific rules. The $\gamma \in [0, 1]$ serves as the discount factor for future rewards. This MDP formulation enables sequential decision-making optimization for language generation tasks.

However, for the retrieval task, it is not merely a language generation task but also involves interaction with the outside world: requesting the retriever to return the context. Using only Token-level MDP cannot establish an effective evaluation of a single search behavior. Therefore, we propose a Retrieval-level MDP representation to better model the hierarchical process of information retrieval. The main differences are as follows:

- **Action Space.** Instead of using discrete vocabulary to represent actions, for the retrieval process, we designed six key actions, which also correspond to the complete retrieval process: First is **Plan** through which preset goals are extracted to ensure that the retrieval process does not deviate from the theme; second is **Reflection**, which is used to check whether there are errors in the current steps and guide subsequent operations; then comes **Search**, which is the core action of interacting with the retrieval database; the relevant information returned by the retrieval database is categorized as **Information**; in addition, the system supports the **Cite** of key evidence to enhance the credibility; finally, the model generates the final **Answer** based on the above steps. Each action is marked with <>, for example,⟨search⟩ represents **Search**. The retrieval action space $\mathcal{A}_\text{r} = \{a_r^\text{plan}, a_r^\text{reflection}, a_r^\text{search}, a_r^\text{information}, a_r^\text{cite}, a_r^\text{answer}\}$.
- **State Space.** The retrieval-level state space $\mathcal{S}_\text{r}$ contains all generation action a_r, and $s_t = (x_0, \ldots, x_m, \underbrace{y_0, \ldots y_i}_{a_{r,0}^{plan}}, \underbrace{y_{i+1}, \ldots y_j}_{a_{r,1}^{search}}, \ldots \underbrace{y_{l+1}, \ldots y_k}_{a_{r,t}^{cite}})$ representing the retrieval-level state at step t.
- **Reward function.** The retrieval-level reward function $R : s_t \times a_{r,t}^{information} \to r_t$ is designed to explicitly evaluate the effectiveness of each retrieval operation during the reasoning process. Here, s_t represents the current reasoning state at step t, $a_{r,t}^\text{information}$ denotes the specific action of obtaining the retrieved information, and r_t quantifies the immediate reward value.

3.2 Reinforcement Learning with Verifiable Rewards

The objective of reinforcement learning is to **maximize the expected cumulative reward**. This is achieved by optimizing the policy $\pi(a|s)$ over the collected trajectories. Formally, this can be expressed as:

$$J(\pi) = \mathbb{E}_{\tau\sim\pi}[R(\tau)] = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{N-1} r_t\right] \tag{1}$$

where $R(\tau)$ is the total reward for trajectory τ, and r_t is the reward received at step t.

For some **verifiable tasks**, we can use rule-based reward model for verification. This approach improves the transparency and interpretability of the reward mechanism. To avoid the need for value functions in order to reduce the use of computational resources, the **Group Relative Policy Optimization (GRPO) algorithm** (Shao et al. 2024) emerges as a method to effectively train LLM with verifiable rewards. The objective function of GRPO is as the follow:

$$J_\text{GRPO}(\theta) = \mathbb{E}_{q,o}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\min\left(r_t\hat{A}_t, \text{clip}(r_t)\hat{A}_t\right) - kD_\text{KL}(\pi_\theta\|\pi_\text{ref})\right] \tag{2}$$

where: $q \sim P(Q)$, $\text{clip}(r_t) \triangleq \text{clip}(r_t, 1-\epsilon, 1+\epsilon)$, $r_t = \frac{\pi_\theta}{\pi_{\theta_\text{old}}}$. Perform rollout on the same question q, and normalize the rewards r within the same group. Set the normalized rewards as the advantage $\hat{A}_{i,t}$ of each token, i.e., $\hat{A}_{i,t} = \frac{r_{i,t}-\text{mean}(r)}{\text{std}(r)}$. By such operation, bring the model closer to the direction with higher rewards.

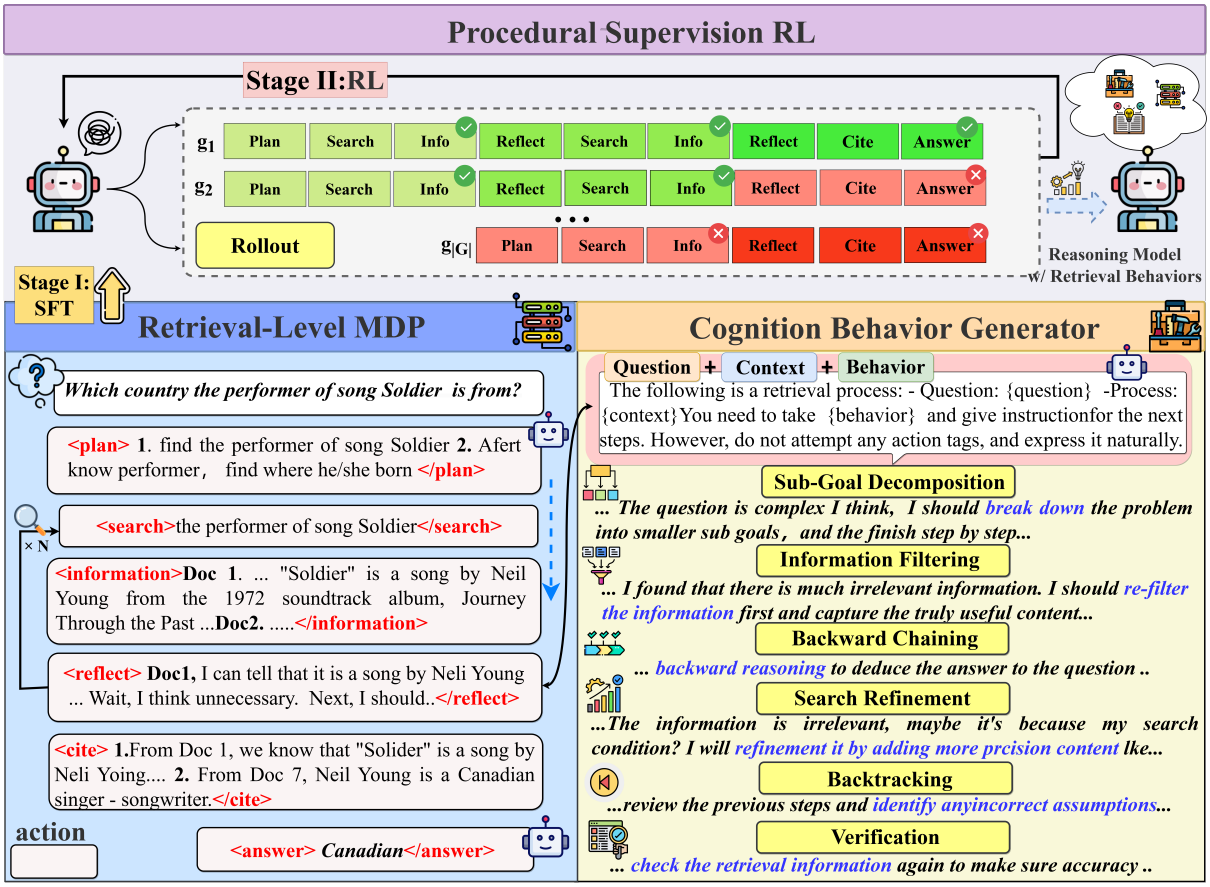

Figure 3: The overall framework of CP-Search. At the top, the two-stage training process of the model is presented, while at the bottom, Retrieval-Level MDP and Retrieval Cognitive Behaviors are shown respectively.

4 Methods

In this section, we will introduce the CP-Search framework, which is shown in Figure 3. The CP-Search framework enhances the retrieval ability of large LLM through a two-stage training: First, in the cognitive behavior modeling phase, various reflective cognitive behaviors in retrieval are designed and then embedded into the retrieval-level MDP. A corresponding dataset is constructed through a pipeline to endow the model with cognitive behaviors during the retrieval process; subsequently, in the reinforcement learning phase, the Progressive Procedural Reward is adopted for policy optimization, dynamically adjusting the probability distribution of behavior perception.

4.1 Datasets of Retrieval Cognitive Behavior

Retrieval Cognitive Behavior. For the retrieval process, the model context includes its own output and the information returned by the retrieval system. Usually, information often has a significant distribution difference from the original training data of the LLM. This difference may affect the model's own reasoning paradigm. A particularly common problem is that the LLM tends to consider the retrieved content to be completely correct, resulting in a significant decline in the context discrimination ability. On the other hand, the model might not have such specific behaviors oriented towards the retrieval process.

Therefore, we propose a novel method that aims to systematically reshape LLMs during the retrieval process. Drawing the insight from (Gandhi et al. 2025), we have developed a structured set of behavioral modalities for the retrieval process, comprising: Verification, Sub-Goal Decomposition, Search Refinement, Information Filtering, Backtracking, and Backward Chaining, they are formally defined as follows:

- **Sub-Goal Decomposition**: A strategy for breaking down complex queries into independently addressable subproblems, reducing cognitive load through hierarchical retrieval.
- **Information Filtering**: The process of scavenging and removing invalid information.
- **Backward Chaining**: A retrieval guidance strategy that reversely infers the required evidence from the target conclusion, which is particularly suitable for hypothesis verification tasks.
- **Search Refinement**: An iterative optimization process that dynamically adjusts query content (e.g. keywords, semantic scope) based on initial retrieval information.

- **Backtracking**: A self-correcting mechanism that returns to critical decision points for re-retrieval upon detecting reasoning path deviations.
- **Verification**: The process of verifying the relevance and reliability of the retrieved information.

We choose these behaviors because they represent strategies that are different from the common linear and monotonic reasoning patterns in LLM. These behaviors enable more dynamic, human-search-like reasoning trajectories, in which solutions can evolve non-linearly. Different behavior constructs are generated through prompt guidance.

Retrieval-level MDP Dataset. The retrieval cognitive behaviors introduced above are crucial for enabling LLM to navigate the complexities of retrieval-augmented generation effectively. In Retrieval-level MDP, these cognitive behaviors are specifically integrated into the **Reflection** action. Specifically, for state $s_t = (\tau_t)$ with trajectory τ at step t, the reflection process in our MDP framework executes: (1) action selection: $a_t^{\text{reflect}} \sim \pi_{\text{reflect}}(a|s_t)$ from predefined cognitive behaviors, (2) reflection generation: $a_t^{\text{reflect}} = f(\tau_t)$ where f represents more powerful LLM, and (3) state update: $s_{t+1} = \text{update}(\tau_t, a_t^{\text{reflect}})$. Repeat the above process until a_{answer} appears. At this point, it can be used as a complete training trajectory.

4.2 Procedural Supervision RL

In order to further increase the probability of the LLM using retrieval cognitive behaviors during the retrieval process, we adopt the reinforcement learning for optimization. Considering that the retrieval information of each step in the chain retrieval process will affect the subsequent steps, the maintenance of the intermediate process is particularly important. To this end, we propose an improved method based on **GRPO**, which improves the accuracy of the model in each round of retrieval by introducing a procedural reward mechanism and is referred to as **CPGRPO**. The following will introduce this method in detail and the design of the reward function.

Chained Progressive Procedural Reward with GRPO. During the chain retrieval process, we compare the retrieved content $\langle\text{information}\rangle$ with the golden evidence $\mathcal{G}$ to evaluate the queried content $\langle\text{search}\rangle$. Formally, let a trajectory of the retrieval process be defined as: $\tau = \{\ldots, a_{r,t-1}^{\text{search}}, a_{r,t}^{\text{information}}\}$ where at each step t, the retrieves information are evaluated against golden evidence $\mathcal{G}$ through the validity function $\Delta_t = \text{Valid}(t) = \mathbb{I}(a_{r,t}^{\text{information}} \in \mathcal{G})$, where $\mathbb{I}(\cdot)$ is the indicator function. The reward mechanism employs two accumulators: γ_+ for valid retrievals and γ_- for invalid retrievals. The accumulators update dynamically as:

$$\begin{cases} \gamma_+ \leftarrow \gamma_+ + \alpha \cdot \Delta_t \\ \gamma_- \leftarrow \gamma_- + \beta \cdot (1 - \Delta_t) \end{cases} \tag{3}$$

where α is reward for valid retrieval and β is penalty for invalid retrieval. The progressive process reward at step t is expressed as $pr_t = f(\gamma_+, \gamma_-, \Delta_t) = \gamma_+\Delta_t + \gamma_-(1 - \Delta_t)$.

For the complete trajectory τ_i, if the retrieval is performed n times, we can obtain the procedural rewards $pr = \{pr^{index(cpt1)}, pr^{index(cpt2)}, \ldots, pr^{index(cptn)}\}$ and $index(cptn)$ is the end token index of the n-th retrieval. By propagating the procedural rewards of each checkpoint backward, we can obtain the procedural rewards of each token:

$$\begin{aligned} pr_i = \{&pr_{i,1}^{index(cpt1)}, pr_{i,2}^{index(cpt1)}, \ldots, \\ &pr_{i,k}^{index(cpt2)}, \ldots, pr_{i,|\tau_i|}^{index(cptn)}\} \end{aligned} \tag{4}$$

where k is the k-th token.

In addition, we have also retained the result rewards. Due to the textual nature of the answers, we use the F1 score as the reward. To align with the Retrieval-level MDP, we require the correct format (actions are placed within the <> tags), as follows:

$$\Phi_{\text{ORM}}(o_i) = \begin{cases} F1, & \text{if format is correct} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

Combining the outcome rewards of each token with the procedural rewards, the advantage values $\hat{A}'_{i,t}$ of CPGRPO, can be expressed as follows:

$$\hat{A}'_{i,t} = |\hat{A}_{i,t}| \cdot pr_{i,t} \tag{6}$$

where the $sign(\hat{A}'_{i,t})$ is depended on $pr_{i,t}$ which means rewards and penalties should be given for correct or incorrect actions in the trajectory separately, rather than for the entire trajectory.

$$\nabla_\theta J_{\text{CPGRPO}} \propto \sum_{t=1}^{|o_i|} \nabla_\theta \log \pi_\theta(a_t|o_{<t}) \cdot \left(|\hat{A}_{i,t}| \cdot pr_{i,t}\right) \tag{7}$$

As can be seen from Equation 7, the procedural rewards directly influence the policy gradient through the advantage function. This formulation preserves the original objective structure while enabling fine-grained token-level control through linear combination of both reward types.

5 Experiment

5.1 Experimental Setup

Datasets and Evaluation. We evaluate on four multi-hop QA datasets: HotpotQA (Yang et al. 2018), 2WikiMultiHopQA (Ho et al. 2020), MuSiQue (Trivedi et al. 2022b), and Bamboogle (Press et al. 2022). Following (Song et al. 2025), we use Accuracy (ACC_R) to check whether generated answers contain the standard answer's key information, and GPT-4o-mini as an automated judge for semantic consistency (ACC_L). For training, we build a Retrieval-level MDP dataset by sampling 1000 examples each from HotpotQA and 2WikiMultiHopQA, keeping only exact matches or F1 $>$ 0.5, yielding 1200 high-quality SFT samples. RL training uses another random 1000 samples from each dataset. MuSiQue and Bamboogle are held out as out-of-domain benchmarks and are not used for training. Test settings follow (Song et al. 2025).

Baselines. We compared CP-Search with several baselines: **1)** a naive LLM that directly generates answers; **2)** chunk- and summary-based methods (Jiang et al. 2023; Xu,

Model	HotpotQA †			2Wiki†			MuSiQue‡			Bamboogle‡			ALL		
	ACC_R	ACC_L	AVG	ACC_R	ACC_L	AVG	ACC_R	ACC_L	AVG	ACC_R	ACC_L	AVG	ACC_R	ACC_L	AVG
Direct Generate	18.2	20.0	19.1	20.8	19.4	20.1	3.2	2.4	2.8	8.0	9.6	8.8	12.6	12.8	12.7
Naïve RAG	28.2	25.2	26.7	8.6	7.6	8.1	4.8	6.2	5.5	16.8	20.0	18.4	14.6	14.8	14.7
LongLLMLingua*	35.8	45.0	40.4	32.4	31.6	32.0	15.0	17.2	16.1	24.8	28.8	26.8	27.0	30.6	28.8
RECOMP*	33.2	39.8	36.5	29.8	30.6	30.2	11.8	13.4	12.6	13.6	17.6	15.6	22.1	25.4	23.7
Selective-Context*	36.6	44.2	40.4	35.0	29.0	32.0	15.2	17.2	16.2	24.0	28.8	26.4	27.7	29.8	28.7
LightRAG-Local	45.8	45.2	45.5	51.4	48.6	50.0	28.0	25.2	26.6	-	-	-	41.7	39.7	40.7
LightRAG-Global	46.4	46.2	46.3	42.8	41.4	42.1	26.8	27.2	27.0	-	-	-	38.7	38.3	38.5
LightRAG-Hybrid	49.2	51.2	50.2	54.8	54.8	54.8	31.0	32.0	31.5	-	-	-	45.0	46.0	45.5
KET-RAG	33.0	35.0	34.0	33.8	35.0	34.4	29.8	28.4	29.1	-	-	-	32.2	32.8	32.5
Self-ASK*	39.2	46.2	42.7	33.6	47.8	40.7	26.0	27.0	26.5	33.6	41.6	37.6	33.1	40.7	36.9
Iter-RetGen*	37.4	45.6	41.5	32.6	27.0	29.8	17.8	18.8	18.3	23.2	25.6	24.4	27.75	29.25	28.5
IRCoT*	43.4	30.8	37.1	49.2	11.4	30.3	19.2	21.4	20.3	27.2	18.4	22.8	34.8	20.5	27.6
CP-Planner*	40.4	41.6	41.0	52.0	47.8	49.9	27.2	26.2	26.7	48.8	52.4	50.6	42.1	42.0	42.1
ReARTeR*	46.8	50.6	48.7	55.4	53.4	54.4	29.6	30.2	29.9	49.6	**54.4**	52.0	45.4	47.2	46.3
Search-R1	53.0	53.2	53.1	45.4	44.4	44.9	19.6	20.8	20.2	47.2	50.4	48.8	41.3	42.2	41.8
R1-Searcher*	**65.4**	**75.0**	**70.2**	63.6	65.0	64.3	28.2	31.4	29.8	52.8	**54.4**	53.6	52.5	56.5	54.5
CP-Search	63.8	74.6	69.2	**72.4**	**72.0**	**72.2**	**32.2**	**33.4**	**32.8**	**55.2**	**54.4**	**54.8**	**55.9**	**58.6**	**57.3**

Table 1: The main results of CP-Search. "AVG" represents the average of the corresponding ACC_R and ACC_L . "*" indicates that the results are cited from official paper (Song et al. 2025). "-": we exclude the results of KG-based methods on Bamboogle due to the fact that building its knowledge graph requires too much cost. †/‡ represents in-domain/out-of-domain datasets.The top two results in each column are highlighted in **bold** and underlined.

Shi, and Choi 2024; Li et al. 2023); **3)** KG-based methods KET-RAG (Huang, Zhang, and Xiao 2025) and LightRAG (Guo et al. 2024); **4)** inference-time approaches, including CoT methods (Self-ASK (Press et al. 2022), Iter-RetGen (Shao et al. 2023), IRCoT (Trivedi et al. 2022a)) and multi-path exploration (CR-Planner (Li et al. 2024), ReARTeR (Sun et al. 2025)); **5)** RL-based methods R1-Searcher (Song et al. 2025) and Search-R1 (Jin et al. 2025a). Qwen2.5-7B-Instruct served as the base model for training, GPT-4o-mini for inference-only tasks, and the text corpus was FlashRAG (Jin et al. 2025b).

Training Setup. CP-Search employs a two-stage training framework: The initial supervised fine-tuning stage utilizes a batch size of 64 and a learning rate of 5×10^{-5} for 10 epochs. This is followed by a reinforcement learning stage configured with a KL divergence coefficient of 0.01, entropy coefficient of 0.03 (Yu et al. 2025), learning rate of 1×10^{-6}, and batch size of 8, sampling 12 trajectories per question. The progressive procedural reward parameters are set as $\alpha = 0.2$ (correctness reward) and $\beta = -0.3$ (error penalty) the default values of γ_+ and γ_- are 1 and -1 respectively. A total of 150 training iterations are carried out.

5.2 Main Result

Table 1 shows the results of CP-Search and other baseline methods on multiple benchmarks. From the data in the table, we can observe that:

Performance Differences Across Methods. There are considerable performance discrepancies between different categories of methods. Methods based on chunk-based perform the worst, followed by those KG-based and sophisticated reasoning mechanisms, which achieve a more balanced performance. Notably, RL-based methods demonstrate superior performance overall. Our proposed method, CP-Search, consistently achieves the best performance across almost all metrics and datasets and achieve 57.3% on the AVG of all datasets.

CP-Search Outperforms RL-based Methods. CP-Search demonstrates a significant advantage over RL-trained methods such as Search-R1 and R1-Searcher: on 2WikiMultiHopQA it achieved an AVG of 72.2%, surpassing R1-Searcher by 7.9%, a gain largely due to enforcing retrieval consistency during training which preserves logical coherence across multi-hop chains and improves identification and use of crucial evidence; additionally, CP-Search attained the highest AVG of 32.8% on MuSiQue, a dataset requiring nuanced reasoning, further validating that embedding retrieval-aware behaviors enables precise contextual understanding and targeted retrieval adjustments for more accurate answers.

Sample-Efficient and Generalizable Retrieval. Another advantage of CP-Search is that it can achieve cold start with small number of samples and endow the model with retrieval cognitive behaviors. In addition, CP-Search demonstrates strong generalization ability. Although HotpotQA and 2WikiMultiHopQA are used for training, the model not only performs well on in-domain datasets but also shows strong generalization ability through good performance on out-of-domain datasets such as Musique and Bamboogle. This indicates that CP-Search has effectively learned how to combine retrieval and reasoning through training, thus achieving robust performance on new complex datasets.

6 Analysis

Ablation Study. To analyze the contribution of each component within CP-Search, we conducted ablation experiments on two datasets: 2WikiMultiHopQA (in-domain) and MuSiQue (out-of-domain). Table 2 summarizes the results. Below we provide a concise, evidence-based interpretation of the ablation outcomes and their implications:

Model	2Wiki		MuSiQue	
	ACC_R	$F1$	ACC_R	$F1$
CP-Search	72.4	67.7	32.2	36.8
w/o Procedural Supervision RL	64.2 (-8.2)	59.7 (-8.0)	25.8 (-6.4)	26.5(-10.3)
w/o Cognitive Behavior	56.0 (-16.4)	43.5 (-24.2)	21.6 (-10.6)	21.0(-15.8)
Procedural Supervision RL				
w/ CPGRPO	72.4	67.7	32.2	36.8
w/ GRPO	68.8 (-3.6)	66.0 (-1.7)	29.8 (-2.4)	35.2 (-1.6)

Table 2: Ablation study on different setting and various reward settings during RL training.

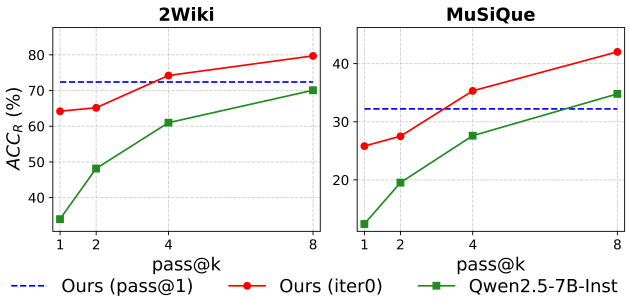

Figure 4: Pass@k of CP-search and base model on the ACC_R. The retrieval condition settings for all methods are the same.

- The empirical results indicate that CP-Search's performance relies critically on the interplay of its components: ablating any single module produces a consistent and measurable drop in performance, which implies that the components contribute complementary capabilities.
- Removing the cognitive-behavior modeling produced the largest degradation, highlighting the pivotal role of behavior-aware design for complex, chained retrieval. This observation also suggests that a relatively small, carefully curated cold-start dataset can be sufficient to impart substantial retrieval and reasoning improvements.
- Further optimization via reinforcement learning amplifies the gains introduced by the supervised behavioral priors. Notably, the CPGRPO variant consistently outperforms standard GRPO, indicating that explicitly optimizing intermediate procedural steps (rather than focusing solely on terminal outcomes) is more effective for complex multi-step retrieval tasks.

RL Analysis. To explore the role of RL in this process, we analyzed the pass@k results of the baseline model (supervised fine-tuned version) and the origin model Qwen2.5-7B-Instruct. As shown in Figure 4, after SFT, the performance will be significantly improved. And the results after RL training fall between pass@2 and pass@4 on both datasets. This indicates that RL optimizes the model's generation process and improves its stability. Furthermore, we found that the pass@8 results were significantly superior,

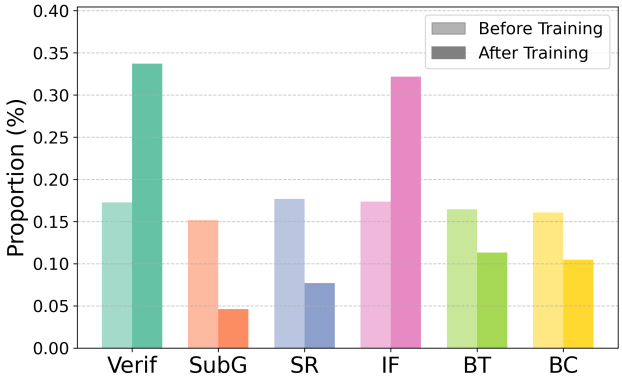

Figure 5: Proportion of retrieval congnitive behaviors before and after training. **Verif**: Verification; **SubG**: Sub-Goal Decomposition; **SR**: Search Refinement; **IF**: Information Filtering; **BT**: Backtracking; **BC**: Backward Chaining.

suggesting that the true enhancement in model capability primarily occurs during the SFT stage, while RL further boosts the model's stability.

Cognitive Behavior Distribution Analysis. As depicted in Figure 5, a distinct evolution in the model's behavioral distribution occurred during RL training. Initially, the model exhibited a relatively diverse range of behaviors. However, as training progressed, there was a pronounced shift towards **Verification** and **Information Filtering** behaviors. Concurrently, the model significantly reduced its engagement in more intricate behaviors such as **Sub-goal Decomposition** and **Search Refinement**. This trend suggests that the training process optimizes the model's retrieval strategy, leading to more efficient and precise information processing.

Conclusion

We introduced CP-Search, a two-stage training framework that significantly enhances LLMs' retrieval for complex, multi-hop reasoning. The first stage, cognitive behavior modeling, uses supervised fine-tuning to embed six crucial retrieval-level cognitive behaviors into the LLM, equipping it with essential reflection processes. In the second stage, we integrated progressive process reward into GRPO, creating the CPGRPO for RL training. This mechanism meticulously evaluates and dynamically rewards or penalizes each retrieval step, ensuring logical consistency and enabling self-correction. Our extensive experiments on multi-hop datasets demonstrate that CP-Search significantly outperforms existing RAG methods.This superior performance highlights how explicitly modeling retrieval-cognitive behaviors, combined with a progressive procedural reward mechanism, enhances the reasoning robustness and adaptive capacity of LLMs in challenging information retrieval scenarios.

Acknowledgments

This study is partially supported by National Key R&D Program of China (2023YFC3502900), National Natural Science Foundation of China (62276082), Shenzhen

Science and Technology Research and Development Fund (KJZD20240903102802003), Shenzhen Science and Technology Research and Development Fund for Sustainable Development Project (GXWD20231128103819001, No.KCXFZ20201221173613036, 20230706140548006) and Guangdong Provincial Key Laboratory (Grant 2023B1212060076).

References

Chen, Z.; Zhao, Z.; Zhu, Z.; Zhang, R.; Li, X.; Raj, B.; and Yao, H. 2024. Autoprm: Automating procedural supervision for multi-step reasoning via controllable question decomposition. *arXiv preprint arXiv:2402.11452*.

Gandhi, K.; Chakravarthy, A.; Singh, A.; Lile, N.; and Goodman, N. D. 2025. Cognitive behaviors that enable self-improving reasoners, or, four habits of highly effective stars. *arXiv preprint arXiv:2503.01307*.

Gao, Y.; Xiong, Y.; Gao, X.; Jia, K.; Pan, J.; Bi, Y.; Dai, Y.; Sun, J.; Wang, H.; and Wang, H. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2(1).

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Guo, Z.; Xia, L.; Yu, Y.; Ao, T.; and Huang, C. 2024. Lightrag: Simple and fast retrieval-augmented generation. *arXiv preprint arXiv:2410.05779*.

Ho, X.; Nguyen, A.-K. D.; Sugawara, S.; and Aizawa, A. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*.

Hochlehnert, A.; Bhatnagar, H.; Udandarao, V.; Albanie, S.; Prabhu, A.; and Bethge, M. 2025. A sober look at progress in language model reasoning: Pitfalls and paths to reproducibility. *arXiv preprint arXiv:2504.07086*.

Huang, Y.; Zhang, S.; and Xiao, X. 2025. KET-RAG: A Cost-Efficient Multi-Granular Indexing Framework for Graph-RAG. arXiv:2502.09304.

Jaech, A.; Kalai, A.; Lerer, A.; Richardson, A.; El-Kishky, A.; Low, A.; Helyar, A.; Madry, A.; Beutel, A.; Carney, A.; et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y. J.; Madotto, A.; and Fung, P. 2023. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12): 1–38.

Jiang, H.; Wu, Q.; Luo, X.; Li, D.; Lin, C.-Y.; Yang, Y.; and Qiu, L. 2023. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. *arXiv preprint arXiv:2310.06839*.

Jin, B.; Zeng, H.; Yue, Z.; Yoon, J.; Arik, S.; Wang, D.; Zamani, H.; and Han, J. 2025a. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Jin, J.; Zhu, Y.; Dou, Z.; Dong, G.; Yang, X.; Zhang, C.; Zhao, T.; Yang, Z.; and Wen, J.-R. 2025b. Flashrag: A modular toolkit for efficient retrieval-augmented generation research. In *Companion Proceedings of the ACM on Web Conference 2025*, 737–740.

Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.-t.; Rocktäschel, T.; et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474.

Li, X.; Xu, W.; Zhao, R.; Jiao, F.; Joty, S.; and Bing, L. 2024. Can we further elicit reasoning in llms? critic-guided planning with retrieval-augmentation for solving challenging tasks. *arXiv preprint arXiv:2410.01428*.

Li, Y.; Dong, B.; Lin, C.; and Guerin, F. 2023. Compressing context to enhance inference efficiency of large language models. *arXiv preprint arXiv:2310.06201*.

Luo, Y.; Yang, Z.; Meng, F.; Li, Y.; Zhou, J.; and Zhang, Y. 2023. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *arXiv preprint arXiv:2308.08747*.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744.

Press, O.; Zhang, M.; Min, S.; Schmidt, L.; Smith, N. A.; and Lewis, M. 2022. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*.

Sanmartin, D. 2024. Kg-rag: Bridging the gap between knowledge and creativity. *arXiv preprint arXiv:2405.12035*.

Shao, Z.; Gong, Y.; Shen, Y.; Huang, M.; Duan, N.; and Chen, W. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. *arXiv preprint arXiv:2305.15294*.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y.; Wu, Y.; et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Shi, Z.; Zhang, S.; Sun, W.; Gao, S.; Ren, P.; Chen, Z.; and Ren, Z. 2024. Generate-then-ground in retrieval-augmented generation for multi-hop question answering. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 7339–7353.

Song, H.; Jiang, J.; Min, Y.; Chen, J.; Chen, Z.; Zhao, W. X.; Fang, L.; and Wen, J.-R. 2025. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*.

Sun, Z.; Wang, Q.; Yu, W.; Zang, X.; Zheng, K.; Xu, J.; Zhang, X.; Song, Y.; and Li, H. 2025. Rearter: Retrieval-augmented reasoning with trustworthy process rewarding. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1251–1261.

Team, K.; Du, A.; Gao, B.; Xing, B.; Jiang, C.; Chen, C.; Li, C.; Xiao, C.; Du, C.; Liao, C.; et al. 2025. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*.

Trivedi, H.; Balasubramanian, N.; Khot, T.; and Sabharwal, A. 2022a. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. *arXiv preprint arXiv:2212.10509*.

Trivedi, H.; Balasubramanian, N.; Khot, T.; and Sabharwal, A. 2022b. MuSiQue: Multihop Questions via Single-hop Question Composition. *Transactions of the Association for Computational Linguistics*, 10: 539–554.

Wang, Z.; Teo, S. X.; Chew, J. J.; and Shi, W. 2025. Instructrag: Leveraging retrieval-augmented generation on instruction graphs for llm-based task planning. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1413–1422.

Xu, F.; Shi, W.; and Choi, E. 2024. RECOMP: Improving retrieval-augmented LMs with context compression and selective augmentation. In *The Twelfth International Conference on Learning Representations*.

Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W. W.; Salakhutdinov, R.; and Manning, C. D. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Yu, Q.; Zhang, Z.; Zhu, R.; Yuan, Y.; Zuo, X.; Yue, Y.; Dai, W.; Fan, T.; Liu, G.; Liu, L.; et al. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

Yue, Y.; Yuan, Y.; Yu, Q.; Zuo, X.; Zhu, R.; Xu, W.; Chen, J.; Wang, C.; Fan, T.; Du, Z.; et al. 2025. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*.

Zeng, W.; Huang, Y.; Liu, Q.; Liu, W.; He, K.; Ma, Z.; and He, J. 2025. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv preprint arXiv:2503.18892*.

Zhang, Y.; Li, M.; Long, D.; Zhang, X.; Lin, H.; Yang, B.; Xie, P.; Yang, A.; Liu, D.; Lin, J.; Huang, F.; and Zhou, J. 2025a. Qwen3 Embedding: Advancing Text Embedding and Reranking Through Foundation Models. arXiv:2506.05176.

Zhang, Y.; Li, Y.; Cui, L.; Cai, D.; Liu, L.; Fu, T.; Huang, X.; Zhao, E.; Zhang, Y.; Chen, Y.; et al. 2025b. Siren's song in the ai ocean: A survey on hallucination in large language models. *Computational Linguistics*, 1–45.