PROCESS-SUPERVISED LEARNING FOR AGENTIC SEARCH VIA DUAL-GRANULARITY ADVANTAGE

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning (RL) has emerged as a promising paradigm for optimizing search agents in complex reasoning tasks. However, traditional outcome-based RL often relies on sparse terminal rewards and suffers from credit assignment challenges, resulting in gradient starvation and high-variance policy updates. This limits optimization stability and can lead to process hallucinations, where agents achieve correct answers via flawed reasoning or redundant retrieval. While recent process-aware approaches attempt to mitigate these issues, they often lack on-policy exploration capability or collapse step-level credit into coarse trajectory-level optimization. To address these limitations, we propose **ProSearch**, a dual-granularity process-supervised RL framework. Leveraging a Monte Carlo Tree Search (MCTS)-derived Process Reward Model (PRM), ProSearch introduces a novel dual-granularity advantage mechanism that combines dense step-level signals with sparse outcome rewards. This provides precise credit assignment for specific actions, mitigating gradient starvation while theoretically improving the Signal-to-Noise Ratio (SNR). Extensive experiments on five multi-hop QA benchmarks demonstrate that ProSearch achieves overall superior performance compared to strong baselines, particularly on complex long-horizon tasks. Further fine-grained analysis reveals that ProSearch not only stabilizes training dynamics but also significantly reduces logical fallacies and redundant exploration.

1 INTRODUCTION

Recent Large Language Models (LLMs) have demonstrated remarkable reasoning capabilities Achiam et al. (2023); Guo et al. (2025), significantly improving their performance on complex, knowledge-intensive tasks. These advances have propelled Retrieval-Augmented Generation (RAG) from static pipelines into agentic search systems, where models act as autonomous decision-makers that actively plan retrieval strategies, interact with external tools, and refine their reasoning based on feedback Li et al. (2025); Guan et al. (2025). However, optimizing such dynamic workflows with standard supervised learning remains challenging. Effective reasoning and retrieval trajectories are often open-ended, making fine-grained ground-truth supervision for intermediate decisions unavailable or expensive. To bridge this gap, Reinforcement Learning (RL) has been increasingly adopted to optimize the model policy from final outcome signals Jin et al. (2025); Song et al. (2025).

However, traditional outcome-based reinforcement learning often suffers from reward sparsity and credit assignment issues, making it inefficient for multi-hop tasks. In such long-horizon trajectories, a coarse-grained scalar reward based on final correctness fails to adequately capture the contributions of intermediate actions. This ambiguity can lead to process hallucinations, where the model is rewarded for reaching the correct answer while relying on flawed reasoning, redundant retrieval, or unverified assumptions Leng et al. (2025). Consequently, outcome-oriented feedback may reinforce spurious correlations rather than authentic reasoning abilities. Furthermore, relying solely on terminal rewards often leads to gradient starvation and high variance, resulting in unstable training.

To address these challenges, recent studies have explored the incorporation of process-level signals via offline or online approaches. Offline methods Zhang et al. (2025); Sun et al. (2025) leverage search algorithms like Monte Carlo Tree Search (MCTS) to synthesize preference pairs for Direct Preference Optimization (DPO), yet their reliance on static data limits adaptation to on-policy distribution shifts. In contrast, online RL methods introduce step-wise guidance but still struggle with

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

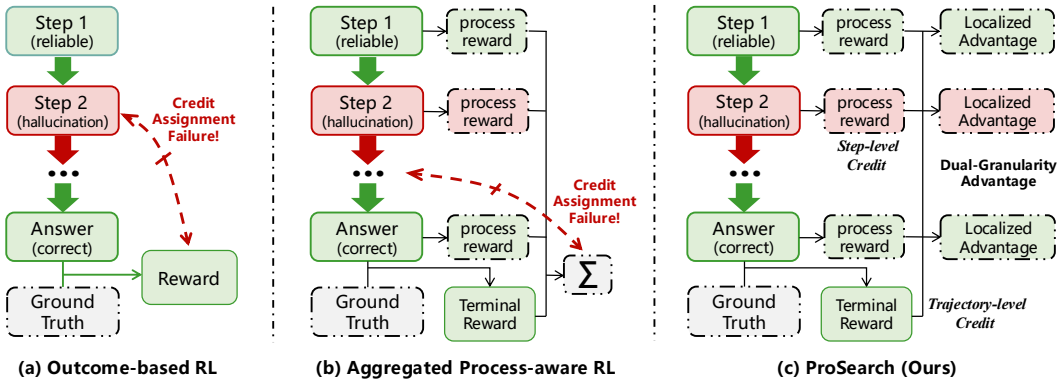


Figure 1: Comparison of optimization paradigms. Unlike outcome-based RL (a) and some existing methods that aggregate step-level rewards into a single return (b), ProSearch (c) enables precise credit assignment via dual-granularity advantages to explicitly penalize process hallucinations.

optimization instability in long-horizon tasks. Specifically, some approaches Wu et al. (2025); He et al. (2025) aggregate step-level feedback into a single trajectory-level return, hindering the policy from assigning credit to critical step-level behaviors. Other methods Zheng et al. (2025); Wang et al. (2025) apply step-wise rewards but rely on bootstrapped advantage estimation, which can propagate noisy signals and lead to high-variance advantage estimates over long reasoning horizons.

In this paper, we propose **ProSearch**, an online process-supervised reinforcement learning framework for long-horizon agentic search. Unlike previous approaches, ProSearch integrates dense process signals and sparse outcome rewards into an online optimization loop. Specifically, we first introduce an **MCTS-driven Process Reward Modeling** stage that formulates step-level evaluation as a localized preference-ranking task. By expanding a search tree and contrasting sibling actions, we construct preference pairs to train a reliable fine-grained evaluator. Then, we optimize the agentic policy via a **Dual-Granularity Process-Supervised RL** stage. To achieve localized advantage shaping, ProSearch combines sparse outcome signals with dense process advantages derived from the evaluator. By providing informative credit assignment for each action, this formulation theoretically improves the Signal-to-Noise Ratio (SNR) and empirically mitigates gradient starvation.

We conduct extensive experiments on five challenging multi-hop QA benchmarks. Empirical results demonstrate that ProSearch achieves overall superior performance, with particularly notable gains in complex long-horizon tasks. Beyond performance, fine-grained trajectory evaluation indicates that ProSearch improves agentic behaviors by mitigating logic fallacies, reducing redundant search steps, and increasing valid reasoning paths. Furthermore, optimization dynamics analysis shows that our dual-granularity advantage mechanism effectively restricts reward volatility and maintains stable gradient updates. Our key contributions can be summarized as follows:

- We propose ProSearch, an online dual-granularity process-supervised RL framework that explicitly aggregates localized step-level process advantages with global outcome signals, which address credit assignment and mitigate gradient starvation in long-horizon policy optimization.
- We introduce an MCTS-driven Process Reward Modeling pipeline that curates high-quality preference pairs, leading to a reliable process reward model for dense step-level supervision.
- We theoretically and empirically demonstrate that our localized advantage shaping guarantees a higher SNR, thereby improving multi-hop reasoning performance.

2 Related Work

Retrieval-Augmented Generation. Retrieval-Augmented Generation (RAG) mitigates LLM hallucinations by integrating external knowledge Lewis et al. (2020). While early RAG relied on static retrieve-then-generate pipelines Guu et al. (2020), the demand for complex multi-hop reasoning drove the development of iterative Shao et al. (2023); Trivedi et al. (2023) and active retrieval strategies Jiang et al. (2023); Asai et al. (2023). However, these approaches remain constrained

by predefined heuristics, limiting their adaptability in open-ended tasks. Consequently, recent research has pivoted towards Agentic Search, formulating retrieval as a multi-step decision-making process Gao et al. (2023). Pioneered by frameworks like ReAct Yao et al. (2022), models now act as autonomous agents that explicitly interleave reasoning traces with search tool interactions. Advanced systems like Search-o1 Li et al. (2025) and DeepRAG Guan et al. (2025) further scale this paradigm to long-horizon queries. Since relying solely on supervised fine-tuning (SFT) or complex prompting restricts their generalization to novel scenarios, recent research has increasingly shifted towards Reinforcement Learning (RL) to optimize these agents.

Reinforcement Learning for Agentic Search. While RL has advanced agentic search by optimizing tool use and reasoning Jin et al. (2025); Song et al. (2025), traditional outcome-supervised methods struggle with sparse rewards and process hallucinations in long-horizon tasks. Recent works address this via process-level supervision. Offline methods Zhang et al. (2025); Sun et al. (2025); Leng et al. (2025) synthesize preference data via MCTS for DPO, but their reliance on static datasets limits on-policy adaptation. Conversely, online methods employ proxy-based evaluation Wu et al. (2025); Xu et al. (2025); Zheng et al. (2025); Wen et al. (2026), multi-agent interactions Wang et al. (2025); He et al. (2025), or tree-search Zhang et al. (2026) for step-wise feedback. However, these online methods often suffer from optimization instability, as they either aggregate intermediate feedback into coarse trajectory-level returns or rely on noisy bootstrapped advantage estimation. In contrast, ProSearch employs a dual-granularity process-supervised RL framework that decouples step-level supervision from global outcomes, enabling precise credit assignment.

3 PROSEARCH

3.1 PROBLEM FORMULATION

We formulate agentic search as an interactive decision-making process. Given a user query q, the agent interacts with a retrieval environment over T steps to derive a final answer y. Let s_t denote the accumulated context at step t, initialized as $s_1 = q$. At each step, the policy π_θ generates an action a_t based on s_t, where a_t comprises internal reasoning and a command (e.g., a search query). Composed of a sequence of discrete tokens $(o_{t,1}, \ldots, o_{t,|a_t|})$, the action is generated auto-regressively:

$$P_\theta(a_t \mid s_t) = \prod_{k=1}^{|a_t|} P_\theta(o_{t,k} \mid s_t, o_{t,<k}). \tag{1}$$

Following the action, the environment returns an observation e_t (e.g., retrieved documents). Note that e_t can be empty ($e_t = \emptyset$) if the action does not require external retrieval. The context is then updated via concatenation as $s_{t+1} = s_t \oplus a_t \oplus e_t$.

This iterative process constructs a reasoning trajectory $\tau = (a_1, e_1, \ldots, a_T)$. The trajectory terminates at step T with a final non-retrieval action a_T, from which the definitive answer y to the query q is extracted. The overall generative probability of the trajectory is formulated as:

$$P_\theta(\tau \mid q) = \prod_{t=1}^{T} P_\theta(a_t \mid s_t). \tag{2}$$

3.2 MCTS-DRIVEN PROCESS REWARD MODELING

In long-horizon agentic search, optimizing policies solely on sparse terminal rewards exacerbates credit assignment issues and often leads to process hallucinations. To obtain a reliable fine-grained evaluator, we propose an **MCTS-driven Process Reward Modeling** stage. To circumvent the scale miscalibration in absolute scoring, we formulate step-level evaluation as a localized preference ranking task. Specifically, we first warm up a base policy to enforce a structured reasoning format. We then leverage this policy as a prior to systematically explore the search space via MCTS. Finally, we extract contrastive sibling pairs from the search tree to train a robust step-level evaluator.

Supervised Policy Warmup. To effectively explore the complex reasoning space for PRM training data collection, we first establish a competent and stable policy prior π_{sft} via supervised fine-

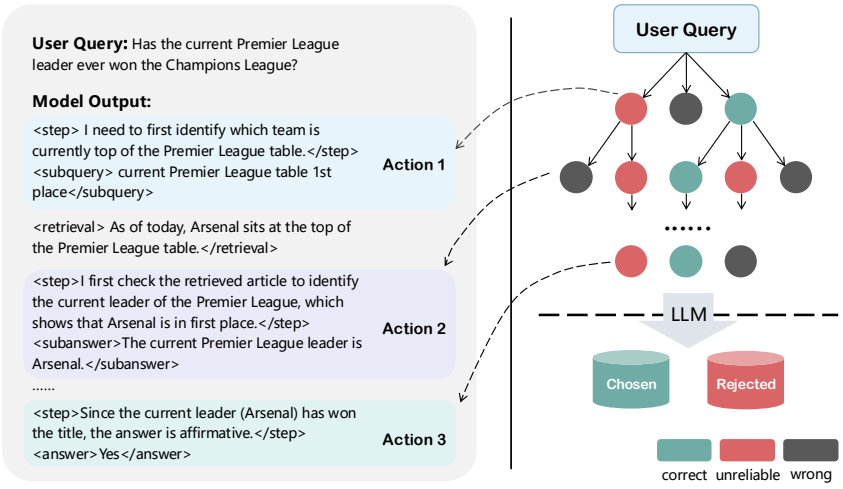


Figure 2: The left panel presents the reasoning-action schema enforced during policy warmup, while the right shows the MCTS mechanism used to collect preference pairs for Process Reward Modeling.

tuning on synthesized demonstrations. This phase enforces a structured reasoning-action schema (illustrated in the left panel of Figure 2) to provide the baseline for the subsequent MCTS exploration. Detailed dataset construction and the format-aware SFT objective are presented in Appendix B.1.

MCTS-based Trajectory Exploration. To generate a rich dataset that distinguishes valid logic from spurious correlations, we employ π_{sft} as the policy prior and utilize MCTS to systematically explore the solution space. A node in the search tree corresponds to the accumulated context s_t, and an edge denotes a generated reasoning step a_t. During the selection phase, we traverse the tree using the Predictor Upper Confidence Bound (PUCT) criterion, which balances the exploitation of high-value paths and the exploration of uncertain paths:

$$a_t = \arg\max_a \left(Q(s_t, a) + c_{\text{puct}} \cdot \pi_{\text{sft}}(a \mid s_t) \frac{\sqrt{\sum_{a'} N(s_t, a')}}{1 + N(s_t, a)} \right). \tag{3}$$

Upon reaching a leaf node, we expand the tree by sampling candidate actions from π_{sft}. We then simulate rollouts and evaluate the final answer against the ground truth to obtain an outcome reward v, and backpropagate this scalar along the search path to update the expected action values Q:

$$Q(s_t, a_t) \leftarrow \frac{Q(s_t, a_t) \cdot N(s_t, a_t) + \gamma^{T-t} \cdot v}{N(s_t, a_t) + 1}, \tag{4}$$

where $\gamma \in (0, 1]$ is a discount factor to penalize redundant trajectories.

Process Reward Modeling. While MCTS maps the solution space, the derived Q-values remain inherently outcome-biased: a logically flawed intermediate step may still accumulate a high Q-value if it serendipitously leads to a correct terminal answer. To train a true PRM that rigorously evaluates fine-grained process validity rather than just global correctness, we extract contrastive step-level preference pairs directly from the search tree. Specifically, we select sibling nodes branching from identical parent contexts s_t but differing in their current actions a_t, and employ an oracle LLM (e.g., GPT-4o) to strictly adjudicate their logical validity. This yields a reliable preference dataset $\mathcal{D} = \{(s_t, a^+, a^-)\}$, where a^+ and a^- denote the logically sound and flawed sibling actions, respectively. Implementation and verification details are provided in Appendix B.2.

Finally, we parameterize the Process Reward Model $\mathcal{R}_\phi$ to map a given context-action pair to a scalar reward. We optimize $\mathcal{R}_\phi$ following the Bradley-Terry ranking objective to maximize the score margin between chosen and rejected steps:

$$\mathcal{L}_{\text{PRM}} = -\mathbb{E}_{(s_t, a^+, a^-) \sim \mathcal{D}} \left[\log \sigma \left(\mathcal{R}_\phi(s_t, a^+) - \mathcal{R}_\phi(s_t, a^-) \right) \right], \tag{5}$$

By minimizing this loss, $\mathcal{R}_\phi$ abstracts the MCTS exploration into a dense, reliable verifier, providing the step-level process supervision required for the subsequent stage.

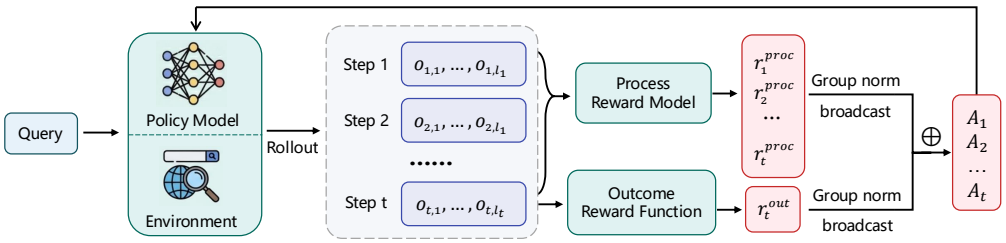


Figure 3: The framework of Process-Supervised Reinforcement Learning. ProSearch aggregates group-normalized advantages from both step-level process signals and final outcome rewards to provide dense supervision.

3.3 Dual-Granularity Process-Supervised Reinforcement Learning

While the initial SFT policy π_{sft} provides a basic structural prior, it fundamentally mimics static demonstrations rather than actively exploring strategies in open-ended environments. To enable autonomous planning and reasoning, we formalize the agentic search as an RL problem following the definitions in Section 3.1. However, in complex multi-hop scenarios, relying solely on sparse terminal rewards introduces credit assignment issues. To address this, we propose a dual-granularity process-supervised RL framework. Specifically, we first refine the initial policy via PRM to bridge the alignment gap, and then optimize it using Group Relative Policy Optimization (GRPO) Shao et al. (2024) augmented with a novel dual-granularity advantage mechanism.

Policy Initialization via PRM-Guided Refinement. Directly optimizing an unaligned policy via RL typically suffers from training instability due to its misalignment with the preference space of PRM. To bridge the gap between the initial policy π_{sft} and our evaluator $\mathcal{R}_\phi$, we introduce a brief Reasoning Refinement stage. We sample N candidate trajectories per query from π_{sft} and curate a refined dataset by retaining actions that both satisfy a process validity threshold ($\mathcal{R}_\phi(s_t, a_t) > 0$) and ultimately lead to the correct answer. By fine-tuning on these trajectories, we distill high-value signals from the PRM into a refined policy π_{rft}, serving as a stable starting point for subsequent RL.

Dual-Granularity Reward Formulation. Following refinement, we optimize the policy via online RL. For a given query q, we sample a group of G independent trajectories $\mathcal{T} = \{\tau_1, \ldots, \tau_G\}$ from the current policy π_θ. We formulate rewards at two distinct granularities:

- **Process Reward (Step-level):** Upon the completion of each intermediate step $a_{i,t}$, we compute a dense reward $r_{i,t}^{\text{proc}}$ using the frozen PRM $\mathcal{R}_\phi$ to assess fine-grained logical validity:

$$r_{i,t}^{\text{proc}} = \mathcal{R}_\phi(s_{i,t}, a_{i,t}) + \nu_1 \cdot \mathbb{I}_{\text{format}}(a_{i,t}), \tag{6}$$

  where $s_{i,t}$ is the accumulated context, and $\mathbb{I}_{\text{format}}(\cdot) \in \{0, 1\}$ penalizes format errors.
- **Outcome Reward (Trajectory-level):** At the end of trajectory τ_i, we extract the final answer $\hat{y}_i$ and assign a sparse outcome reward r_i^{out} to evaluate global utility:

$$r_i^{\text{out}} = \text{F1}(\hat{y}_i, y^*) + \nu_2 \cdot \mathbb{I}_{\text{traj}}(\tau_i), \tag{7}$$

  where $\text{F1}(\cdot)$ computes the accuracy against ground truth y^*, and $\mathbb{I}_{\text{traj}}(\cdot)$ verifies format.

Advantage Estimation. To optimize the policy, we need to assign advantages to individual tokens. Building upon GRPO, we leverage group-wise statistics to approximate the baseline. We independently normalize the outcome and process rewards across the sampled group $\mathcal{T}$:

$$A_i^{\text{out}} = \frac{r_i^{\text{out}} - \mu^{\text{out}}}{\sigma^{\text{out}}}, \quad A_{i,t}^{\text{proc}} = \frac{r_{i,t}^{\text{proc}} - \mu^{\text{proc}}}{\sigma^{\text{proc}}}, \tag{8}$$

where (μ, σ) denotes the mean and standard deviation of the rewards within the group.

For token-level credit assignment, rather than recursively discounting future rewards that can compound variance over long generations, we adopt a localized shaping approach. We define the **total**

Table 1: Overall results of ProSearch and baselines on five benchmarks. The best and second-best scores are highlighted in **bold** and underlined. † indicates statistically significant improvements over Search-R1, the strongest baseline in average F1, with $p < 0.05$.

Method	PopQA		HotpotQA		2Wiki		MuSiQue		Bamboogle		Avg.	
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
Standard Baselines												
Naïve Generation	11.7	16.6	9.8	19.0	15.8	21.0	2.1	9.5	4.8	11.0	8.8	15.4
Standard RAG	27.0	35.9	20.0	31.5	16.1	23.0	3.3	10.3	12.0	20.1	15.7	24.2
Advanced Baselines												
Iter-RetGen	32.9	37.8	15.9	21.2	9.3	11.5	4.8	8.5	17.6	23.7	16.1	20.5
IRCoT	33.6	42.2	27.2	37.9	25.1	30.4	5.1	13.1	16.8	28.6	21.6	30.4
FLARE	39.3	46.3	28.9	40.1	24.1	29.5	5.8	13.5	27.2	37.8	25.1	33.4
Search-o1	38.4	43.9	28.6	38.1	22.6	27.9	13.0	19.9	38.4	50.1	28.2	36.0
Reinforcement Learning Baselines												
Search-R1	45.9	50.1	**42.7**	**54.6**	39.9	45.2	20.6	29.5	43.2	54.1	38.5	46.7
ReasonRAG	41.5	46.2	38.4	48.9	43.6	50.4	12.8	20.6	36.0	45.5	34.5	42.3
HiPRAG	34.3	42.7	36.8	49.1	37.0	43.2	15.6	25.0	39.2	51.6	32.6	42.3
StepSearch	39.8	45.0	34.0	46.4	33.5	41.0	20.2	30.4	44.0	54.0	34.3	43.4
ProSearch (Ours)	**47.2**†	**51.6**†	41.4	52.8	**46.0**†	**51.1**†	**23.5**†	**34.1**†	**45.6**	**56.4**	**40.7**†	**49.2**†

advantage for the k-th token $o_{i,t,k}$ within step $a_{i,t}$ by aggregating the global and step-level signals:

$$A_{i,t,k} = A_i^{\text{out}} + \beta \cdot A_{i,t}^{\text{proc}}. \tag{9}$$

From an optimization perspective, this aggregation naturally balances the bias-variance tradeoff in RL. While the outcome advantage A_i^{out} provides a global estimate of the overall utility, its sparse and delayed nature drowns out the valid gradient signal over long reasoning chains. Conversely, introducing the process advantage as a localized shaping signal effectively boosts the correct update direction, guaranteeing a higher SNR for the policy gradient estimator. A formal mathematical analysis is provided in Appendix A.

Policy Optimization. Finally, we optimize the policy π_θ by maximizing the clipped advantage-weighted policy objective. The final loss function is formulated as:

$$\mathcal{L}_{\text{RL}}(\theta) = -\mathbb{E}_{q\sim\mathcal{D},\mathcal{T}\sim\pi_{\text{old}}}\left[\frac{1}{G}\sum_{i=1}^{G}\sum_{t=1}^{T_i}\sum_{k=1}^{|a_{i,t}|}\min\Big(\rho_{i,t,k}A_{i,t,k},\ \text{clip}(\rho_{i,t,k},1-\varepsilon,1+\varepsilon)A_{i,t,k}\Big)\right], \tag{10}$$

where $\rho_{i,t,k} = \frac{\pi_\theta(o_{i,t,k}|s_{i,t,k})}{\pi_{\text{old}}(o_{i,t,k}|s_{i,t,k})}$ represents the probability ratio, $s_{i,t,k}$ encompasses the full context preceding the generated token, and ε is the clipping coefficient. To prevent policy degradation, we augment $\mathcal{L}_{\text{RL}}(\theta)$ with an approximate token-level KL penalty $\mathbb{D}_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}})$.

4 Experiments

4.1 Dataset and Metrics

We evaluate ProSearch and all baseline models on five diverse QA benchmarks, including PopQA Mallen et al. (2023), HotpotQA Yang et al. (2018), 2WikiMultiHopQA Ho et al. (2020), MuSiQue Trivedi et al. (2022), and Bamboogle Press et al. (2023). For these benchmarks, we report Exact Match (EM) and F1 Score, and use paired t-test ($p < 0.05$) to assess the significance of improvements over Search-R1. For process analysis, we employ DeepSeek-V3.2 Liu et al. (2025) to strictly assess the entire reasoning trajectory, classifying it into: Correct, Fact Hallucination, Logical Fallacy, or Redundant Steps (Details are in Appendix C.1). Furthermore, we evaluate the reliability of the learned process reward model in Appendix D.3.

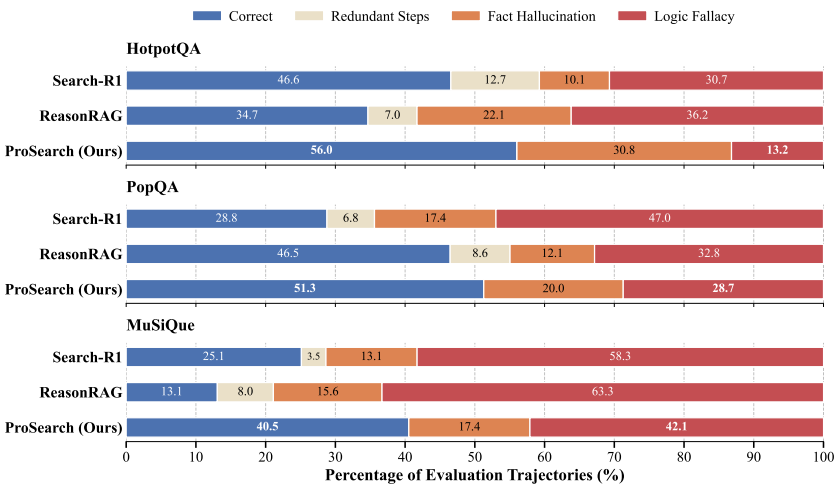


Figure 4: Fine-grained analysis of reasoning trajectories on HotpotQA, PopQA, and MuSiQue. Trajectories are classified into four categories by LLM. Compared to baselines, ProSearch significantly increases the proportion of strictly correct reasoning paths (blue) while effectively mitigating process hallucinations caused by logical fallacies (red) and reducing redundant exploration (beige).

4.2 Baselines

We benchmark ProSearch against 10 methods across three categories: (1) **Standard Baselines** represent the non-iterative lower bounds, including Naïve Generation (relying solely on parametric knowledge) and Standard RAG Lewis et al. (2020) (single-step retrieve-then-generate). (2) **Advanced Baselines** encompass inference-time dynamic frameworks: Iter-RetGen Shao et al. (2023) and IRCoT Trivedi et al. (2023) progressively interleave retrieval and reasoning; FLARE Jiang et al. (2023) actively triggers search based on confidence; and Search-o1 Li et al. (2025) acts as an autonomous agent explicitly controlling the workflow. (3) **RL-based Baselines** represent state-of-the-art policy optimization: Search-R1 Jin et al. (2025) optimizes via outcome-based PPO Schulman et al. (2017); ReasonRAG Zhang et al. (2025) performs offline DPO using MCTS-synthesized preference data; and HiPRAG Wu et al. (2025) utilizes hierarchical process rewards for online RL; and StepSearch Zheng et al. (2025) applies step-wise PPO to optimize intermediate search decisions. Detailed configurations are provided in Appendix C.2.

4.3 Implementation Details

We implement ProSearch using Qwen3-8B Yang et al. (2025) as the backbone policy. For the retrieval environment, we employ the 2018 Wikipedia dumpKarpukhin et al. (2020) and E5-base Wang et al. (2022) to retrieve the top-3 documents. For Supervised Policy Warmup, we perform full-parameter fine-tuning on 109k MuSiQue context-action pairs. For Process Reward Modeling, we run MCTS on 728 queries to collect 8,255 contrastive pairs labeled by GPT-4o. For Refinement, we fine-tune the policy on 105k PRM-filtered high-quality trajectories. For RL, we apply LoRA to optimize the policy on 10k held-out queries with a group size of $G = 8$ and a dual-granularity weight of $\beta = 0.3$. More details are provided in Appendix C.3.

4.4 Main Results

Table 1 compares ProSearch with ten baselines. Our key findings include:

- **Necessity of Policy Optimization:** RL-based methods consistently outperform advanced inference-time frameworks such as Search-o1, highlighting the benefit of explicit policy optimization for open-ended multi-hop reasoning.
- **Advantage in Long-Horizon Reasoning:** ProSearch outperforms both outcome-based Search-R1 and step-wise PPO-based StepSearch, with clear gains on challenging multi-hop datasets.

Table 2: Ablation study of ProSearch. **Component ablations** evaluate the effects of the refinement and RL stages, while **credit assignment under GRPO** compares outcome-only optimization with aggregated process rewards.

Method	PopQA		HotpotQA		2Wiki		MuSiQue		Bamboogle		Avg.	
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
ProSearch	47.2	51.6	**41.4**	**52.8**	**46.0**	**51.1**	23.5	**34.1**	**45.6**	**56.4**	**40.7**	**49.2**
Component ablations												
w/o Refinement	**47.3**	**51.7**	38.0	49.7	43.2	48.6	23.2	32.8	39.2	52.2	38.2	47.0
w/o RL	32.1	36.9	36.8	48.3	38.9	44.4	**23.6**	33.0	39.2	48.7	34.1	42.3
SFT Policy	27.9	32.7	28.8	38.8	29.0	33.3	23.3	32.5	36.0	46.8	29.0	36.8
Credit assignment under GRPO												
Outcome only	30.2	34.8	33.9	44.8	38.6	43.1	23.5	32.4	38.4	49.9	32.9	41.0
Aggregated PRM	47.1	51.6	41.0	52.5	40.7	46.7	22.2	32.5	44.8	55.1	39.1	47.7

Compared with Search-R1, ProSearch improves F1 score by +5.9 on 2Wiki and +4.6 on MuSiQue, indicating that dual-granularity process supervision is effective for long-horizon reasoning.

- **Out-of-Domain Generalization:** ProSearch maintains strong performance across different data distributions and achieves significant gains on unseen benchmarks such as PopQA and 2Wiki, suggesting improved transfer beyond the training distribution.

4.5 Process Analysis

To better understand how our dual-granularity advantage mechanism affects reasoning quality, we analyze fine-grained reasoning trajectories using the evaluation protocol described in Section 4.1. As illustrated in Figure 4, our key observations are:

- **Improved Reasoning Quality:** ProSearch yields a substantial increase in entirely valid reasoning paths. For instance, this proportion on MuSiQue rises from 25.1% to 40.5%, demonstrating that our performance gains are accompanied by improvements in multi-step reasoning quality.
- **Reduction of Logical Fallacies:** ProSearch reduces the Logic Fallacy trajectories across three datasets, e.g., from 30.7% to 13.2% on HotpotQA compared with Search-R1. This is consistent with the intended role of PRM signals in discouraging flawed intermediate reasoning.
- **Less Redundant Exploration:** ProSearch consistently decreases the Redundant Steps, indicating that our localized advantage shaping guides the agent to learn more efficient retrieval strategies.

4.6 Ablation Studies

To disentangle the contributions of different training stages and credit-assignment strategies, we conduct two groups of ablation studies in Table 2: (1) component ablations, which examine the effects of PRM-guided refinement and online RL; and (2) credit-assignment ablations, which compare different ways of incorporating process supervision into GRPO.

Effect of Training Components. Starting from the SFT policy, PRM-guided offline refinement substantially improves the average F1 from 36.8 to 42.3, demonstrating that filtering trajectories with process-level supervision provides a stronger policy initialization. However, removing online RL still results in a large gap to the full ProSearch (42.3 vs. 49.2 Avg. F1), indicating that static refinement alone is insufficient for learning effective search strategies in an interactive environment. Meanwhile, directly applying process-supervised RL to the SFT policy (*w/o Refinement*) achieves 47.0 Avg. F1, but remains 2.2 points below ProSearch. This result suggests that PRM-guided refinement provides a better initialization for subsequent online optimization.

Effect of Credit Assignment. We further isolate how process signals should be incorporated during online RL. Outcome-only GRPO achieves 41.0 Avg. F1, whereas introducing PRM-based pro-

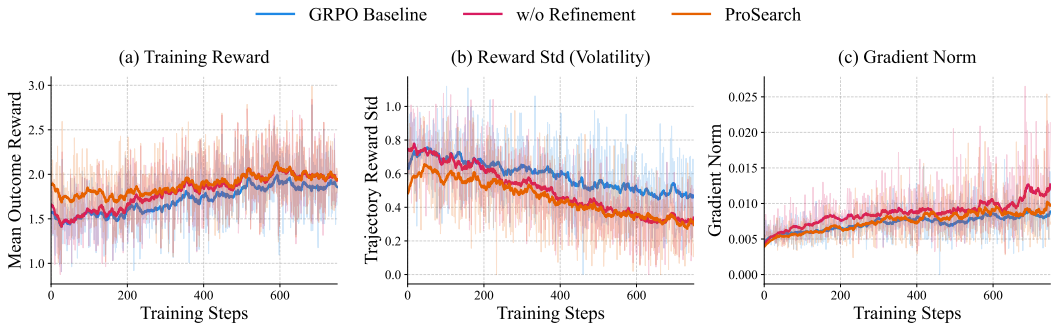


Figure 5: Training dynamics during the RL phase. We compare ProSearch (orange) against the outcome-based GRPO baseline (blue) and the unrefined variant without PRM alignment (red). Subfigures illustrate the progression of (a) Mean Outcome Reward, (b) Trajectory Reward Standard Deviation, and (c) Gradient Norm.

cess rewards with trajectory-level aggregation improves performance to 47.7. This substantial gain confirms the value of process supervision beyond sparse terminal rewards. ProSearch further improves the average F1 to 49.2 by assigning localized process advantages to individual reasoning steps while retaining the trajectory-level outcome signal. The improvement is particularly evident on 2Wiki, where ProSearch achieves 51.1 F1 compared with 46.7 for aggregated PRM, supporting the effectiveness of dual-granularity credit assignment for long-horizon reasoning.

4.7 Optimization Dynamics

To explicitly understand how our dual-granularity advantage mechanism improves the learning process, we analyze the training dynamics. Figure 5 illustrates the progression of three key metrics during the RL phase: (a) Training F1 Score / Mean Outcome Reward (`reward`), (b) Trajectory Reward Standard Deviation (`reward_std`), and (c) Gradient Norm (`grad_norm`), comparing ProSearch against the *GRPO baseline* and the unrefined variant (*w/o Refinement*).

Mitigating Gradient Starvation via Localized Shaping. In long-horizon tasks, relying solely on sparse terminal rewards leaves intermediate steps without direct feedback. As shown in Figures 5, this induces blind exploration with high trajectory volatility (high `reward_std`) and gradient starvation (stagnant `grad_norm`) in the GRPO baseline. In contrast, by aggregating dense process advantages as a localized shaping signal, ProSearch not only sustains a robust `grad_norm` with clear directional updates, but also reduces `reward_std`. This empirical drop in volatility validates our theoretical analysis (Appendix A) that process supervision effectively restricts noise accumulation, ensuring policy updates are driven by high-SNR signals to generate reliable reasoning chains.

Stabilizing Training Dynamics via Refinement. As shown in Figure 5, directly exposing the unaligned SFT policy to process-supervised RL (*w/o Refinement*) induces erratic spikes in the `grad_norm`. This instability occurs because the naive policy frequently encounters reward fluctuations when first exploring the unconstrained reasoning space, leading to abrupt and potentially destructive parameter shifts. By pre-aligning the policy with the PRM's preference space, the Refinement stage eliminates these drastic gradient shocks, enforcing the smooth and consistent policy updates necessary for stable long-horizon optimization.

5 Conclusion

In this paper, we propose ProSearch, a process-supervised reinforcement learning framework designed to address the credit assignment problem in long-horizon agentic search. Leveraging a MCTS-derived PRM, ProSearch introduces a novel dual-granularity advantage mechanism that combines dense step-level signals with sparse outcome rewards to provide localized advantage shaping. Our theoretical analysis establishes a higher SNR under the stated assumptions, while empirical results demonstrate improved optimization stability and reasoning quality.

AI USE STATEMENT

Generative AI models were used in several stages of the research workflow. Specifically, LLMs were used to synthesize demonstrations for supervised policy warmup, assess the logical validity of MCTS-derived candidate actions, and support fine-grained trajectory analysis. These experimental uses are described in the main text. In addition, generative AI tools were used for language editing and to improve the readability of the manuscript. The authors reviewed all AI-assisted text and take responsibility for the final manuscript and all artifacts.

REPRODUCIBILITY STATEMENT

Detailed descriptions of the training procedure, experimental setup, and hyperparameters are provided in the main text and appendix. Code and model checkpoints will be released upon publication.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*, 2023.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, Haofen Wang, et al. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2(1):32, 2023.

Xinyan Guan, Jiali Zeng, Fandong Meng, Chunlei Xin, Yaojie Lu, Hongyu Lin, Xianpei Han, Le Sun, and Jie Zhou. Deeprag: Thinking to retrieve step by step for large language models. *arXiv preprint arXiv:2502.01142*, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *International conference on machine learning*, pp. 3929–3938. PMLR, 2020.

Jie He, Victor Gutiérrez-Basulto, and Jeff Z Pan. From sufficiency to reflection: Reinforcement-guided thinking quality in retrieval-augmented reasoning for llms. *arXiv preprint arXiv:2507.22716*, 2025.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, 2020.

Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 7969–7992, 2023.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pp. 6769–6781, 2020.

Yongqi Leng, Yikun Lei, Xikai Liu, Meizhi Zhong, Bojian Xiong, Yurong Zhang, Yan Gao, Yao Hu, Deyi Xiong, et al. Decex-rag: Boosting agentic retrieval-augmented generation with decision and execution optimization via process supervision. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 1412–1425, 2025.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 5420–5438, 2025.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 9802–9822, 2023.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711, 2023.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 9248–9274, 2023.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*, 2025.

Zhongxiang Sun, Qipeng Wang, Weijie Yu, Xiaoxue Zang, Kai Zheng, Jun Xu, Xiao Zhang, Yang Song, and Han Li. Rearter: Retrieval-augmented reasoning with trustworthy process rewarding. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1251–1261, 2025.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 10014–10037, 2023.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`, 2020.

Junlin Wang, Zehao Wu, Shaowei Lu, Yanlan Li, and Xinghao Huang. Sirag: Towards stable and interpretable rag with a process-supervised multi-agent framework. *arXiv preprint arXiv:2509.18167*, 2025.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

Tongyu Wen, Guanting Dong, and Zhicheng Dou. Smartsearch: Process reward-guided query refinement for search agents. *arXiv preprint arXiv:2601.04888*, 2026.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pp. 38–45, 2020.

Peilin Wu, Mian Zhang, Kun Wan, Wentian Zhao, Kaiyu He, Xinya Du, and Zhiyu Chen. Hiprag: hierarchical process rewards for efficient agentic retrieval augmented generation. *arXiv preprint arXiv:2510.07794*, 2025.

Peiran Xu, Zhuohao Li, Xiaoying Xing, Guannan Zhang, Debiao Li, and Kunyu Shi. Hybrid reward normalization for process-supervised non-verifiable agentic tasks. *arXiv preprint arXiv:2509.25598*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Tianhua Zhang, Kun Li, Junan Li, Yunxiang Li, Hongyin Luo, Xixin Wu, James Glass, and Helen Meng. Treeps-rag: Tree-based process supervision for reinforcement learning in agentic rag. *arXiv preprint arXiv:2601.06922*, 2026.

Wenlin Zhang, Xiangyang Li, Kuicai Dong, Yichao Wang, Pengyue Jia, Xiaopeng Li, Yingyi Zhang, Derong Xu, Zhaocheng Du, Huifeng Guo, et al. Process vs. outcome reward: Which is better for agentic rag reinforcement learning. *arXiv preprint arXiv:2505.14069*, 2025.

Xuhui Zheng, Kang An, Ziliang Wang, Yuhang Wang, and Yichao Wu. Stepsearch: Igniting llms search ability via step-wise proximal policy optimization. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 21816–21841, 2025.

A Theoretical Insights

This section provides a theoretical justification for why our dual-granularity advantage mechanism mitigates the credit assignment problem. We formalize the **Gradient Starvation** issue in outcome-based RL for long-horizon policy optimization, and demonstrate how ProSearch mathematically guarantees a higher Signal-to-Noise Ratio (SNR) for policy optimization.

Problem Formulation. Building on Section 3.1, we model the agentic search as a Markov Decision Process (MDP) over a horizon T. To optimize the policy, any RL algorithm fundamentally requires an advantage estimator $\hat{A}(s_t, a_t)$ to approximate the true advantage $A^*(s_t, a_t)$, which reflects the relative utility of action a_t at step t. The fidelity of this estimation is governed by its **Mean Squared Error (MSE)**:

$$\begin{aligned} \text{MSE}\left(\hat{A}\right) &= \mathbb{E}\left[\left(\hat{A}(s_t, a_t) - A^*(s_t, a_t)\right)^2\right] \\ &= \text{Bias}^2(\hat{A}) + \text{Var}(\hat{A}). \end{aligned} \tag{11}$$

In standard agentic search, the environment lacks intermediate signals and solely provides a sparse terminal reward R_T. This delayed feedback poses a challenge, formalized below.

Lemma 1 (Horizon-Induced Error in Outcome-based Estimators Schulman et al. (2015))**.** *Let $\hat{A}_{out}(s_t, a_t) = A^*(s_t, a_t) + e_{out}$ be an outcome-based advantage estimator with error e_{out}. To estimate the utility of an early action a_t, terminal signals must propagate backward over an effective horizon $H = T - t$:*

- ***Monte Carlo Estimators:*** *These estimators are unbiased (Bias $= 0$) but suffer from cumulative Markovian noise. The variance scales linearly with the remaining horizon: Var$(\hat{A}_{out}) = \mathcal{O}(H)$.*
- ***Actor-Critic Bootstrapping:*** *These estimators attempt to reduce variance by utilizing a learned Value Network $V_\phi(s)$ via Generalized Advantage Estimation (GAE). However, in sparse-reward tasks, compounding Temporal Difference (TD) approximation errors lead to a worst-case bias that scales quadratically with the effective horizon: Bias$^2(\hat{A}_{out}) = \mathcal{O}(H^2)$.*

In either paradigm, the total estimation error MSE$(\hat{A}_{out}) = \mathbb{E}[e_{out}^2] \triangleq E_{out}^2(H)$ monotonically increases with the remaining horizon H.

Assumption 1 (Bounded Error of the Process Evaluator)**.** *In contrast, ProSearch's Process Reward Model (PRM) yields a dense step-level estimate $\hat{A}_{proc}(s_t, a_t) = A^*(s_t, a_t) + e_{proc}$. Because the PRM evaluates logical validity strictly based on the local context (s_t, a_t), its estimation error e_{proc} is decoupled from the future trajectory rollout. Thus, we assume its error is bounded by a neural variance constant σ_P^2, which is independent of T:*

$$MSE(\hat{A}_{proc}) = \mathbb{E}[e_{proc}^2] \leq \sigma_P^2. \tag{12}$$

To evaluate optimization stability, we analyze the **SNR** of the advantage estimation (the ratio of the squared true signal to the MSE of the noise).

Proposition 2 (SNR Enhancement via Dual-Granularity Advantage)**.** *Under Lemma 1 and Assumption 1, ProSearch provides a strictly higher SNR than outcome-only estimators (SNR$_{ProSearch}$ > SNR$_{base}$) whenever the horizon-induced outcome error $E_{out}^2(H)$ satisfies $E_{out}^2(H) > \frac{\beta}{2+\beta}\sigma_P^2$.*

Proof. For the baseline estimator $\hat{A}_{\text{out}}$, the signal is $A^*(s_t, a_t)$ and the noise is $E_{\text{out}}^2(H)$. The SNR is:

$$\text{SNR}_{\text{base}} = \frac{(A^*(s_t, a_t))^2}{E_{\text{out}}^2(H)}. \tag{13}$$

For ProSearch, the composite advantage $\hat{A}^{\text{ProSearch}}$ integrates both signals:

$$\begin{aligned} \hat{A}^{\text{ProSearch}} &= \hat{A}_{\text{out}} + \beta \hat{A}_{\text{proc}} \\ &= (A^* + e_{\text{out}}) + \beta(A^* + e_{\text{proc}}) \\ &= \underbrace{(1+\beta)A^*}_{\text{Scaled Signal}} + \underbrace{(e_{\text{out}} + \beta e_{\text{proc}})}_{\text{Total Noise}}. \end{aligned} \tag{14}$$

Assuming $\mathbb{E}[e_{\text{out}} e_{\text{proc}}] = 0$, the MSE is at most $E_{\text{out}}^2(H) + \beta^2\sigma_P^2$ and the SNR is:

$$\text{SNR}_{\text{ProSearch}} \geq \frac{(1+\beta)^2 (A^*)^2}{E_{\text{out}}^2(H) + \beta^2 \sigma_P^2}. \tag{15}$$

To establish the condition $\text{SNR}_{\text{ProSearch}} > \text{SNR}_{\text{base}}$, a sufficient condition is:

$$\frac{(1+\beta)^2}{E_{\text{out}}^2(H) + \beta^2\sigma_P^2} > \frac{1}{E_{\text{out}}^2(H)}. \tag{16}$$

By cross-multiplying and rearranging the terms, we derive:

$$(1+2\beta+\beta^2)E_{\text{out}}^2(H) > E_{\text{out}}^2(H) + \beta^2\sigma_P^2 \implies \beta(2+\beta)E_{\text{out}}^2(H) > \beta^2\sigma_P^2. \tag{17}$$

Dividing by $\beta(2+\beta)$ gives:

$$E_{\text{out}}^2(H) > \frac{\beta}{2+\beta}\sigma_P^2. \tag{18}$$

Because $\frac{\beta}{2+\beta} < 1$, Equation 18 presents a theoretical guarantee: **as long as the horizon-induced outcome error E_{out}^2 exceeds a fraction of the PRM's intrinsic local error σ_P^2, ProSearch strictly improves the gradient SNR.** Therefore, the result is particularly relevant to long-horizon tasks when outcome-estimation error is large relative to local process-estimation error. □

Connection to Empirical Observations. Proposition 2 theoretically grounds the empirical findings presented in Section 4:

- **Failure of Outcome-based Baselines:** For methods like standard GRPO and Search-R1 (PPO-based), early reasoning steps suffer from $\text{SNR}_{\text{base}} \approx 0$. Lacking a valid step-level gradient signal, the policy network relies on random parameter drift to connect early actions to delayed rewards. This theoretically explains the stagnant gradient norm (`grad_norm`) and extreme trajectory volatility (`reward_std`) observed in Figure 5, directly manifesting as process hallucinations and redundant search steps.
- **Success of ProSearch:** By anchoring the optimization to a localized PRM, ProSearch explicitly injects the βA^* signal. Equation 18 proves that the PRM acts as a regularizer that cuts through the Markovian noise chain. When outcome-based feedback provides a noisy estimate of intermediate-action quality, sufficiently accurate local process signals can improve advantage-estimation SNR under the condition in Equation 18, providing complementary guidance for policy optimization.

B Additional Methodology Details

B.1 Supervised Policy Warmup

To equip the LLM with the capability for complex multi-hop reasoning and autonomous retrieval, we begin with a Supervised Policy Warmup.

Since standard QA benchmarks typically provide sub-query chains and corresponding retrieval documents without any explicit reasoning process, we leverage a superior LLM (e.g., GPT-4o) to synthesize detailed chain-of-thought trajectories (the prompt templates are provided in Table 3 and Table 4). To facilitate fine-grained supervision, we structure these trajectories into a discrete schema (illustrated in Figure 2) using special control tokens: `<step>` for internal planning, `<subquery>` for search keywords, `<retrieval>` for external evidence, `<subanswer>` for intermediate conclusions, and `<answer>` for the final output.

Leveraging these structured trajectories, we perform supervised fine-tuning to establish the reference policy π_{sft}. To further enforce the structural format, we adopt a format-aware training objective for each action generation:

$$\begin{aligned}\mathcal{L}_{\text{sft}}(\theta) = &- \sum_{o_{t,k} \notin \mathcal{T}_{\text{ctrl}}} \log \pi_\theta(o_{t,k} \mid s_t, o_{t,<k}) \\ &- \lambda \sum_{o_{t,k} \in \mathcal{T}_{\text{ctrl}}} \log \pi_\theta(o_{t,k} \mid s_t, o_{t,<k}),\end{aligned} \tag{19}$$

where s_t is the accumulated context, $o_{t,k}$ denotes the k-th generated token of action a_t, and $\mathcal{T}_{\text{ctrl}}$ represents the set of special control tokens (e.g., `<step>`, `<subquery>`). By setting $\lambda > 1$, we explicitly direct the model's attention to adhere to the predefined reasoning-action schema.

Table 3: System prompt template for the teacher LLM to generate chain-of-thought.

[System Prompt]
You are a meticulous thought process simulator. Your task is to reconstruct the internal monologue of an AI agent solving a problem.
You will be given:

- The original question.
- A set of decomposed reasoning steps (`new_chain_of_thought`) containing the agent's actions (subqueries) and findings (paragraphs and subanswers).
- The final answer.

Your goal is to generate a step-by-step **internal monologue** that simulates how the agent would "think" its way from the question to the final answer.
Output a single JSON object in the following schema:

```
{
  "think_process": [
    {
      "id": "subquery_1",
      "think": "The initial thought process. Based on the user's
question..."
    },
    {
      "id": "subanswer_1",
      "think": "Simulate information extraction. Now that I have the
search result..."
    },
    ...
    {
      "id": "final_answer",
      "think": "Synthesize the final answer. I now have all the
pieces..."
    }
  ]
}
```

Crucial Guidelines:

1. **Adopt a First-Person Perspective:** Write each `think` step from the perspective of the AI agent.
2. **Strictly Sequential Reasoning:** The thought process must *only* use information from steps *before* it.
3. **Focus on "Thinking," Not "Explaining":** Describe reasoning for subqueries, extracting actions for subanswers, and synthesizing for final answers.
4. **Be Concise:** 1–2 fluent sentences per entry.
5. **Output Format:** Output exactly one valid JSON object.

B.2 Oracle LLM Process Evaluator

To curate high-quality preference pairs for Process Reward Modeling, we utilize a superior oracle LLM (e.g., GPT-4o) as an automatic judge. After the MCTS exploration, the judge evaluates sibling actions branching from identical parent contexts. The explicit prompt template used for this step-level evaluation is provided in Table 5

C Experimental Setup details

C.1 LLM Judge Evaluation

As introduced in Section 4.1, we conduct a fine-grained analysis of the reasoning trajectories to better understand the internal planning and reasoning capabilities. Due to the high computational

Table 4: User prompt template for the teacher LLM to generate chain-of-thought.

[User Prompt]
Original question:
{*query*}

Decomposed reasoning steps (new_Chain_of_thought):
{*new_cot*}

Final answer:
{*answer*}

—
Your task is to reconstruct the internal monologue of an AI agent that solved this problem. Generate a step-by-step thinking process based on the information above.
Follow these crucial guidelines:

1. **Adopt a First-Person Perspective.**
2. **Strictly Sequential Reasoning (No Future Peeking).**
3. **Focus on "Thinking," Not "Explaining."**

Output exactly one valid JSON object following the required schema and nothing else.

Table 5: The prompt template for the LLM judge to evaluate the sibling actions.

[System Prompt]
You are an expert AI Tutor creating a training dataset for a reasoning model. Your task is to review the "History" and a list of "Candidate Next Steps" (generated by MCTS), and select exactly ONE pair of (Positive, Negative) actions.

SELECTION RULES:

1. **Positive Action**: The most logical, efficient step that moves towards the answer.
2. **Negative Action**: A step that is clearly worse (e.g., hallucination, logic error, or **redundant** search for info already in history).
3. **Semantic Deduplication (CRITICAL)**:
   - The Negative action MUST be semantically different from the Positive action.
   - Do NOT select a negative sample if it is just a rephrasing of the positive one (e.g., "In what year..." vs "In which year...").
4. **NO FORCED SELECTION**:
   - If all candidates are good, OR all are bad, OR the negatives are just duplicates of the positive:
   - **RETURN NULL** (set `has_valid_pair` to `false`). Do NOT force a selection.

OUTPUT FORMAT (JSON):

```
{
  "has_valid_pair":  true/false,
  "positive_id":  <id of the best candidate>,
  "negative_id":  <id of the bad candidate>,
  "reason":  "<brief explanation>"
}
```

cost and token limits associated with evaluating entire reasoning chains, we randomly sample 1,000 instances each from three representative datasets: PopQA, HotpotQA, and MuSiQue, resulting in a total of 3,000 evaluation instances.

We employ DeepSeek-V3.2 Liu et al. (2025) as the LLM judge. To ensure a rigorous and standardized evaluation, the judge is instructed to assess both the intermediate reasoning steps and the final

Table 6: The prompt template for the LLM judge to evaluate the reasoning trajectories.

You are an objective and expert AI reasoning Evaluator. Your task is to evaluate a model's reasoning trace and classify it into ONE of four categories based on its thoughts, actions, and final answer.

[Context]
Question: {*question*}
Ground Truth Answer: {*ground_truth*}

[Model Trace to Evaluate]
{*trace*}

[Categories and Definitions]

1. **Fact Hallucination:** The model generates fabricated facts, figures, or claims inside its `<step>`, `<subquery>` or `<subanswer>` that are not supported by the context or general knowledge.
2. **Logic Fallacy:** The model's reasoning is flawed. It draws incorrect conclusions from the information, contradicts itself, or fails to logically deduce the correct answer. This also includes cases where the reasoning seems fine but the final answer contradicts the Ground Truth.
3. **Redundant Steps:** The model performs highly repetitive and unnecessary actions (e.g., searching the EXACT SAME query multiple times without making any progress).
   NOTE: Valid multi-hop reasoning (breaking down a question into multiple different search steps to gather partial information) is totally fine and MUST NOT be classified as redundant.
4. **Correct:** The reasoning steps are logically sound, efficient, free of hallucinations, and the final `<answer>` matches or is semantically equivalent to the Ground Truth Answer.

[CRITICAL INSTRUCTION]
The trace contains simulated search results inside `<retrieval>...</retrieval>` tags. You **MUST COMPLETELY IGNORE** any factual errors, contradictions, or weirdness inside the `<retrieval>` blocks. Treat whatever is inside `<retrieval>` as "given context". You are **ONLY** evaluating the model's own generations: what it writes inside `<step>`, `<subquery>`, `<subanswer>` and `<answer>`.

[OUTPUT FORMAT]
You MUST return ONLY a valid XML block in the following format. Do not add markdown or extra text.

```
<evaluation>
  <reasoning>Provide a concise, step-by-step justification for your
evaluation.</reasoning>
  <category>MUST be exactly one of:  [Fact Hallucination, Logic
Fallacy, Redundant Steps, Correct]</category>
</evaluation>
```

answer, strictly classifying the entire trajectory into one of the following four mutually exclusive categories:

- **Correct:** The reasoning process is logically sound, strictly grounded in the provided context, and successfully leads to the correct final answer.
- **Fact Hallucination:** The reasoning trajectory contains fabricated information that is not supported by or contradicts the retrieved documents.
- **Logical Fallacy:** The reasoning chain exhibits logical errors, contradictory deductions, or invalid leap of logic, regardless of whether the final answer is correct.
- **Redundant Steps:** The model successfully arrives at the correct answer, but the reasoning process includes unnecessary, repetitive, or circular steps, indicating suboptimal planning efficiency.

To minimize the variance of the LLM judge and enforce strict adherence to these criteria, we design a detailed instruction prompt. The judge is required to first output a step-by-step analysis before concluding with the final categorical label. The exact prompt used for this evaluation is presented in Table 6

C.2 Detailed Baseline Descriptions

To comprehensively evaluate the effectiveness of ProSearch, we benchmark it against 10 competitive methods categorized into the following three paradigms:

- **Standard Baselines:** These methods represent the fundamental, non-iterative approaches that rely on static pipelines or internal knowledge, serving as the lower bounds for performance.
  - **Naïve Generation:** Relies solely on the pre-trained LLM's parametric knowledge to answer queries without any external retrieval mechanism.
  - **Standard RAG Lewis et al. (2020):** Employs a traditional single-step "retrieve-then-generate" pipeline based exclusively on the initial user query.
- **Advanced Baselines:** These approaches encompass inference-time dynamic frameworks that introduce multi-step, active, or agentic retrieval mechanisms without training.
  - **Iter-RetGen Shao et al. (2023):** Progressively interleaves retrieval and generation by using the model's previous outputs as queries for subsequent retrieval rounds.
  - **IRCoT Trivedi et al. (2023):** Utilizes the intermediate Chain-of-Thought (CoT) reasoning steps as queries to dynamically retrieve and append relevant documents to the context.
  - **FLARE Jiang et al. (2023):** Actively triggers search actions when the generation confidence for upcoming tokens falls below a predefined threshold.
  - **Search-o1 Li et al. (2025):** Acts as an autonomous agent using complex inference-time prompting to explicitly plan and control the multi-step retrieval and reasoning workflow.
- **Reinforcement Learning-based Baselines:** These state-of-the-art methods explicitly optimize the model's reasoning and retrieval policy, serving as our primary competitors.
  - **Search-R1 Jin et al. (2025):** Optimizes the agentic policy via online Proximal Policy Optimization (PPO) using outcome-based terminal rewards based on final answer correctness.
  - **ReasonRAG Zhang et al. (2025):** Performs offline Direct Preference Optimization (DPO) utilizing process-level static preference data synthesized via Monte Carlo Tree Search (MCTS).
  - **HiPRAG Wu et al. (2025):** Employs online reinforcement learning guided by hierarchical process rewards derived from rule-based sub-goal heuristics.
  - **StepSearch Zheng et al. (2025):** Applies step-wise PPO to optimize intermediate search decisions.

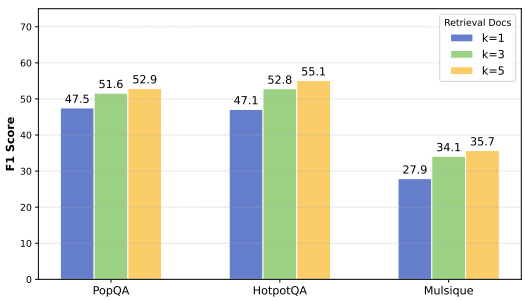
C.3 Implementation Details

We implement our framework based on the HuggingFace Transformers Wolf et al. (2020) and TRL von Werra et al. (2020) Libraries, with Qwen3-8B Yang et al. (2025) serving as the backbone for both our method and all baselines. For the retrieval environment, we use the 2018 Wikipedia dump Karpukhin et al. (2020) as the knowledge source and E5-base Wang et al. (2022) as the retriever, retrieving the top 3 documents for each query.

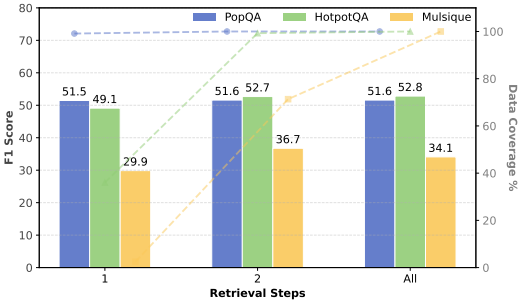
We conduct all experiments on 4 NVIDIA A100 (80GB) GPUs. For the Supervised Policy Warmup, we perform full-parameter fine-tuning on 109k MuSiQue context–action pairs for one epoch to initialize the model's reasoning capability. In the Process Reward Modeling stage, we sample 728 queries from HotpotQA and MuSiQue, and then perform MCTS with 200 simulations, an expansion width of $K = 5$, a maximum depth of 10, an exploration constant $c_{\text{puct}} = 2.5$, and a discount factor $\gamma = 0.99$. For these trees, we collect 8,255 contrastive pairs labeled by GPT-4o, and train the scalar regression head from the SFT checkpoint for three epochs. For the refinement, we select 105k high-quality trajectories from the two datasets filtered by the trained PRM, and fine-tune the policy model for one additional epoch to enhance reasoning ability. Finally, in the Process-Supervised RL stage, we apply LoRA fine-tuning to optimize the policy on 10k held-out queries drawn from both datasets for 1 epoch, with a generation group size of $G = 8$ and a dual-granularity weight $\beta = 0.3$.

D Supplementary Experiments

In this section, we provide additional empirical analyses to further validate the effectiveness, robustness, and efficiency of ProSearch. We first study the impact of retrieval parameters and the



(a) Performance comparison (F1 Score) on PopQA, HotpotQA, and MuSiQue datasets with different numbers of retrieved documents ($k \in \{1, 3, 5\}$).



(b) Cumulative Average F1 Score (bars) and Data Coverage (dashed lines) across different retrieval steps (≤ 1, ≤ 2, All) on PopQA, HotpotQA, and MuSiQue datasets.

sensitivity to the dual-granularity advantage weight β. We then evaluate the reliability and generalization of the learned Process Reward Model (PRM) beyond its training preference pairs. Finally, we analyze efficiency from both inference and training perspectives, including search-tool usage during deployment and the supervision, environment interaction, and external API costs required during optimization.

D.1 Impact of Retrieval

In this subsection, we investigate the performance of ProSearch under different numbers of retrieved documents and retrieval steps to evaluate its context utilization and planning adaptivity.

D.1.1 Impact of the Number of Retrieved Documents

As shown in Figure 6a, the performance gain from increasing the retrieval context (k) correlates strongly with task complexity. On the single-hop PopQA dataset, the performance curve saturates early as k increases, indicating that a single relevant document is typically sufficient for factual recall. Notably, the model's performance does not decrease at $k = 5$, suggesting strong robustness against noise from irrelevant documents. In contrast, the complex multi-hop benchmark MuSiQue demonstrates significant sensitivity to context size, with the F1 score jumping by 6.2 points when k increases to 3. This finding suggests that a restricted context window ($k = 1$) creates an information bottleneck for reasoning. By providing multiple documents, ProSearch can effectively identify the key evidence, which is crucial for resolving long-horizon dependencies.

D.1.2 Impact of the Number of Retrieval Steps

Figure 6b illustrates the cumulative average F1 score and data coverage across retrieval steps, highlighting the dynamic planning capability of ProSearch. The data coverage curves reveal distinct adaptivity patterns, where nearly 100% of PopQA queries are efficiently resolved within the first step, thereby minimizing unnecessary computational cost for simple tasks. Conversely, the model autonomously extends the reasoning chain for complex benchmarks such as HotpotQA and MuSiQue, which often require more than single step. Furthermore, while longer reasoning chains in typical iterative frameworks often suffer from performance degradation due to error propagation, ProSearch maintains robust accuracy with cumulative F1 scores increasing as retrieval steps grow. For instance, the F1 score on MuSiQue increases from 29.9 to 34.1, demonstrating that process-level supervision helps guide search in complex spaces and stabilize long-horizon reasoning.

D.2 Sensitivity Analysis

We investigate the sensitivity of model performance to the dual-granularity weight β. As shown in Table 7, performance exhibits an inverted-U trend, peaking at $\beta = 0.3$. At $\beta = 0.0$, the model relies solely on sparse outcome signals, struggling to identify the specific reasoning steps that lead to the correct answer, which results in the lowest performance. In contrast, introducing process supervision ($\beta > 0$) significantly boosts performance, with $\beta = 0.3$ achieving the optimal results

Table 7: Sensitivity analysis of β on the average performance across all benchmarks. A balanced weight of $\beta = 0.3$ yields the best results for both EM and F1. All runs in this sensitivity analysis start from the same PRM-refined policy.

Metric	Dual-Granularity Weight β					
	0.0	0.1	**0.3**	0.5	0.7	0.9
EM	38.3	40.3	**40.7**	39.2	38.5	38.4
F1 score	46.7	48.3	**49.2**	47.7	47.1	46.8

Table 8: Inference-time retrieval efficiency on a matched subset of 599 questions: 199 from HotpotQA, 200 from MuSiQue, and 200 from PopQA. Both retrieval calls and F1 are macro-averaged across the three datasets. Lower retrieval calls and higher F1 are better.

Method	Avg. Retrieval Calls	Avg. F1
Search-R1	2.53	43.74
ReasonRAG	1.64	32.63
StepSearch	2.14	38.37
ProSearch	**1.64**	**48.01**

(49.2% F1). This indicates that the dense feedback from the PRM serves as a critical guide to stabilize the reasoning trajectory and resolve the credit assignment problem. However, as β increases further ($\beta \geq 0.3$), we observe a consistent decline. This suggests that although the PRM provides dense signals, it inevitably contains estimation errors. Consequently, an overweighted PRM causes the model to overfit to noisy step-level signals rather than the ground-truth objective, degrading the final reasoning accuracy.

D.3 PRM Evaluation

Since ProSearch relies on the learned Process Reward Model (PRM) to provide step-level supervision during both policy refinement and online RL, we further evaluate whether the PRM can reliably distinguish valid from flawed intermediate actions.

We first evaluate the PRM on a held-out set of 166 contrastive preference pairs that are excluded from PRM training. The PRM correctly ranks the preferred action above its rejected counterpart in **84.9%** of the pairs, demonstrating that the localized preference-ranking formulation learned from MCTS-derived sibling comparisons generalizes well beyond the training examples.

We further evaluate the PRM on 200 complete on-policy trajectories sampled from HotpotQA, MuSiQue, and PopQA. A separate GPT-5.5-medium judge independently annotates local step quality. For intermediate actions, the frozen PRM distinguishes high-quality from low-quality steps with an AUC of **0.779**. Furthermore, without relying on LLM-judge annotations, we examine whether PRM scores correlate with final task success. Across 567 on-policy trajectories, the mean PRM score over intermediate actions achieves an AUC of **0.717** for distinguishing gold-EM-correct from incorrect trajectories.

Together, these evaluations support meaningful preference-ranking and intermediate-action quality signals, with additional evidence of association with downstream task performance. These findings support the use of the frozen PRM as a process-level supervision signal in ProSearch.

D.4 Efficiency Analysis

D.4.1 Inference-time Retrieval Efficiency.

We first examine whether ProSearch learns more efficient search behaviors at inference time. Specifically, we measure the average number of retrieval interactions per example on the same 599 evaluation instances from HotpotQA, MuSiQue, and PopQA, and report the corresponding F1 score to analyze the accuracy-efficiency trade-off.

Table 9: The data requirements for Search-R1 and ReasonRAG are sourced directly from their original papers. The data requirements for HiPRAG* are projected by applying its step-wise external evaluation mechanism to our GRPO training configuration (10,000 queries × 8 rollouts × estimated 3 steps per rollout).

Method	Process Signal Source	API Calls	Offline	Online
Search-R1	N/A	0	0	∼90,000
ReasonRAG	Offline Synthesized Pairs	0	∼5,000	0
HiPRAG*	External LLM APIs	∼240,000	0	10,000
ProSearch	**Local Distilled PRM**	**∼ 10,000**	**728**	**10,000**

As shown in Table 8, ProSearch requires only **1.64 retrieval calls** per example on average, compared with **2.53** for Search-R1 and **2.14** for StepSearch, corresponding to reductions of **35.2%** and **23.4%**, respectively. Importantly, this reduction is not achieved by sacrificing answer quality: ProSearch attains the highest matched average F1 of **48.01**, compared with 43.74 for Search-R1 and 38.37 for StepSearch.

ReasonRAG exhibits a comparable average number of retrieval calls (**1.64**), but achieves substantially lower answer quality (**32.63** F1). At an approximately matched retrieval budget, ProSearch therefore improves F1 by **15.38 points**, suggesting that its efficiency comes from more effective retrieval decisions rather than simply terminating search earlier.

Overall, these results indicate that localized process supervision helps the policy avoid unnecessary retrieval interactions while preserving and improving final task performance. This observation is consistent with the trajectory-level analysis in Section 4.5, where ProSearch produces fewer redundant reasoning and search steps.

D.4.2 Training-time Supervision Efficiency.

In this section, we analyze the training data efficiency of ProSearch. To ensure a fair comparison, we categorize the training data requirements into offline queries, online rollouts, and external API calls. As summarized in Table 9, ProSearch achieves an optimal balance, outperforming recent baselines in holistic efficiency.

Standard outcome-based RL methods like Search-R1 Jin et al. (2025) rely solely on the terminal reward but suffer from blind exploration, requiring massive environment interactions (e.g., ∼**90k** training queries) to converge. Conversely, online process-supervised methods like HiPRAG Wu et al. (2025) invoke external proprietary APIs for reward reshaping, improving the sample efficiency in policy training. However, this introduces continuous on-the-fly API calls, leading to prohibitive monetary and latency overheads. Meanwhile, offline process-optimization approaches like ReasonRAG Zhang et al. (2025) rely on executing exhaustive MCTS (e.g., ∼**5k** queries) to synthesize static preference pairs. Since each MCTS query involves hundreds of simulated node expansions and oracle evaluations, this method incurs substantial computational cost.

In contrast, ProSearch effectively mitigates these limitations through a decoupled architecture. By investing a minimal budget of only **728 queries** into MCTS, we distill a generalized local PRM that provides continuous, dense supervision without relying on costly external APIs. This small upfront investment drastically reduces the sample complexity of the subsequent RL phase to merely **10k queries**, allowing ProSearch to achieve superior performance while minimizing the total computational and annotation overhead.