BranPO: Scalable Contrastive Branch Sampling for Long-Horizon Agentic Reinforcement Learning

Anonymous ACL submission

Abstract

Agentic reinforcement learning enables large language models to perform multi-turn planning and tool use, but long-horizon training remains challenging under sparse trajectory-level rewards, where a single outcome is uniformly assigned to all decisions. Prior methods introduce finer-grained supervision via tree-based exploration or process-level evaluation, but often incur high cost or produce noisy credit signals. In agentic trajectories, early mistakes may still be corrected by later actions, while seemingly promising intermediate states can fail due to poor subsequent decisions. We call this property non-monotonic correctness, which makes outcome rewards or state values insufficient for guiding what actions should be taken from each state. To address this, we propose Branching Relative Policy Optimization (**BranPO**), a value-free method that constructs localized contrastive supervision without dense rewards. BranPO truncates trajectories at intermediate prefixes and resamples continuations to form contrastive branches that share the same prefix but diverge in final outcomes, thereby isolating decisions that drive success or failure. We further introduce difficulty-aware branch sampling and Redundant Step Masking to improve sampling efficiency and suppress redundant updates. Experiments show that BranPO consistently outperforms diverse baseline categories across multiple multi-hop QA benchmarks without additional training cost, and generalizes to broader long-horizon agentic tasks with consistent improvements.

1 Introduction

Large language models (LLMs) have increasingly evolved from passive text generators into autonomous agents capable of multi-turn interaction, planning, and tool use. Agentic reinforcement learning (RL) further improves these capabilities through policy optimization (Liu et al., 2025a; MiniMax, 2025; Moonshot, 2025), enabling progress on long-horizon tasks such as deep research (Team et al., 2025b), code generation (Feng et al., 2025b), and GUI-based assistance (Qin et al., 2025)

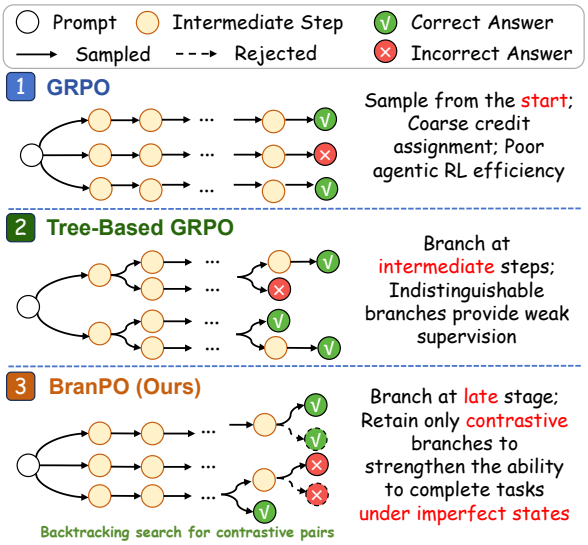

Figure 1: Comparison of sampling process between GRPO, tree-based GRPO, and BranPO.

Despite recent progress, training multi-turn agents remains difficult under sparse rewards (Xue et al., 2025; Qian et al., 2025; Song et al., 2025), where a single trajectory-level outcome supervises all actions (Liu et al., 2025c). Existing methods seek finer-grained credit assignment through process reward models (Zhang et al., 2025a,b; Zeng et al., 2025), tree-based (Yang et al., 2025b; Ji et al., 2026), or entropy-guided branch sampling (Shen et al., 2025; Dong et al., 2026a,b). However, these approaches remain limited in long-horizon interaction. Process rewards often rely on LLM-as-a-Judge or task-specific evaluators, which can be unreliable under incomplete or delayed evidence. Tree-based and entropy-guided methods require additional branch rollouts; yet as horizons grow, outcome-critical steps become sparse, so many sampled branches have continuations that either all succeed or all fail, incurring substantial compu-

tation for weak learning signals.

We argue that this limitation reflects a structural property of multi-turn agentic tasks, which we call **non-monotonic correctness**. In such tasks, correctness does not accumulate monotonically along a trajectory. A flawed intermediate state can be recovered through later corrective actions, whereas a promising prefix may still fail due to poor subsequent decisions. This makes isolated judgments of intermediate states unreliable, because the value of a prefix depends on how the agent continues from it. Earlier uncertainty or errors may still be resolved through later query reformulation, evidence integration, or final answer construction, making late-stage actions especially consequential in determining whether a prefix is corrected into success or carried into failure. Therefore, effective supervision should prioritize continuations from intermediate states, including imperfect ones, while reinforcing high-quality earlier decisions, so that the agent learns to build useful prefixes and correct rather than propagate earlier uncertainty.

Motivated by this, we propose **Bran**ching Relative **P**olicy **O**ptimization (**BranPO**), a contrastive dynamic branch sampling method for multi-turn agent training. As shown in Figure 1, BranPO turns completed trajectories into targeted training signals by searching backward for late-stage branching points, avoiding expensive tree expansion over the full interaction horizon. From each selected prefix, BranPO resamples alternative continuations and retains only contrastive branches whose continuations lead to different final outcomes, concentrating learning on decisions that truly affect task success. BranPO further applies hierarchical group-relative optimization: prefix-level advantages distinguish more and less promising branching contexts, while branch-level advantages contrast successful and failed continuations under the same shared prefix. This yields a compact, high-signal objective for long-horizon, non-monotonic agentic tasks.

To improve efficiency, we introduce difficulty-aware branch sampling, which adaptively allocates sampling budgets based on accuracy. To mitigate the continuation bias, we propose Redundant Step Masking (RSM), which identifies and masks redundant tail steps during policy updates, thereby reinforcing informative supervision.

Across diverse multi-hop QA benchmarks, BranPO achieves consistent improvements over GRPO and strong agentic RL variants, including explicit process-reward methods, tree-based branch sampling, and entropy-guided branch sampling, without increasing the training budget. Additionally, BranPO acquires long-horizon search behaviors substantially faster, and its benefits generalize to 30+ turn agentic tasks and tasks beyond search.

In summary, our contributions are as follows:

- We introduce BranPO, the first agentic RL algorithm that adapts policy optimization to the non-monotonic credit structure of long-horizon multi-turn agentic tasks via contrastive, outcome-divergent branch learning.
- We introduce difficulty-aware branch sampling and Redundant Step Masking to improve exploration efficiency and suppress redundant behaviors.
- Extensive experiments demonstrate that BranPO consistently outperforms prior methods in long-horizon agentic tasks.

2 Related Work

2.1 Search Agents

Search agents have emerged as a pivotal approach for open-domain problem solving by integrating reasoning with external tool use. Representative studies such as Search-o1 (Li et al., 2025a), WebThinker (Li et al., 2025b), and Search-R1 (Jin et al., 2025) demonstrate how iterative reason-retrieve-synthesize loops can be optimized through reinforcement learning. Despite recent progress in query generation (Jiang et al., 2025a), self-reflection (Fu et al., 2025), and data efficiency (Jiang et al., 2025b), search agents continue to face significant challenges in long-horizon tasks (Gao et al., 2025). In particular, the lack of fine-grained process supervision limits existing feedback mechanisms from effectively guiding intermediate steps within long interaction sequences (Liu et al., 2026).

2.2 Credit Assignment in RL

Credit assignment remains a fundamental bottleneck in training LLMs for complex, long-horizon tasks. Standard RL approaches (Ouyang et al., 2022) typically rely on sparse, outcome rewards, resulting in high variance and ambiguity. Existing methods for finer-grained supervision fall into two categories: (1) *Monte Carlo estimation*, which branches from intermediate states and reruns rollouts to estimate expected rewards. For

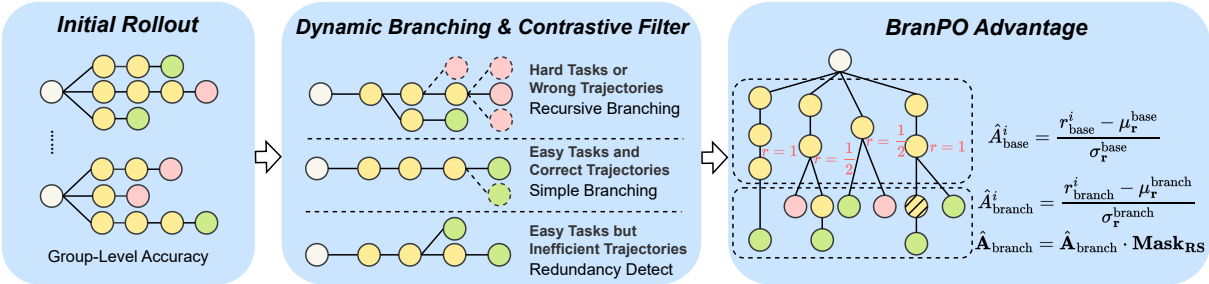

Figure 2: Overview of BranPO. After the initial rollout, group accuracy is computed and branching budgets are assigned based on task accuracy and trajectory reward. Simple branching is applied to correct trajectories in easy tasks, while recursive branching is used for hard tasks or incorrect trajectories. Redundancy checks are performed on inefficient trajectories within easy tasks. Within a task, shared prefixes from all trajectories form a base group, while branched continuations from all trajectories constitute a branch group for relative advantage computation, with advantages corresponding to inefficient steps masked.

example, VinePPO (Kazemnejad et al., 2024) operates at the token level, Tree-GRPO (Ji et al., 2026) and ReasonRAG (Zhang et al., 2025a) at the turn level, while other work exploits uncertainty signals such as entropy (Shen et al., 2025; Dong et al., 2026a,b) to guide efficient branching. (2) *Explicit step-level reward methods*, which directly construct step-level supervision, such as GiGPO (Feng et al., 2025a), StepSearch (Wang et al., 2025), MT-GRPO (Zeng et al., 2025), and CriticSearch (Zhang et al., 2025b). However, explicit step-level annotation scales poorly. Fine-grained credit assignment based on outcome-level rewards remains a more scalable and promising direction for long-horizon agentic learning.

3 Methodology

3.1 Non-Monotonic Correctness

We formulate multi-turn agentic reasoning as a finite-horizon Markov Decision Process where the model interacts with an environment over multiple steps. At each turn t, the agent observes the full interaction history $h_t = (a_1, o_1, \ldots, a_{t-1}, o_{t-1})$, where o_t denotes the environment observation (e.g., search results) following action a_t, and produces the next action a_t, a variable-length token sequence constituting a complete LLM response. The policy $\pi_\theta(a_t \mid h_t)$ is thus history-conditioned. An episode forms a trajectory $\tau = (a_1, o_1, \ldots, a_T)$ and receives only a final outcome reward $R(\tau)$.

Most existing RL methods for agentic tasks assign this single outcome reward uniformly to all actions in the trajectory, implicitly treating each action as contributing equally, and in the same direction, toward the final result (Shao et al., 2024; Jin et al., 2025; Ji et al., 2026). More sophisticated approaches attempt finer-grained credit assignment by estimating intermediate state values through Monte Carlo sampling: for each intermediate history, multiple rollouts are generated and the resulting win rate is used as a value estimate (Zhang et al., 2025a). Despite their granularity, both families of methods share a common *monotonicity assumption*: that a high-value intermediate state reliably leads to success, while a low-value state is unlikely to be recovered, as if correctness accumulates predictably along the trajectory.

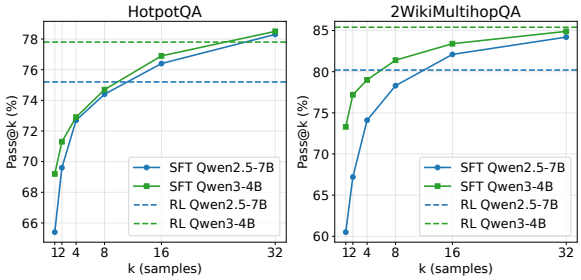

Figure 3: Pass@K results of SFT-trained Qwen2.5 and Qwen3 with partial resampling at the trajectory tail.

However, multi-turn search trajectories violate this assumption. Following the ReAct paradigm (Yao et al., 2022), a search agent alternates between issuing queries and synthesizing retrieved evidence. Unlike single-turn reasoning, multi-turn search continuously exposes the model to new observations, which may reshape its subsequent reasoning. As a result, trajectory correctness can be *non-monotonic*: early mistakes may later be corrected through query reformulation, evidence reinterpretation, or continued exploration, while seemingly promising trajectories may still fail due to premature answering, hallucination, or faulty evidence aggregation.

To quantify this phenomenon, we conduct a

partial resampling study on multi-hop QA benchmarks. Specifically, we evaluate 1K-example subsets of HotpotQA (Yang et al., 2018) and 2WikiMultihopQA (Ho et al., 2020), using Qwen2.5-7B-Instruct (Yang et al., 2024) and Qwen3-4B-Instruct (Yang et al., 2025a) models fine-tuned on search trajectories. For each example, we first sample a complete rollout, truncate it before the final answer generation step, and then resample 32 continuations from the same intermediate state. The model is allowed to either answer immediately or continue searching. Performance is evaluated using Pass@K with an LLM-as-a-Judge.

As shown in Figure 3, resampling from the same late-stage state improves Pass@32 by 10%–25% over Pass@1, indicating that different continuations from identical contexts can lead to divergent outcomes and that SFT-trained agents already possess a spontaneous ability to recover from imperfect states (see Appendix F.1 for examples).

This observation exposes a key limitation of existing credit assignment strategies. In multi-turn search, the same history can lead to different outcomes depending on what the agent does next, making the history itself an insufficient unit of credit. Uniform outcome assignment is too coarse: in a failed trajectory, it penalizes all actions alike, including those that may have moved the agent toward the correct answer. Process rewards are more fine-grained, but they still assign scores to intermediate states rather than explicitly specifying which actions should be taken from them. Thus, neither approach explicitly supervises which behaviors under the current history lead to a correct final answer.

3.2 Difficulty-Aware Branch Sampling

Building on this observation, we shift the supervision target from evaluating how good a state is to comparing what continuations should be taken from that state. To obtain such localized signals, we introduce *branch sampling*.

Given an initial trajectory, we truncate it at a late-stage state and resample a suffix continuation from the truncated point. This converts credit assignment from evaluating a single complete trajectory into comparing alternative decisions under the same prefix. To ensure informative training signal, we retain only *contrastive branches*: if the original rollout is incorrect, we search for a correct continuation, and vice versa. Branches whose outcomes match the original are discarded, so that each retained pair provides a meaningful positive–negative contrast.

A fixed branching strategy is inefficient across tasks of varying difficulty. BranPO therefore adopts a *difficulty-aware branch sampling* strategy. After the initial rollout, group accuracy is computed and branching budgets are assigned based on task accuracy and trajectory reward. For correct trajectories on easy tasks, *simple branching* is applied: a single truncation-and-resample attempt is made. For hard tasks or incorrect trajectories, *recursive branching* is used: starting from the last reasoning step, the method progressively backtracks the truncation point and resamples at each step, until either a contrastive branch is found or the branching budget is exhausted. As illustrated in Figure 2, this adaptive allocation concentrates computation on challenging instances where alternative continuations are most informative, while avoiding unnecessary exploration on trivial cases. The full algorithm is summarized in Appendix A.1.

3.3 Branching Advantage Estimation

We now formalize the proposed method, termed Branching Relative Policy Optimization (BranPO). BranPO builds on the agentic MDP defined in Section 3.1 and instantiates the branching strategy introduced above through a principled advantage estimation scheme.

Following GRPO, BranPO treats each task prompt q as a group and samples N trajectories $\{\tau^{(q,j)}\}_{j=1}^{N}$ from the current policy. Branching is performed independently for each trajectory. For the n-th trajectory, we truncate it at a selected turn k to obtain a shared prefix

$$\mathcal{B}_{q,n} = \tau_{\leq k}^{(q,n)} = (a_1, o_1 \ldots, a_k, o_k) \quad (1)$$

which defines a state s_{k+1} in the agentic MDP. From this state, we sample a set of branch continuations

$$\mathcal{C}_{q,n} = \{\tau_{>k}^{(q,n,i)}\}_{i=1}^{|\mathcal{C}_{q,n}|} \quad (2)$$

Each branch yields a complete trajectory with outcome reward $r_{\text{branch}}^{(q,n,i)} = R(\mathcal{B}_{q,n}, \mathcal{C}_{q,n}^{(i)})$.

When both successful and failed branches originate from the same prefix, credit assignment for the shared prefix becomes ambiguous. To mitigate this ambiguity and reduce gradient variance, we define the base trajectory reward for the prefix as the mean return of its branches:

$$r_{\text{base}}^{(q,n)} = \frac{1}{|\mathcal{C}_{q,n}|} \sum_{i=1}^{|\mathcal{C}_{q,n}|} r_{\text{branch}}^{(q,n,i)} \quad (3)$$

BranPO normalizes advantages following GRPO. The base advantage assigned to the shared prefix $\mathcal{B}_{q,n}$ is computed across trajectories sampled from the same prompt:

$$\hat{A}_{\text{base}}^{(q,n)} = \frac{r_{\text{base}}^{(q,n)} - \mu_r^{\text{base}}(q)}{\sigma_r^{\text{base}}(q)} \tag{4}$$

where $\mu_r^{\text{base}}(q)$ and $\sigma_r^{\text{base}}(q)$ is the mean and standard deviation of the group $\{r_{\text{base}}^{(q,j)} \mid 1 \leq j \leq N\}$.

For each branch continuation $\mathcal{C}_{q,n}^{(i)}$, the advantage is computed across all branches from the same prompt:

$$\hat{A}_{\text{branch}}^{(q,n,i)} = \frac{r_{\text{branch}}^{(q,n,i)} - \mu_r^{\text{branch}}(q)}{\sigma_r^{\text{branch}}(q)} \tag{5}$$

with $\mu_r^{\text{branch}}(q)$ and $\sigma_r^{\text{branch}}(q)$ is the mean and standard deviation of the group $\left\{r_{\text{branch}}^{(q,j,l)} \mid 1 \leq j \leq N,\, 1 \leq l \leq |\mathcal{C}_{q,j}|\right\}$.

The resulting policy gradient decomposes credit assignment across the trajectory. Specifically, the base advantage $\hat{A}_{\text{base}}^{(q,n)}$ is applied to all actions in the shared prefix $\mathcal{B}_{q,n}$ through the log probability $\log p_\theta(\mathcal{B}_{q,n})$. For each branch continuation, the corresponding branch-level advantage $\hat{A}_{\text{branch}}^{(q,n,i)}$ is applied to the actions in the suffix via the conditional log probability $\log p_\theta(\mathcal{C}_{q,n}^{(i)} \mid \mathcal{B}_{q,n})$.

Why BranPO is more effective? The structure of BranPO naturally unifies trajectory-level RL with contrastive preference optimization within a single framework. As derived in Appendix D, BranPO assigns credit differently to prefixes and suffixes.

For the shared prefix $\mathcal{B}$ (the search history), it functions as a variance-reduced variant of GRPO. By averaging returns across branched continuations, it stabilizes the learning signal for early steps that face inherently ambiguous credit assignment (Appendix D.3).

For the branched suffixes, BranPO induces a contrastive gradient structure analogous to preference optimization. By contrasting a correct continuation $\mathcal{C}^+$ against an incorrect one $\mathcal{C}^-$ from the same prefix, the suffix gradient takes the form:

$$\nabla_\theta \mathcal{L}_{\text{suffix}} \propto \underbrace{\nabla_\theta \log \pi_\theta(\mathcal{C}^+ \mid \mathcal{B}) - \nabla_\theta \log \pi_\theta(\mathcal{C}^- \mid \mathcal{B})}_{\text{Contrastive preference optimization}} \tag{6}$$

This gradient explicitly increases the likelihood gap between successful and failed continuations conditioned on the same history, sharing the core mechanism of DPO methods (Rafailov et al., 2023) that optimize pairwise preferences. The key distinction is that BranPO derives this contrastive signal online from the agent's own branched rollouts rather than from a fixed preference dataset, and applies it selectively to the suffix where divergence actually occurs. This hybrid structure allows BranPO to maintain robust optimization for early-stage retrieval actions through group-wise trajectory comparison, while leveraging the sample efficiency of contrastive supervision for late-stage synthesis decisions where credit can be precisely localized.

3.4 Redundant Step Masking

Redundant tool usage is a common failure mode in multi-turn search agents (Team et al., 2025a). Agents may continue to invoke the search tool even after sufficient evidence has been obtained, resulting in redundant actions with little additional benefit. An example is shown in Appendix F.2. Under the branch sampling strategy introduced in Section 3.2, this issue can be further amplified. When truncation occurs near the trajectory tail, continuations that keep searching may be preferentially sampled if intermediate evidence appears incomplete, biasing training toward excessive verification.

While additional search steps can be reasonable for difficult instances, uncontrolled redundancy leads to inefficient training and suboptimal policies. To address this issue, we propose *Redundant Step Masking (RSM)*, which augments BranPO with a dedicated redundancy-detection mechanism. For correct trajectories whose interaction length exceeds the average length of correct trajectories within the same group, instead of applying the default BranPO branching strategy, we explicitly branch from the penultimate model response. If an alternative continuation reaches a correct answer with fewer steps, the original extra step is identified as redundant.

During optimization, the advantages associated with these redundant actions are masked and excluded from gradient computation, while the shorter correct continuation is retained as a valid training branch. Specifically, for branches identified as containing redundant steps, we apply a mask to the branch advantage:

$$\hat{A}_{\text{branch}} = \hat{A}_{\text{branch}} \cdot \text{Mask}_{\text{RS}} \tag{7}$$

where $\text{Mask}_{\text{RS}} \in \{0, 1\}$ zeros out the advantages corresponding to redundant actions. By decoupling correctness from efficiency at the step level, RSM

Method	2WikiMQA		HotpotQA		MuSiQue		Bamboogle		Average	
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
Outcome Reward										
Search-R1† (Jin et al., 2025)	31.2	37.6	39.4	50.2	18.1	26.2	38.4	50.1	31.8	41.0
SE-Search† (Li et al., 2026)	35.5	42.2	43.8	55.9	19.5	29.0	48.8	60.1	36.9	46.8
Process Reward										
ReasonRAG† (Zhang et al., 2025a)	43.6	50.4	38.4	48.9	12.8	20.6	36.0	45.5	32.7	41.3
StepSearch† (Wang et al., 2025)	36.6	43.1	38.6	50.2	**22.6**	31.2	40.0	53.4	34.5	44.5
GiGPO (Feng et al., 2025a)	42.3	48.0	42.2	53.4	19.2	27.1	44.0	50.1	36.9	44.7
CriticSearch† (Zhang et al., 2025b)	42.8	50.1	44.2	56.0	19.4	28.1	47.2	59.2	38.4	48.4
TIPS† (Xie et al., 2026)	43.0	50.6	43.0	54.7	17.1	26.6	36.8	52.2	35.0	46.0
Branch Sampling										
Tree-GRPO (Ji et al., 2026)	41.9	48.6	44.0	56.2	20.9	30.3	44.0	57.1	37.7	48.1
ARPO (Dong et al., 2026b)	42.4	51.3	46.1	59.1	18.2	27.3	40.0	50.9	36.7	47.2
AEPO (Dong et al., 2026a)	46.7	53.5	45.7	58.2	19.3	28.4	44.0	54.6	38.9	48.7
BranPO (Ours)	**48.1**	**56.7**	**48.9**	**61.8**	21.2	**32.4**	**50.4**	**62.0**	**42.2**	**53.2**

Table 1: Performance comparison on multi-hop QA benchmarks. All methods except StepSearch are trained on Search-R1 training set, with HotpotQA as the in-domain test set. StepSearch is trained on the MuSiQue training set. † means the results are obtained from the original paper. The best results are indicated with **bold**.

prevents reinforcement of unnecessary tool usage while preserving informative contrastive learning signals, which is beneficial for stable optimization in long-horizon agentic settings.

4 Experiments

Datasets. We train BranPO on two widely used search datasets: Search-R1 (Jin et al., 2025) and ASearcher (Gao et al., 2025) . Search-R1 training set mainly consists of 1–2 hop search questions from NQ (Kwiatkowski et al., 2019) and HotpotQA (Yang et al., 2018). ASearcher contains more challenging multi-hop search tasks, allowing us to assess the performance of our algorithm in longer-horizon scenarios. We filter out invalid questions from the original ASearcher dataset. For each protocol, we keep the corresponding search engine and corpus consistent with the original work.

Training. For experiments following Search-R1, we keep the training configuration as close as possible to the original Search-R1 setup. For the ASearcher benchmark, we adopt a two-stage training strategy following AgentGym-RL (Xi et al., 2025): we first limit the interaction horizon to 4 steps and then extend it to 8 steps. Given our computational budget, this curriculum is sufficient to solve the tasks efficiently while stabilizing long-horizon RL training. We use a fully on-policy training setup rather than the asynchronous GRPO used in ASearcher, which allows us to isolate the effect of different sampling and credit-assignment strategies.

For training stability, we warm-start all RL-based methods from the same SFT-initialized model; details of the SFT procedure are provided in the Appendix A.2. Full RL training details are provided in Appendix A.3, and algorithm settings are described in Appendix A.1. As shown in Appendix A.5, BranPO achieves comparable wall-clock training time to other methods, ensuring fair comparison and comparable training cost.

Evaluation. We evaluate on four multi-hop QA benchmarks: HotpotQA (Yang et al., 2018), 2WikiMultihopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023). Following the corresponding protocols, we report F1 score and an LLM-as-a-Judge (LasJ) metric for ASearcher, and Exact Match (EM) and F1 for Search-R1. We additionally evaluate web search ability on text-only questions from GAIA (Mialon et al., 2023) in Appendix B. Detailed evaluation protocols are provided in Appendix A.4.

4.1 Main Results

We first evaluate BranPO under the Search-R1 setting with Qwen2.5-7B-Instruct (Yang et al., 2024) as the backbone, comparing against outcome-reward RL methods, explicit process-reward RL methods, and prior branch-sampling methods. As shown in Table 1, BranPO achieves the best aver-

Method	2WikiMQA		HotpotQA		MuSiQue		Bamboogle		Average	
	F1	LasJ	F1	LasJ	F1	LasJ	F1	LasJ	F1	LasJ
Qwen2.5-7B-Instruct										
GRPO	64.8	73.6	60.4	71.9	34.6	42.0	51.3	60.8	52.8	62.1
Tree-GRPO	67.6	75.1	58.3	69.4	34.6	42.0	49.6	57.6	52.5	61.0
ARPO	63.3	74.3	58.6	68.9	31.7	40.0	51.8	59.2	51.4	60.6
AEPO	65.1	76.0	60.4	71.7	32.5	40.4	**53.4**	**61.6**	52.9	62.4
BranPO	**72.0**	**80.2**	**62.7**	**75.2**	**35.1**	**43.8**	52.7	**61.6**	**55.6**	**65.2**
Qwen3-4B-Instruct-2507										
GRPO	71.9	82.6	61.4	74.5	37.3	44.9	58.3	70.4	57.2	68.1
Tree-GRPO	68.7	78.2	58.4	71.2	32.0	41.0	54.6	65.6	53.4	64.0
ARPO	67.0	77.0	57.6	73.4	31.9	39.8	51.8	64.0	52.1	63.6
AEPO	69.5	78.9	60.6	72.0	32.5	41.2	56.8	68.0	54.9	65.0
BranPO	**75.6**	**85.4**	**64.5**	**77.8**	**38.9**	**48.7**	**60.3**	**72.0**	**59.8**	**71.0**

Table 2: Performance on multi-hop QA tasks when training on ASearcher dataset, with F1 and LLM-as-a-Judge (LasJ) score on each dataset and average. The best results are indicated with **bold**.

age performance, reaching 42.2% EM and 53.2% F1, outperforming the strongest prior method by 3.3% EM and 4.5% F1 on average. These results suggest that branch-level preference optimization provides more effective credit assignment than previous methods, while avoiding the need for manually designed per-turn reward.

Results under the ASearcher setting. We further evaluate BranPO on the more challenging ASearcher dataset using Qwen2.5-7B-Instruct (Yang et al., 2024) and Qwen3-4B-Instruct (Yang et al., 2025a). In addition to GRPO, we focus on branch-sampling baselines, because these methods show strong performance in the shorter-horizon Search-R1 setting. During evaluation, we increase the maximum number of allowed search actions to 16 to assess longer-horizon behavior. The results are summarized in Table 2.

Across datasets and backbones, BranPO achieves the best F1 and LasJ scores, outperforming both GRPO and branch-sampling-based variants. While prior branch sampling methods do not consistently surpass GRPO on ASearcher, BranPO does, indicating that its gains come from branch-level preference optimization rather than branch sampling alone. This suggests that BranPO yields more stable credit assignment across extended decision sequences, enabling the model to sustain effective exploration under imperfect intermediate states in complex multi-hop search tasks.

Training dynamics. The two-stage training dynamics of Qwen2.5-7B-Instruct under ASearcher setting are illustrated in the Figure 4. It tracks the progression of training rewards, the evolution of the average number of search steps in training rollouts, the changing proportions of trajectories containing branches with different reward outcomes, and the fraction of trajectories that BranPO identifies as exhibiting redundant steps during training. As shown by the average number of search steps, BranPO learns faster than GRPO to increase search usage to ensure higher answer accuracy. Meanwhile, the proportion of trajectories with contrastive reward branches steadily decreases, indicating that late-stage errors are progressively reduced and that BranPO effectively optimizes tail decisions. As the interaction budget is extended, the proportion of trajectories containing redundant steps first increases and then declines, suggesting that RSM suppresses unnecessary exploratory behaviors. Nevertheless, trajectories with contrastive reward branches remain accounting for more than 20% of all trajectories, indicating that substantial learning signals still remain for further optimization.

4.2 Generalization to Extremely Long-Horizon Tasks

When the interaction horizon becomes extremely long (e.g., 32+ steps), directly applying BranPO introduces considerable computational overhead due to recursive truncation and backtracking from the final step. To improve scalability, we introduce a segment-level variant of BranPO.

Given a trajectory with M interaction steps, we evenly divide it into S segments, each containing M/S steps. BranPO then performs backtracking and branching at the segment level instead of ev-

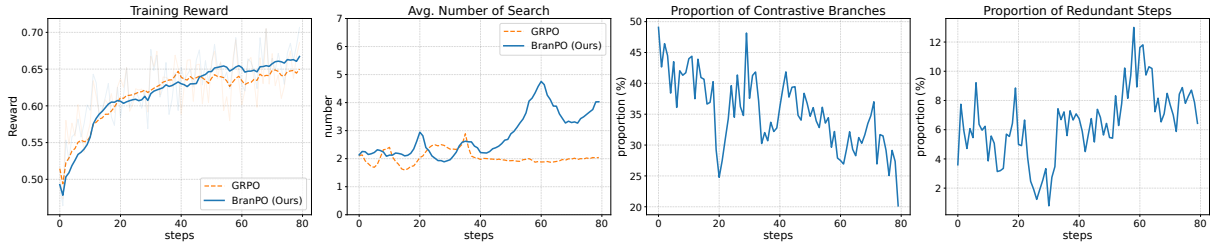

Figure 4: Training dynamics on Qwen2.5-7B-Instruct, including the training rewards, the change in the average number of search steps per trajectory, the proportion of trajectories that contain branches with different rewards, and the proportion of trajectories that are detected to include redundant steps under BranPO.

Method	Pick	Look	Clean	Heat	Cool	Pick2	All
GRPO	90.8	66.1	89.3	74.7	72.5	64.7	77.6
GiGPO	97.7	82.7	**98.8**	83.7	**89.3**	79.2	90.8
BranPO	**99.0**	**89.7**	**98.8**	**95.9**	88.0	**86.1**	**93.6**

Table 3: Performance on ALFWorld benchmark.

Method	2WikiMQA	HotpotQA	MuSiQue	Avg.
BranPO	69.4	59.8	34.5	54.6
w/o DBS	67.8	59.4	30.8	52.7
w/o RSM	68.2	60.0	33.0	53.7

Table 4: Ablation study of BranPO components. DBS denotes for difficulty-aware branch sampling, RSM denotes for redundant step masking. The evaluation metric is F1 score.

ery step, substantially reducing computational cost while preserving effective contrastive learning signals at key decision points.

We evaluate this variant on ALFWorld (Shridhar et al., 2021), a long-horizon embodied task benchmark in a text-based environment, following the GiGPO (Feng et al., 2025a) setting. We use Qwen2.5-7B-Instruct with a maximum interaction horizon of 50, $S = 4$ segments, and a maximum recursive depth of 3. Results are shown in Table 3. BranPO outperforms the strong baseline GiGPO by an average of 2.8 percentage points and achieves better performance than both GRPO and GiGPO on most subtasks, demonstrating strong generalization to other long-horizon domains.

4.3 Ablation Study

We perform ablation studies on Qwen3-4B-Instruct to assess the impact of Difficulty-aware Branch Sampling (DBS) and Redundant Step Masking (RSM) in BranPO. After the first-stage training, models are evaluated on three multi-hop QA benchmarks, with results reported in Table 4. Removing either DBS or RSM leads to performance degradation, demonstrating that both components are essential. DBS improves sampling efficiency by emphasizing informative trajectories, while RSM enhances step-level supervision by filtering redundant reasoning steps. Further discussion of RSM in extended horizon training is included in Appendix E.

Hyperparameter Sensitivity BranPO introduces two hyperparameters: the maximum number of branch samples per step (B) and the maximum number of backtracking steps (R). We study their effects under different settings, with detailed results provided in Appendix C. Even with a lightweight configuration ($B = 4, R = 1$), BranPO still consistently outperforms GRPO, while larger values further improve performance, indicating strong robustness and favorable scaling behavior.

5 Conclusion

In this work, we present Branching Relative Policy Optimization (BranPO), an RL framework for long-horizon agentic tasks that addresses the credit assignment challenges posed by sparse rewards and the non-monotonic nature of correctness. BranPO constructs contrastive branches through tail-focused resampling and optimizes them with hierarchical group-relative advantages, transforming sparse outcome signals into localized preference supervision without step-level rewards or expensive tree expansion. Difficulty-aware branch sampling and Redundant Step Masking further improve efficiency and suppress continuation bias. Experiments show that BranPO achieves consistent gains over outcome-reward RL methods, explicit process-reward RL methods, and prior branch sampling methods on multi-hop QA, with generalization to 30+ step agentic tasks. Beyond empirical gains, this work highlights a practical insight: precise credit assignment is not essential; concentrating learning on outcome-divergent decisions under shared histories suffices for robust recovery from imperfect states.

Limitations

Although BranPO provides more efficient updates than trajectory-level baselines, it does not fully resolve the sparse reward challenge. In extremely long-horizon tasks, generating rollouts to obtain outcome signals remains costly, and tail-focused resampling only partially alleviates this overhead. Integrating lightweight intermediate reward models or strategies that guide higher-quality rollouts to further reduce sampling cost is a promising direction for future work.

References

Guanting Dong, Licheng Bao, Zhongyuan Wang, Kangzhi Zhao, Xiaoxi Li, Jiajie Jin, Jinghan Yang, Hangyu Mao, Fuzheng Zhang, Kun Gai, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. 2026a. Toward generalized web agent training: A deep dive into entropy-balanced reinforcement learning. In *Proceedings of the ACM Web Conference 2026*, WWW '26, page 2126–2137, New York, NY, USA. Association for Computing Machinery.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. 2026b. Agentic reinforced policy optimization. In *The Fourteenth International Conference on Learning Representations*.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. 2025a. Group-in-group policy optimization for LLM agent training. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Yunlong Feng, Yang Xu, Xiao Xu, Binyuan Hui, and Junyang Lin. 2025b. Towards better correctness and efficiency in code generation. *arXiv preprint arXiv:2508.20124*.

Daocheng Fu, Jianbiao Mei, Licheng Wen, Xuemeng Yang, Cheng Yang, Rong Wu, Tao Hu, Siqi Li, Yufan Shen, Xinyu Cai, and 1 others. 2025. Re-searcher: Robust agentic search with goal-oriented planning and self-reflection. *arXiv preprint arXiv:2509.26048*.

Jiaxuan Gao, Wei Fu, Minyang Xie, Shusheng Xu, Chuyi He, Zhiyu Mei, Banghua Zhu, and Yi Wu. 2025. Beyond ten turns: Unlocking long-horizon agentic search with large-scale asynchronous rl. *arXiv preprint arXiv:2508.07976*.

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

Yuxiang Ji, Ziyu Ma, Yong Wang, Guanhua Chen, Xiangxiang Chu, and Liaoni Wu. 2026. Tree search for LLM agent reinforcement learning. In *The Fourteenth International Conference on Learning Representations*.

Pengcheng Jiang, Jiacheng Lin, Lang Cao, Runchu Tian, SeongKu Kang, Zifeng Wang, Jimeng Sun, and Jiawei Han. 2025a. Deepretrieval: Hacking real search engines and retrievers with large language models via reinforcement learning. *arXiv preprint arXiv:2503.00223*.

Pengcheng Jiang, Xueqiang Xu, Jiacheng Lin, Jinfeng Xiao, Zifeng Wang, Jimeng Sun, and Jiawei Han. 2025b. s3: You don't need that much data to train a search agent via rl. *arXiv preprint arXiv:2505.14146*.

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

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. 2024. Vineppo: Refining credit assignment in rl training of llms. *arXiv preprint arXiv:2410.01679*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, and 1 others. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Jian Li, Yizhang Jin, Dongqi Liu, Hang Ding, Jiafu Wu, Dongsheng Chen, Yunhang Shen, Yulei Qin, Ying Tai, Chengjie Wang, and 1 others. 2026. Se-search: Self-evolving search agent via memory and dense reward. *arXiv preprint arXiv:2603.03293*.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025a. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366*.

Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yongkang Wu, Ji-Rong Wen, Yutao Zhu, and Zhicheng Dou. 2025b. Webthinker: Empowering large reasoning models with deep research capability. *arXiv preprint arXiv:2504.21776*.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang,

Chaofan Lin, Chen Dong, and 1 others. 2025a. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*.

Dongyi Liu, Yifan Niu, Qinwen Wang, Han Xiao, and Jia Li. 2026. Pica: Pivot-based credit assignment for search agentic reinforcement learning. *arXiv preprint arXiv:2605.09287*.

Jiacai Liu, Yingru Li, Yuqian Fu, Jiawei Wang, Qian Liu, and Yu Shen. 2025b. When speed kills stability: Demystifying RL collapse from the training-inference mismatch. https://richardli.xyz/rl-collapse. Notion Blog.

Zichen Liu, Anya Sims, Keyu Duan, Changyu Chen, Simon Yu, Xiangxin Zhou, Haotian Xu, Shaopan Xiong, Bo Liu, Chenmien Tan, and 1 others. 2025c. Gem: A gym for agentic llms. *arXiv preprint arXiv:2510.01051*.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2023. Gaia: a benchmark for general ai assistants. In *The Twelfth International Conference on Learning Representations*.

MiniMax. 2025. Minimax m2 & agent: Ingenious in simplicity. https://www.minimax.io/news/minimax-m2.

Moonshot. 2025. Kimi-researcher: End-to-end rl training for emerging agentic capabilities. https://moonshotai.github.io/Kimi-Researcher/.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

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

Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. 2025. Toolrl: Reward is all tool learning needs. *arXiv preprint arXiv:2504.13958*.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, and 1 others. 2025. Ui-tars: Pioneering automated gui interaction with native agents. *arXiv preprint arXiv:2501.12326*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Leyang Shen, Yang Zhang, Chun Kai Ling, Xiaoyan Zhao, and Tat-Seng Chua. 2025. Carl: Critical action focused reinforcement learning for multi-step agent. *arXiv preprint arXiv:2512.04949*.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pages 1279–1297.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Cote, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2021. Alfworld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*.

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

Sijun Tan, Michael Luo, Colin Cai, Tarun Venkat, Kyle Montgomery, Aaron Hao, Tianhao Wu, Arnav Balyan, Manan Roongta, Chenguang Wang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. 2025. rllm: A framework for post-training language agents. https://pretty-radio-b75.notion.site/rLLM-A-Framework-for-Post-Training-Language-Agents-21b81902c146819db63cd98a54ba5f31. Notion Blog.

MiroMind Team, Song Bai, Lidong Bing, Carson Chen, Guanzheng Chen, Yuntao Chen, Zhe Chen, Ziyi Chen, Jifeng Dai, Xuan Dong, and 1 others. 2025a. Mirothinker: Pushing the performance boundaries of open-source research agents via model, context, and interactive scaling. *arXiv preprint arXiv:2511.11793*.

Tongyi DeepResearch Team, Baixuan Li, Bo Zhang, Dingchu Zhang, Fei Huang, Guangyu Li, Guoxin Chen, Huifeng Yin, Jialong Wu, Jingren Zhou, and 1 others. 2025b. Tongyi deepresearch technical report. *arXiv preprint arXiv:2510.24701*.

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

Ziliang Wang, Xuhui Zheng, Kang An, Cijun Ouyang, Jialu Cai, Yuhang Wang, and Yichao Wu. 2025. Stepsearch: Igniting llms search ability via step-wise proximal policy optimization. *arXiv preprint arXiv:2505.15107*.

Zhiheng Xi, Jixuan Huang, Chenyang Liao, Baodai Huang, Honglin Guo, Jiaqi Liu, Rui Zheng, Junjie Ye, Jiazheng Zhang, Wenxiang Chen, and 1 others. 2025. Agentgym-rl: Training llm agents for long-horizon decision making through multi-turn reinforcement learning. *arXiv preprint arXiv:2509.08755*.

Yutao Xie, Nathaniel Thomas, Nicklas Hansen, Yang Fu, Li Erran Li, and Xiaolong Wang. 2026. Tips: Turn-level information-potential reward shaping for search-augmented llms. *arXiv preprint arXiv:2603.22293*.

Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Xiaosen Zheng, Zejun MA, and Bo An. 2025. SimpleTIR: End-to-end reinforcement learning for multi-turn tool-integrated reasoning. In *NeurIPS 2025 Fourth Workshop on Deep Learning for Code*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Zhicheng Yang, Zhijiang Guo, Yinya Huang, Xiaodan Liang, Yiwei Wang, and Jing Tang. 2025b. Treerpo: Tree relative policy optimization. *arXiv preprint arXiv:2506.05183*.

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

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.

Siliang Zeng, Quan Wei, William Brown, Oana Frunza, Yuriy Nevmyvaka, Yang Katie Zhao, and Mingyi Hong. 2025. Reinforcing multi-turn reasoning in LLM agents via turn-level credit assignment. In *ICML 2025 Workshop on Computer Use Agents*.

Wenlin Zhang, Xiangyang Li, Kuicai Dong, Yichao Wang, Pengyue Jia, Xiaopeng Li, Yingyi Zhang, Derong Xu, Zhaocheng Du, Huifeng Guo, and 1 others. 2025a. Process vs. outcome reward: Which is better for agentic rag reinforcement learning. *arXiv preprint arXiv:2505.14069*.

Yaocheng Zhang, Haohuan Huang, Zijun Song, Yuanheng Zhu, Qichao Zhang, Zijie Zhao, and Dongbin Zhao. 2025b. Criticsearch: Fine-grained credit assignment for search agents via a retrospective critic. *arXiv preprint arXiv:2511.12159*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, YeYanhan YeYanhan, and Zheyan Luo. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410.

A Implement Details

A.1 Algorithm Details

The pseudocode for BranPO with Difficulty-aware Sampling is presented in Algorithm 1. An F1 score of 0.8 is used as a threshold to determine whether an answer is considered correct during contrastive sampling. And task accuracy is calculated by the average F1 score of the initial rollouts in the same task group. In our experiments, the branch and recursion schedule criteria were selected to maximize the difference signals within the branches without increasing the total step time. Consequently, our branch sampling criteria are defined as follows:

$$\text{Bran}(r, \text{acc}) = \begin{cases} 1, & \text{if acc} = 1 \\ 2, & \text{otherwise} \end{cases} \tag{8}$$

$$\text{Recur}(r, \text{acc}) = \begin{cases} 3, & \text{if acc} < 0.5 \text{ and } r < 0.8 \\ 2, & \text{if acc} < 1 \text{ or } r < 0.8 \\ 1, & \text{otherwise} \end{cases} \tag{9}$$

which means:

- If the task accuracy is equal to 1 (all correct), only the final step is truncated and one branch is sampled.
- If the task accuracy is below 0.5 and the trajectory-level F1 score is below 0.8, the trajectory is recursively truncated by moving backward from the last step to the third-to-last step, with two branches sampled at each truncation, until a correct branch is found.
- Otherwise, the trajectory is recursively truncated from the last step to the second-to-last step, with two branches sampled at each truncation, until a contrastive branch is found.

The selection of the branch and recursion budget hyperparameters above is primarily intended to ensure fairness in training time between the BranPO algorithm and other methods. We also experimented with additional hyperparameter settings to study the sensitivity of BranPO. Detailed results are provided in C. The results show that even under lower training budgets, BranPO can still outperform other methods.

Regarding Redundant Mask detection, for samples where accuracy > 0.5 and $r = 1$, we execute a truncation at the penultimate step. Subsequently, we branch 5 times until a more efficient step is identified.

A.2 Cold Start

When conducting RL with Instruct models, we observed instability in the later stages of training. To mitigate this issue, we adopted a SFT to stabilize multi-turn training. The construction pipeline for the cold-start data is as follows. First, we perform zero RL training with GRPO using the Search-R1 (Jin et al., 2025) training data, which consists of NQ (Kwiatkowski et al., 2019) and HotpotQA (Yang et al., 2018), with the turn limit set to 4 and training for 80 iterations. Next, from the rollout data generated during training, we filter out the correct samples, randomly select 10,000 instances, and then reconstruct the tag format to obtain a multi-turn SFT dataset. For each model, the cold-start data are derived from its own zero RL training. Finally, we conduct SFT using LLaMA-Factory (Zheng et al., 2024) with a learning rate of 1×10^{-5}, a batch size of 256, and train for 1 epoch.

A.3 RL Training

Our training framework is based on rLLM (Tan et al., 2025), an agentic RL training system built on veRL (Sheng et al., 2025). We introduce two fixes to the original rLLM framework:

- **Token-in-token-out.** In multi-turn training, each round's prompt includes the interaction history from previous rounds. In the original rLLM implementation, the prompt is re-encoded by the tokenizer at every turn. We observe that the re-tokenized sequence may not exactly match the tokens previously generated by the LLM, introducing off-policy effects that harm training stability. To address this issue, we cache the token IDs produced by the LLM at each turn and concatenate them with the existing prompt tokens and encoded environment tokens to form the next-turn prompt, avoiding repeated re-encoding and decoding.
- **Rollout Correction.** This technique mitigates off-policy effects caused by distributional discrepancies between the inference engine and the training engine. We apply truncated token-level importance sampling to reduce this mismatch (Liu et al., 2025b).

During training, we use the F1 score as the primary reward. We also incorporate a format reward: if the output does not follow either `<thinking>...</thinking><search>...</search>` or `<thinking>...</thinking><answer>...</a`

Algorithm 1 BranPO update step with Difficulty-aware Branch Sampling

Require: Policy π_θ, reward $R(\cdot)$, initial rollout number N, branch schedule $\textsc{Bran}(\cdot)$, recursion schedule $\textsc{Recur}(\cdot)$

```
function InitialRollout(g, N)
    Sample trajectories {τ_i^(g)}_{i=1}^N ~ π_θ
    Calculate group accuracy acc^(g) based on {τ_i^(g)}_{i=1}^N
    return ({τ_i^(g)}_{i=1}^N, acc^(g))
end function

function Branching(τ_i^(g), acc^(g))
    for d = 1 to Recur(R(τ_i^(g)), acc^(g)) do
        Select truncation index k_d near tail B_{g,i} ← τ_{i,≤k_d}^(g)
        C_{g,i} ← (τ_{i,>k_d}^(g), R(τ_i^(g)))
        for j = 1 to Bran(R(τ_i^(g)), acc^(g)) do
            Sample τ_{>k_d}^(g,i,j) ~ π_θ(· | B_{g,i})
            τ̃^(g,i,j) ← (B_{g,i}, τ_{>k_d}^(g,i,j))
            if R(τ̃^(g,i,j)) ≠ R(τ̃^(g,i)) then
                Add (τ̃^(g,i,j), R(τ̃^(g,i,j))) to C_{g,i}
                return C_{g,i}
            end if
        end for
    end for
    return C_{g,i}
end function
for task g do
    ({τ_i^(g)}_{i=1}^N, acc^(g)) ← InitialRollout(g, N)
    for i = 1 to N do
        C_{g,i} ← Branching(τ_i^(g), acc^(g))
        r_base^(g,i) ← (1/|C_{g,i}|) Σ_{(τ,r)∈C_{g,i}} r
        Â_base^(g,i) ← (r_base^(g,i) − μ_r^base) / σ_r^base
        for branch j ∈ C_{g,i} do
            Â_branch^(g,i,j) ← (r_branch^(g,i,j) − μ_r^branch(g,i)) / σ_r^branch(g,i)
        end for
    end for
end for
Update θ using GRPO-style policy gradient with BranPO advantages
```

Hyperparameter	Value
Batch size	256
PPO mini-batch size	256
Max response length	3,072 (4-turn limit), 7,168 (8-turn limit)
KL loss coefficient	0.001
KL type	k3
Loss aggregate mode	token-mean
Rollout IS threshold	2.0
Rollout IS batch normalize	True
Rollout engine	vLLM (Kwon et al., 2023)
Rollout temperature	0.95
Optimizer	AdamW
Learning rate	5×10^{-6} (ASearcher); 1×10^{-6} (Search-R1)
Learning rate warm-up ratio	0.0
Adam β_1, β_2	0.9, 0.999
Gradient clip norm	1.0
Training backend	FSDP
retrieval topk	3

Table 5: Training Hyperparameters.

nswer>, or the model answer the question without one search, a reward of -0.5 is assigned to the entire trajectory.

The prompt template used in our experiments follows (Jin et al., 2025) and is shown in Table 6. We replace <think> and </think> with <thinking> and </thinking> since <think> is a special token in Qwen3 series models. All other training hyperparameters are reported in Table 5.

Answer the given question. You must conduct reasoning inside <thinking> and </thinking> first every time you get new information. After reasoning, if you find you lack some knowledge, you can call a search engine by <search> query </search> and it will return the top searched results between <information> and </information>. You can search as many times as your want. If you find no further external knowledge needed, you can directly provide the answer inside <answer> and </answer>, without detailed illustrations. For example, <answer> Beijing </answer>. Question: question.

Table 6: Prompt Template.

A.4 Evaluation Details

Baselines We mainly compare BranPO with three categories of baseline reinforcement learning algorithms:

1. **Outcome-reward-based RL methods**, including GRPO (Shao et al., 2024), Search-R1 (Jin et al., 2025), and SE-Search (Li et al., 2026).

2. **Explicit turn-level process reward methods**, including ReasonRAG (Zhang et al., 2025a), StepSearch (Wang et al., 2025), GiGPO (Feng et al., 2025a), CriticSearch (Zhang et al., 2025b), and TIPS (Xie et al., 2026).

3. **Branch-sampling-based GRPO methods**, including Tree-GRPO (Ji et al., 2026) and entropy-guided branching strategies ARPO (Dong et al., 2026b) and AEPO (Dong et al., 2026a).

These three categories cover the vast majority of algorithmic improvements in agentic RL. Due to computational resource constraints, under the ASearcher setting, we only compare against GRPO and branch-sampling-based GRPO methods, as these methods generally exhibit stronger performance in short-horizon scenarios.

Our training data follows ASearcher (Gao et al., 2025); however, ASearcher still adopts the GRPO

algorithm, with the number of interaction rounds extended to 32. Nevertheless, in the ASearcher dataset, the vast majority of problems can already be solved within 8 interaction steps. We therefore believe that extending the interaction horizon to 32 steps on this dataset is unnecessary, as it mainly introduces longer training time and may encourage redundant search behaviors. Therefore, we do not compare directly with ASearcher, but instead compare with the results of GRPO under our training budget.

Evaluation Data We adopt the exact same evaluation data and sampling protocol as in ASearcher (Gao et al., 2025). Specifically, our evaluation suite includes:

- **Multi-hop QA datasets**: 2WikiMultihopQA (Ho et al., 2020) (12,576 examples), HotpotQA (Yang et al., 2018) (7,405 examples), MuSiQue (Trivedi et al., 2022) (2,417 examples), and Bamboogle (Press et al., 2023) (125 examples);
- **Web QA dataset**: GAIA (Mialon et al., 2023) (103 text-only instances).

Under the ASearcher setting, for all datasets except Bamboogle and GAIA, we only use the randomly sampled 1,000 examples adopted in ASearcher (Gao et al., 2025).

Evaluation Settings During evaluation, we use the following generation parameters: temperature = 0.7 and top_p = 0.8.

Method	Time Composition (s)			Total (s)
	Rollout	Compute log_prob	Update actor	
GRPO	108	75	251	439
Tree-GRPO	103	71	248	424
BranPO	191	55	182	429

Table 7: Wall-clock time analysis of GRPO and BranPO. We report the average time over the first five training steps.

A.5 Training Efficiency

To ensure a fair comparison between GRPO, Tree-GRPO and BranPO, we analyze their wall-clock training time under a single training round. We report the average time of the first five training steps of Qwen2.5-7B-Instruct at Table 7, since later-stage training time is strongly affected by the average number of environment interactions per trajectory.

As shown in Table 7, compared to GRPO and Tree-GRPO, BranPO introduces additional rollout overhead due to branching-based trajectory construction. However, this increased rollout cost is largely offset by a reduced policy update time, as BranPO requires fewer effective training samples for optimization. As a result, the overall wall-clock training time of BranPO remains comparable to that of GRPO and Tree-GRPO.

For ARPO and AEPO, we adopt their original rollout configurations to fully exploit the benefits of their entropy-guided branching strategies: the initial rollout size is set to 8, and the final number of trajectories is expanded to 16. In contrast, the rollout setting of GRPO in Table 7 uses only 8 trajectories. Therefore, although ARPO and AEPO incur substantially higher computational costs than GRPO, Tree-GRPO, and BranPO, their performance still remains inferior to our method.

B Generalize to web scenario.

We further evaluate BranPO in a real-world web search scenario to assess its generalization ability, with results reported in Table 9. We evaluate our methods on the GAIA benchmark (Mialon et al., 2023), reporting the Avg@4 metric which averages the F1 scores and LLM-as-a-Judge scores across four independent runs. During evaluation, each search retrieves up to 10 web pages, which are then processed by a summarization model instantiated with Qwen3-30B-A3B-Instruct.

On the GAIA benchmark, BranPO consistently outperforms GRPO on both Qwen2.5-7B-Instruct and Qwen3-4B-Instruct, demonstrating strong generalization to real-world web search tasks.

For LLM-as-a-Judge, the prompt we adopt follows ASearcher (Gao et al., 2025), as shown in Table 10.

To provide the model with cleaner and more relevant information during web search tasks, we employ a summarization step after retrieving web pages. Specifically, we use a dedicated summarization model to extract query-relevant content from each retrieved webpage. Across all evaluations, we use `Qwen3-30B-A3B-Instruct` as the summarization model, with the prompt template shown in Table 11.

	2WikiMQA	HotpotQA	MuSiQue	Avg.
GRPO	62.4	58.6	31.1	50.7
Tree-GRPO	63.9	56.7	31.9	50.8
BranPO				
No branch (B=0,R=1)	58.6	56.8	30.1	48.5
B=2, R=1	61.9	58.3	31.7	50.6
B=4, R=1	65.9	59.7	**33.7**	53.1
B=2, R=2	64.2	58.8	31.5	51.5
B=4, R=2	**68.0**	60.7	32.5	53.7
B=2, R=3	67.7	60.9	32.2	53.6
B=4, R=3	67.9	**61.1**	32.5	**53.8**

Table 8: Performance under different branching and recursion budgets.

	Lv.1		Lv.2		Lv.3		All	
Method	F1	LasJ	F1	LasJ	F1	LasJ	F1	LasJ
`Qwen2.5-7B-Instruct`								
GRPO	38.2	38.5	19.3	22.2	**6.0**	0.0	24.9	25.8
BranPO	**41.2**	**41.1**	**26.9**	**28.4**	3.6	0.0	**29.6**	**29.9**
`Qwen3-4B-Instruct-2507`								
GRPO	53.5	57.1	36.8	38.5	**6.1**	**2.1**	39.5	41.3
BranPO	**57.9**	**61.6**	**36.9**	**40.9**	4.8	**2.1**	**41.1**	**44.2**

Table 9: Avg@4 performance on GAIA with real web search engine. The best results are indicated with **bold**.

You are an evaluation assistant. Please determine if the predicted answer is equivalent to any of the labeled answers.

Question: question

Labeled Answers: gt_answers_str

Predicted Answer: pred_answer

Did the model give an answer **equivalent** to any of the labeled answers? Respond with "Correct" if equivalent to at least one, otherwise "Incorrect".

Output in JSON format:

```json
{
"rationale": "your rationale",
"judgement": "Correct" or "Incorrect"
}
```

Table 10: LLM-as-a-Judge Prompt Template.

You are an AI research assistant. Your task is to analyze the provided webpage content and extract information relevant to the user's query.

Search Query: search_query

Webpage Content: page_content

Please analyze the content and return a **valid JSON object** with the following fields:
- "is_helpful": (boolean) Whether the content contains information relevant to the user's query.
- "summary": (string) A concise, information-dense summary of the content relevant to the query. If 'is_helpful' is false, briefly explain why (e.g., "Content is behind a paywall" or "Topic unrelated").
- "relevance_score": (integer) A score from 0-10 indicating how relevant this page is.

Ensure the output is raw JSON, without Markdown formatting (like "'json).

Table 11: Web page summarization Prompt Template

C Hyperparameter Analysis

As introduced in Section A.1, BranPO involves two key hyperparameters: the maximum branching factor (B) and the maximum recursion depth (R). Specifically, B corresponds to the upper bound of extra branch samples at each truncation step, while R denotes the maximum number of backtracking steps allowed in the recursive truncation process. In practice, these values correspond to the largest outputs of the piecewise functions Bran(r, acc) and Recur(r, acc).

To investigate their impact, we evaluate BranPO under different (B, R) configurations , based on the `Qwen2.5-7B-Instruct` model after the first-stage training with ASearcher dataset.The results are shown in Table 8.

From the results, we draw two main observations. First, even with a reduced recursion depth, such as B=4, R=1, BranPO consistently outperforms both GRPO and Tree-GRPO, indicating that strong gains can be achieved with limited backtracking. Second, BranPO demonstrates a clear scaling pattern on multi-hop reasoning tasks: increasing either the branching factor or recursion depth generally leads to improved performance, suggesting that richer exploration of alternative trajectories provides more informative learning signals.

Overall, these results highlight the robustness of BranPO to hyperparameter choices and its ability to effectively trade off computation and performance.

D Connections between GRPO, DPO, and BranPO

This appendix provides a compact theoretical account of (i) the exact relationship between BranPO and GRPO, (ii) why BranPO reduces variance in prefix credit assignment, and (iii) why BranPO's suffix optimization admits a preference-style (DPO-like) interpretation that can be more sample-efficient.

D.1 Setup and Notation

We consider an episodic agentic MDP with trajectories $\tau = (a_1, \ldots, a_T)$ induced by a policy π_θ. Let $R(\tau) \in \{0, 1\}$ denote the outcome reward. Let π_{ref} be a fixed reference policy and $\beta > 0$ the KL coefficient.

For a truncation index k, define the trajectory prefix and suffix as

$$\mathcal{B} \triangleq \tau_{\leq k} = (a_1, \ldots, a_k), \tag{10}$$

$$\mathcal{C} \triangleq \tau_{>k} = (a_{k+1}, \ldots, a_T) \tag{11}$$

We write concatenation as $(\mathcal{B}, \mathcal{C})$.

BranPO samples M suffix continuations $\{\mathcal{C}^{(i)}\}_{i=1}^{M}$ from the conditional policy $\pi_\theta(\cdot \mid \mathcal{B})$ and evaluates branch rewards

$$r_{\text{branch}}^{(i)} \triangleq R(\mathcal{B}, \mathcal{C}^{(i)}), \tag{12}$$

$$r_{\text{base}}(\mathcal{B}) \triangleq \frac{1}{M} \sum_{i=1}^{M} r_{\text{branch}}^{(i)} \tag{13}$$

Both GRPO and BranPO use the same KL-regularized policy optimization loss; BranPO differs only in the construction/assignment of advantages. Abstracting away implementation details

(e.g., token-level sums, masking), the shared structure is

$$\mathcal{L}(\theta) = -\mathbb{E}\Big[\sum_{t=1}^{T} \hat{A}_t \log \pi_\theta(a_t \mid s_t)\Big] + \beta\, \mathbb{E}\Big[\sum_{t=1}^{T} \mathrm{KL}\big(\pi_\theta(\cdot \mid s_t) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid s_t)\big)\Big] \tag{14}$$

GRPO assigns a single trajectory-level advantage to all steps, while BranPO assigns a base advantage to prefix steps and branch advantages to suffix steps. Group normalization (as in GRPO) applies an affine transform within each task group and therefore does not change the arguments below except for per-group scaling; we omit it when analyzing unbiasedness/variance and reintroduce it in the final statements.

D.2 BranPO Reduces to GRPO

Theorem D.1 (Reduction to GRPO)**.** *Assume BranPO samples exactly one continuation per prefix ($M = 1$) and uses the unmodified rollout suffix (equivalently, no effective truncation/branching). Then $r_{\mathrm{base}}(\mathcal{B}) = R(\tau)$ and the BranPO update is identical to GRPO, including the KL regularization term.*

Proof. When $M = 1$ and the continuation is the rollout suffix, we have $r^{(1)}_{\mathrm{branch}} = R(\tau)$ and thus

$$r_{\mathrm{base}}(\mathcal{B}) = \frac{1}{1} r^{(1)}_{\mathrm{branch}} = R(\tau) \tag{15}$$

Hence the advantage assigned to prefix steps equals the usual trajectory return (up to the same within-group normalization used by GRPO), and there are no additional branch trajectories beyond the rollout itself. Because the policy-gradient term and the KL term are unchanged by design, the resulting loss and gradient match GRPO exactly. □

Takeaway. In the absence of effective branches, trajectory prefixes are strictly correct or incorrect. Under these conditions, BranPO becomes equivalent to GRPO, demonstrating that it retains the same capacity to optimize the prefix.

D.3 Prefix Variance Reduction

The core difficulty in multi-turn agent RL is that early actions in $\mathcal{B}$ may be correct but are penalized due to stochastic or fragile failures in the late suffix. BranPO addresses this by estimating the *conditional downstream value* of a prefix,

$$Q^{\pi_\theta}(\mathcal{B}) \triangleq \mathbb{E}_{u \sim \pi_\theta(\cdot \mid \mathcal{B})}\big[R(\mathcal{B}, \mathcal{C})\big] \tag{16}$$

using multiple suffix samples and the sample mean $r_{\mathrm{base}}(\mathcal{B})$.

Lemma D.2 (Unbiased prefix value estimate without filtering)**.** *If $\{u^{(i)}\}_{i=1}^{M}$ are i.i.d. samples from $\pi_\theta(\cdot \mid \mathcal{B})$, then*

$$\mathbb{E}\big[r_{\mathrm{base}}(\mathcal{B}) \mid \mathcal{B}\big] = Q^{\pi_\theta}(\mathcal{B}) \tag{17}$$

Proof. Conditioned on $\mathcal{B}$, the branch rewards $\{r^{(i)}_{\mathrm{branch}}\}$ are i.i.d. with mean $Q^{\pi_\theta}(\mathcal{B})$. Linearity of expectation yields

$$\mathbb{E}\big[r_{\mathrm{base}}(\mathcal{B}) \mid \mathcal{B}\big] = \frac{1}{M}\sum_{i=1}^{M} \mathbb{E}\big[r^{(i)}_{\mathrm{branch}} \mid \mathcal{B}\big] \tag{18}$$

$$= Q^{\pi_\theta}(\mathcal{B}) \tag{19}$$

□

Lemma D.3 (Conditional variance reduction)**.** *Under the assumptions of Lemma D.2,*

$$\mathrm{Var}\big(r_{\mathrm{base}}(\mathcal{B}) \mid \mathcal{B}\big) = \frac{1}{M}\mathrm{Var}\big(R(\mathcal{B}, \mathcal{C}) \mid \mathcal{B}\big) \tag{20}$$

$$\leq \frac{1}{4M} \tag{21}$$

Proof. By i.i.d. sampling, the variance of the mean scales as $1/M$:

$$\mathrm{Var}\big(r_{\mathrm{base}}(\mathcal{B}) \mid \mathcal{B}\big) = \frac{1}{M^2}\sum_{i=1}^{M} \mathrm{Var}\big(r^{(i)}_{\mathrm{branch}} \mid \mathcal{B}\big)$$

$$= \frac{1}{M}\mathrm{Var}\big(r^{(1)}_{\mathrm{branch}} \mid \mathcal{B}\big) \tag{22}$$

Since $R(\cdot) \in \{0, 1\}$, $\mathrm{Var}(R) \leq 1/4$, which gives the final inequality. □

Consider the (unclipped) REINFORCE-style prefix term for a fixed prefix $\mathcal{B}$:

$$G_{\mathrm{prefix}} \triangleq r_{\mathrm{base}}(\mathcal{B})\, \nabla_\theta \log \pi_\theta(\mathcal{B}), \tag{23}$$

$$\log \pi_\theta(\mathcal{B}) \triangleq \sum_{t=1}^{k} \log \pi_\theta(a_t \mid s_t) \tag{24}$$

Conditioned on $\mathcal{B}$, the randomness in G_{prefix} arises solely from $r_{\mathrm{base}}(\mathcal{B})$, hence

$$\mathrm{Var}\big(G_{\mathrm{prefix}} \mid \mathcal{B}\big) = \|\nabla_\theta \log \pi_\theta(\mathcal{B})\|^2\, \mathrm{Var}\big(r_{\mathrm{base}}(\mathcal{B}) \mid \mathcal{B}\big) \tag{25}$$

By Lemma D.3, this conditional variance contracts by a factor $1/M$ compared to using a single rollout return. This is the formal mechanism by which branching stabilizes prefix credit assignment.

In practice, BranPO keeps only *contrastive* branching outcomes. This selection changes the sampling distribution and generally introduces bias relative to $Q^{\pi_\theta}(\mathcal{B})$. However, the same mechanism still reduces variance of the *prefix learning signal* because (i) it aggregates multiple suffix outcomes, and (ii) it compresses the range of possible base returns.

To make the compression explicit, consider binary rewards $R(\tau) \in \{0, 1\}$ and a contrastive rule that retains prefixes only when the retained branch set contains at least one success and at least one failure:

$$\exists\, i, j \;\text{ s.t. }\; r_{\text{branch}}^{(i)} = 1,\; r_{\text{branch}}^{(j)} = 0$$

Lemma D.4 (Range compression under contrastive filtering)**.** *If $R \in \{0, 1\}$ and the retained branch set for a prefix $\mathcal{B}$ contains at least one 1 and one 0, then*

$$r_{\text{base}}(\mathcal{B}) \in \Big[\frac{1}{M},\, 1 - \frac{1}{M}\Big]$$

and therefore, for any random variable supported on this interval,

$$\mathrm{Var}\big(r_{\text{base}}(\mathcal{B})\big) \le \frac{(1 - 2/M)^2}{4} < \frac{1}{4}$$

Proof. With at least one success and one failure among M Bernoulli samples, the number of successes is an integer in $\{1, \ldots, M-1\}$, hence the mean is in $[1/M, 1 - 1/M]$. Any random variable supported on $[a, 1-a]$ has variance at most $(1-2a)^2/4$ (achieved by a two-point distribution at the endpoints), yielding the bound with $a = 1/M$. □

Takeaway. Unlike GRPO, which relies on a single binary outcome outcome ($\{0, 1\}$) per rollout, contrastive branching keeps the base reward $r_{\text{base}}(\mathcal{B})$ away from extremes. This mitigates large-magnitude, high-variance prefix updates driven by tail stochasticity, effectively stabilizing training without requiring expensive Monte Carlo sampling.This mechanism aligns directly with BranPO's design goal: to efficiently attenuate ambiguous credit assignment in scenarios where a single prefix can lead to both success and failure.

D.4 Suffix Updates as Preference-Style Optimization

BranPO's suffix learning signal becomes preference-style because branches share the same prefix $\mathcal{B}$ and differ only in the continuation $\mathcal{C}$.

Suppose from a fixed prefix $\mathcal{B}$ we obtain a "good" suffix $\mathcal{C}^+$ and a "bad" suffix $\mathcal{C}^-$ such that

$$R(\mathcal{B}, \mathcal{C}^+) > R(\mathcal{B}, \mathcal{C}^-)$$

This defines an implicit preference pair conditioned on $\mathcal{B}$.

Consider the suffix-only log-probabilities $\log \pi_\theta(\mathcal{C} \mid \mathcal{B})$. If the branch advantages are centered so that the two suffixes receive opposite weights,

$$\hat{A}_{\text{branch}}(\mathcal{C}^+) = +c, \qquad \hat{A}_{\text{branch}}(\mathcal{C}^-) = -c$$

then the suffix contribution to the policy-gradient term is proportional to

$$\begin{aligned} &-c\,\nabla_\theta \log \pi_\theta(\mathcal{C}^+ \mid \mathcal{B}) + c\,\nabla_\theta \log \pi_\theta(\mathcal{C}^- \mid \mathcal{B}) \\ &\quad = -c\,\nabla_\theta\Big(\log \pi_\theta(\mathcal{C}^+ \mid \mathcal{B}) - \log \pi_\theta(\mathcal{C}^- \mid \mathcal{B})\Big) \end{aligned} \tag{26}$$

Thus, BranPO's suffix updates increase the conditional log-likelihood ratio

$$\Delta_\theta(\mathcal{B}) \triangleq \log \pi_\theta(\mathcal{C}^+ \mid \mathcal{B}) - \log \pi_\theta(\mathcal{C}^- \mid \mathcal{B})$$

while simultaneously applying the same KL regularization to π_{ref} as in GRPO.

Connection to DPO. DPO optimizes a logistic preference likelihood with a reference correction. For a conditional preference pair $(\mathcal{B}, \mathcal{C}^+, \mathcal{C}^-)$, define

$$z_\theta(\mathcal{B}) \triangleq \beta\Big(\Delta_\theta(\mathcal{B}) - \Delta_{\text{ref}}(\mathcal{B})\Big), \tag{27}$$

$$\Delta_{\text{ref}}(\mathcal{B}) \triangleq \log \pi_{\text{ref}}(\mathcal{C}^+ \mid \mathcal{B}) - \log \pi_{\text{ref}}(\mathcal{C}^- \mid \mathcal{B}) \tag{28}$$

A DPO-style loss on suffixes is

$$\mathcal{L}_{\text{DPO-suffix}}(\theta) = -\mathbb{E}\Big[\log \sigma\big(z_\theta(\mathcal{B})\big)\Big]$$

Its gradient for one pair is

$$\nabla_\theta \mathcal{L}_{\text{DPO-suffix}}(\theta) = -\beta\big(1 - \sigma(z_\theta(\mathcal{B}))\big)\,\nabla_\theta \Delta_\theta(\mathcal{B}) \tag{29}$$

When $z_\theta(\mathcal{B})$ is near 0 (i.e., the model is near-indifferent between $\mathcal{C}^+$ and $\mathcal{C}^-$ under the reference-corrected logit),

$$1 - \sigma(z_\theta(\mathcal{B})) \approx \frac{1}{2}$$

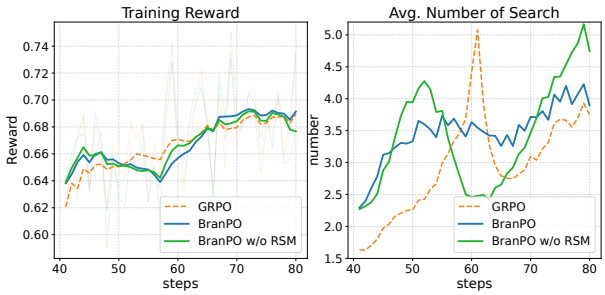

Figure 5: Training dynamics when scaling the number of training interactions to 8, showing the reward and the average number of search steps during training for GRPO, BranPO, and BranPO without RSM on Qwen3-4B-Instruct. RSM effectively suppresses unnecessary growth in search steps.

so the DPO gradient is approximately a constant multiple of $-\nabla_\theta \Delta_\theta(\mathcal{B})$, matching the BranPO suffix direction above (up to a scalar and the explicit KL penalty already present in GRPO/BranPO). Therefore, BranPO's suffix optimization is *DPO-like* in the precise sense of performing pairwise log-ratio improvement on contrastive continuations conditioned on the same intermediate state.

Takeaway. In search agents, errors concentrate near the tail. Branching reuses the same expensive prefix $\mathcal{B}$ and explores only the suffix distribution $\pi_\theta(\cdot \mid \mathcal{B})$, yielding:

- **Compute concentration:** additional sampling cost scales with suffix length rather than full trajectory length.
- **Denser learning signal:** contrastive continuations provide a strong discriminative signal localized to the tail, while prefix gradients are stabilized by $r_{\text{base}}(\mathcal{B})$ (Appendix D.3).

This explains why BranPO can be substantially more sample-efficient than GRPO when failures are dominated by late-stage synthesis.

E Additional Analysis of Redundant Step Masking

This section provides additional analysis of the role of Redundant Step Masking (RSM) under extended training horizons. Starting from the first-stage trained models, we continue optimization into the second stage with the maximum number of training interactions scaled to 8.

As shown in Figure 5, RSM plays a critical role in stabilizing long-horizon optimization by suppressing the unnecessary growth of search steps. In contrast, removing RSM leads to less stable training dynamics, characterized by excessive exploration and a rapid increase in search depth. These results indicate that RSM is particularly important for maintaining efficient and controlled search behavior when scaling the interaction horizon.

F Case Study

F.1 Correctness Non-Monotonic

We present representative cases illustrating **non-monotonic correctness** in multi-turn agentic trajectories. All trajectories shown in these tables are generated by Qwen2.5-7B-Instruct after SFT.

Table 12 presents a positive example of self-correction within the agentic trajectory. In this case, the initial search returns documents that primarily mention Juan Jufré in the context of his activities in Chile. Based on this incomplete evidence, the agent initially misjudges the situation, suspecting that the question's reference to an "Argentine governor" may be a spelling error or a confusion with another historical figure. Crucially, however, the agent does not abandon the task or fabricate an answer. Instead, it recognizes the uncertainty in its intermediate conclusion and decides to refine its search keywords to further verify and correct its earlier judgment, eventually arriving at the correct answer. This case highlights the importance of persistent verification and adaptive search strategies in mitigating non-monotonic correctness.

Table 13 illustrates the first failure mode. In this case, the agent retrieves an outdated piece of information regarding the finance minister of Maharashtra but fails to issue follow-up queries to verify whether this information remains current. Instead of continuing the search to obtain the most up-to-date answer, the agent prematurely draws a conclusion based on the stale result, ultimately producing an incorrect answer.

Table 14 demonstrates the second failure mode. Here, the agent successfully retrieves documents

that collectively contain all the information needed to answer the question correctly. However, rather than carefully synthesizing and summarizing the retrieved evidence, the agent directly generates an answer without grounding it in the search results, leading to a hallucinated response that contradicts the very documents it had gathered.

F.2 Redundant Steps

Here we show an example of redundant steps generated by Qwen2.5-7B-Instruct. As shown in Table 15, the model is able to produce the answer confidently after the first search, however, it does another two search actions to get the answer.

F.3 BranPO

We present example trajectories generated by two BranPO-trained models. As shown in Table 16, the models are able to accurately decompose the questions into multiple search steps and extract key information from the search engine responses. However, we observe that the generated queries still contain meaningless tokens such as "who is." Table 17 shows a failure case. Toward the end of this trajectory, the model's outputs become unstable. This further indicates that finer-grained rewards or higher-quality cold-start initialization are necessary to improve training stability.

Question: Juan Jufré was the governor of the Argentine province that historially comprised which provinces?

Ground Truth: San Juan, San Luis and Mendoza

<thinking> I need to determine which Argentine province Juan Jufré was the governor of, and then find out which provinces were historically part of that province. </thinking>
<search> who was Juan Jufré </search>
<information> [Document 1] Jofre Jofre or Jofré may refer to the following people - Given name - Jofre de Foixà (died c. 1300), Italian poet, musician..... [Document 2] Juan Jufré Accomplishments. Jufré made a fortune developing important economic activities in Chile. He built the first shipyard next to the Maule River, in his encomienda of Pocoa..... [Document 3] Juan Jufré r a textile factory amid his two encomiendas (Mataquito and Peteroa) on the banks of the Mataquito River.....
</information>
<thinking> I found that J̈uan Jufréïs a name associated with Chile, not Argentina. However, the question refers to J̈uan Jufréäs the governor of an Argentine province. This suggests a possible misspelling or confusion. The name J̈uan Jufrém̈ight be a Spanish variant of J̈uan Freireör similar. In Argentina, a common name is J̈uan Joséör J̈uan, and J̈ufrém̈ight be a misspelling of F̈rédaör F̈érte. Alternatively, it could be a confusion with J̈uan Bautistaör J̈uan de la Cruz. Another possibility is that the name is a misspelling of J̈uan Bautista Alberdiör J̈uan Bautistaẗelated to a province. I will conduct a search to find the correct name of the Argentine governor possibly intended in the question. </thinking>
<search> governor of Argentine province with name similar to Juan Jufré </search>
<information> [Document 1] Juan Jufré de Loayza y Montesa was a Spanish conquistador who participated in the 1541 expedition of Pedro de Valdivia to Chile.....
[Document 2] Governor of Jujuy. Province The Governor of Jujuy is a citizen of the Jujuy Province, in Argentina, holding the office of governor for the corresponding period.....
[Document 3] Governor of Formosa Province The Governor of Formosa is a citizen of the Formosa Province, in Argentina.....
</information>
<thinking> From the search, I found that Juan Jufré is referred to as the governor of the Argentine province of Cuyo. Now I need to determine which provinces were historically part of Cuyo. </thinking>
<search> provinces that historically comprised Cuyo in Argentina </search>
<information> [Document 1] Cuyo Province. The Province of Cuyo was a historical Province of Argentina. Created on 14 November 1813 by a decree issued by the Second Triumvirate, it had its capital on Mendoza, and was composed by the territories of the present Argentine Provinces of Mendoza, San Juan and San Luis..... [Document 2] Cuyo (Argentina) Cuyo is the name given to the wine-producing, mountainous area of central-west Argentina. Historically it comprised the provinces of San Juan, San Luis and Mendoza..... [Document 3] For centuries in colonial times, the region was part of the Captaincy General of Chile, at the time part of the Viceroyalty of Peru..... </information>
<thinking> The Argentine province that Juan Jufré was the governor of is Cuyo, which historically comprised the provinces of Mendoza, San Juan, and San Luis. </thinking>
<answer> Mendoza, San Juan, and San Luis </answer>

Table 12: Case Study: Self correction.

Question: Who is the finance minister of the state that generates the most revenue in India?
Ground Truth: Sudhir R. Mungantiwar

— Shared Trajectory —

<thinking> To answer this question, I first need to find out which state in India generates the most revenue. India is a federal structure, and each state has the authority to levy taxes, but not all states are equal in terms of revenue generation. </thinking>
<search> state with the most revenue generation in India </search>
<information> [Document 1] Income in India the world's largest middle income market.
Compared to other countries.....
[Document 2] Economic development in India and even reach the government's 2011 target of .....
[Document 3] Economy of India all crops has grown since 1950, due to the special emphasis placed on agriculture in the five-year plans and steady improvements in irrigation, technology..... </information>
<thinking> From the information provided, it seems that the economy of states like Gujarat, Maharashtra, Tamil Nadu, and others have been growing significantly. However, the specific state that generates the most revenue is not explicitly stated. Typically, the state of Maharashtra is known for being a significant contributor to the country's economy and revenue, due to its industrial and commercial activities. Let's verify this with specific data. </thinking>
<search> finance minister of Maharashtra </search>
<information> [Document 1] Diliprao Deshmukh Diliprao Deshmukh
Diliprao Dagdojirao Deshmukh () (born 18 April 1950) is an Indian politician, ...
[Document 2]Eknath Khadse Eknath Khadse
Eaknath Ganpatrao Khadse (born 2 September 1952) is a politician and senior leader of the Bharatiya Janata Party (BJP) in Maharashtra state. He is a Member of Legislative Assembly of Maharashtra from Muktainagar taluka.....
[Document 3] Sanjay Shamrao Dhotre Sanjay Shamrao Dhotre
Sanjay Shamrao Dhotre (born 26 February 1959) is an Indian politician from ..... </information>

Case 1: Failure

<thinking> Given the information, the state of Maharashtra, which is known for being economically significant, has had a finance minister in recent years. Eknath Khadse and Sanjay Dhotre have both held the position of Finance Minister of Maharashtra. To definitively answer the question, we need the current finance minister. Since the date of the query is not specified, let's assume we are looking for the most current information.
Since Sanjay Dhotre is currently a minister in the central government, and without specific recent data, let's consider the most recent finance minister of Maharashtra, which is Eknath Khadse as of the last available information. </thinking>
<answer> Eknath Khadse </answer>

Case 2: Success

<thinking> Based on the information provided, Diliprao Deshmukh was a Minister of State for Finance & Planning in the state of Maharashtra. More notably, Eknath Khadse held the position of Finance Minister of Maharashtra from 2014 to 2015. Since Maharashtra is one of the states that generate significant revenue for India, we can conclude that the finance minister of the state that generates the most revenue in India, in this case, is likely one from Maharashtra. However, to be precise, we need to identify the current finance minister of Maharashtra. </thinking>
<search> current finance minister of Maharashtra </search>
<information> [Document 1] Eknath Khadse Eknath Khadse
Eaknath Ganpatrao Khadse (born 2 September 1952) is a politician and senior leader of the Bharatiya Janata Party (BJP) .....
[Document 2]Sanjay Shamrao Dhotre Sanjay Shamrao Dhotre
Sanjay Shamrao Dhotre (born 26 February 1959) is an Indian politician from the state of Maharashtra .....
[Document 3]Sudhir Mungantiwar Sudhir Mungantiwar
Sudhir Mungantiwar (born 30 July 1962) is a Legislator from the state of Maharashtra, India. He currently serves as the Cabinet Minister..... </information>
<thinking> Based on the current information:

- Eknath Khadse was Finance Minister of Maharashtra from 2014 to 2015.
- Sanjay Shamrao Dhotre is a Minister of State in the Government of India, not currently the Finance Minister of Maharashtra.
- Sudhir Mungantiwar is the current Cabinet Minister of Finance & Planning in the Government of Maharashtra, having taken office in October 2014.

</thinking>
<answer> Sudhir Mungantiwar </answer>

Table 13: Case Study: Insufficient Search.

Question: Which of these artists—Artemisia Gentileschi, Rosa Barba, or Lavinia Fontana—has the earliest birth date?
Ground Truth: Lavinia Fontana

— Shared Trajectory —

<thinking> I need to find the birth dates of each artist to compare and determine which one is the earliest. </thinking>
<search> Artemisia Gentileschi birth date </search>
<information> [Document 1] Gentileschi Gentileschi
Gentileschi can refer to:
Orazio Gentileschi (1563-1639), an Italian painter
Artemisia Gentileschi (1593-1653), a Baroque painter (and Orazio's daughter)
[Document 2] Artemisia Gentileschi and expressive painters of her generation, and is due to be celebrated at the exhibition at the National Gallery in London in 2020.....
[Document 3] Artemisia (film) Artemisia (film)
Artemisia is a 1997 French-German-Italian biographical film about Artemisia Gentileschi, the female Italian Baroque painter. The film was directed by Agnès Merlet, and stars Valentina Cervi and ..... </information>
<thinking> From the search, I found that Artemisia Gentileschi was born on July 8, 1593. Now I will find the birth dates of Rosa Barba and Lavinia Fontana. </thinking>
<search> Rosa Barba birth date </search>
<information> [Document 1] Rosa Barba Rosa Barba
Rosa Barba (born 1972, Agrigento, Italy) is a German-Italian visual artist and filmmaker. Barba is known for using the medium of film and its materiality to create cinematic film installations.....
[Document 2] Meche Barba Meche Barba
Meche Barba (born Mercedes Barba Feito; September 24, 1922 – January 14, 2000), was an American-born Mexican film actress and dancer of the Golden age of Mexican cinema in the 1940s and 1950s.....
[Document 3] Rosalba of the 2012 Italian Olympic Judo team.
Rosalba Neri (b. 1939), an Italian actress also known by the name "Sara Bey" or "Bay"..... </information>
<thinking> From the search, Rosa Barba was born in 1972. Now I will find the birth date of Lavinia Fontana. </thinking>
<search> Lavinia Fontana birth date </search>
<information> [Document 1] Lavinia Fontana Lavinia Fontana
Lavinia Fontana (1552 – August 11, 1614) was a Bolognese Mannerist painter best known for her portraiture. She was trained by her.....
[Document 2] Lavinia Fontana mythological themes which sometimes included female nudes.
Fontana married Gian Paolo Zappi (alternate spellings include.....
[Document 3] Lavinia Fontana was elected into the Accademia di San Luca of Rome. She died in the city on August 11, 1614 and was subsequently buried at Santa Maria sopra Minerva..... </information>

Case 1: Failure	**Case 2: Success**
<thinking> Based on the information provided, Lavinia Fontana was born in 1552. </thinking> <answer> Artemisia Gentileschi </answer>	<thinking> From the search, Lavinia Fontana was born in 1552. Now that we have the birth years of all three artists: • Artemisia Gentileschi: 1593 • Rosa Barba: 1972 • Lavinia Fontana: 1552 Lavinia Fontana has the earliest birth date. </thinking> <answer> Lavinia Fontana </answer>

Table 14: Case Study: Hallucination.

Question: Who is the author of The Confidential Clerk?
Ground Truth: T. S. Eliot

<thinking> To answer who is the author of "The Confidential Clerk," I need to search for this title to see if I have any information about it. </thinking>
<search> The Confidential Clerk book </search>
<information> [Document 1] The Confidential Clerk The Confidential Clerk The Confidential Clerk is a comic verse play by T. S. Eliot. Synopsis. Sir Claude Mulhammer..... [Document 2] The Confidential Clerk in 1953, before transferring to the West End. It played for almost a year at the Duke of York's Theatre before embarking on a UK Tour. It was produced by Henry Sherek and directed by E..... [Document 3] U.S.A. Confidential U.S.A. Confidential U.S.A. Confidential is a 1952 book written by Jack Lait and Lee Mortimer (Crown Publishers). Its theme is crime and corruption..... </information>
<thinking> Now I have a basic idea of what "The Confidential Clerk" is. It's a comic verse play. I need to find out who wrote it. </thinking>
<search> who wrote the confidential clerk play by t.s. eliot </search>
<information> [Document 1] Rosa Barba Rosa Barba
Rosa Barba (born 1972, Agrigento, Italy) is a German-Italian visual artist and filmmaker. Barba is known for using the medium of film and its materiality to create cinematic film installations.....
[Document 2] Meche Barba Meche Barba
Meche Barba (born Mercedes Barba Feito; September 24, 1922 – January 14, 2000), was an American-born Mexican film actress and dancer of the Golden age of Mexican cinema in the 1940s and 1950s.....
[Document 3] Rosalba of the 2012 Italian Olympic Judo team.
Rosalba Neri (b. 1939), an Italian actress also known by the name "Sara Bey" or "Bay"..... </information>
<thinking> Now we have confirmed that "The Confidential Clerk" is a play written by T. S. Eliot. I have the information I need to answer the question. </thinking>
<search> who is the author of The Confidential Clerk </search>
<information> [Document 1] The Confidential Clerk The Confidential Clerk The Confidential Clerk is a comic verse play by T. S. Eliot. Synopsis. Sir Claude Mulhammer..... [Document 2] The Confidential Clerk in 1953, before transferring to the West End. It played for almost a year at the Duke of York's Theatre before embarking on a UK Tour..... [Document 3] U.S.A. Confidential U.S.A. Confidential U.S.A. Confidential is a 1952 book written by Jack Lait and Lee Mortimer (Crown Publishers). Its theme is crime and corruption..... </information>
<thinking> Based on the provided information, "The Confidential Clerk" is a play written by T. S. Eliot. Therefore, the author of "The Confidential Clerk" is T. S. Eliot. </thinking>
<answer> T. S. Eliot </answer>

Table 15: Case Study: Redundant Step Generated by Qwen2.5-7B-Instruct from PopQA.

Question: Who is the federal leader of the political party Ken Epp belongs to?
Ground Truth: Andrew Scheer

<thinking> I need to determine the name of the political party that Ken Epp belongs to, and then find out the federal leader of that party. </thinking>
<search> who is ken epp </search>
<information> [Document 1] Ken Epp Ken Epp
Ken Epp (born May 11, 1939 in Swift Current, Saskatchewan) is a Canadian politician.
Epp was a member of the Conservative Party of Canada in the House of Commons of Canada, representing the riding of.....
[Document 2] Ken Epp in Swift Current, Saskatchewan) is a Canadian politician.
Epp was a member of the Conservative Party of Canada in..... [Document 3] Epp (surname) ian and former politician
Herb Epp (1934–2013), Canadian politician
Jake Epp (born 1939), Canadian executive and former politician
Ken Epp (born 1939), Canadian politician..... </information>
<thinking> Ken Epp is a Canadian politician. He is a member of the Conservative Party of Canada. I need to find out the current federal leader of the Conservative Party of Canada. </thinking>
<search> who is the federal leader of the Conservative Party of Canada </search>
<information> [Document 1] 2017 Conservative Party of Canada leadership election 2017 Conservative Party of Canada leadership election.....
[Document 2] Conservative Party of Canada Conservative Party of Canada
The Conservative Party of Canada (), colloquially known as the Tories, is a federal political party in Canada.....
[Document 3] Conservative Party of Canada transfer of federal government powers to the provinces modeled after the Meech Lake Accord and a tougher stand on "law and order" issues..... </information>
<thinking> Ken Epp is a Canadian politician and a member of the Conservative Party of Canada. The Conservative Party of Canada is the political party he belongs to. The current federal leader of the Conservative Party of Canada is Andrew Scheer. Therefore, the federal leader of the political party Ken Epp belongs to is Andrew Scheer. </thinking>
<answer> Andrew Scheer </answer>

Table 16: Case Study: Successful search trajectory generated by Qwen3-4B-Instruct-BranPO from MuSiQue.

Question: Who is Hieronymus, Son Of Charles Martel's paternal grandmother?
Ground Truth: Alpaida

<thinking> To answer this, I need to know who Hieronymus is and who is his father. I also need to find information about the paternal grandmother of Hieronymus. However, I'm not sure if Hieronymus is a person I'm familiar with, especially in relation to Charles Martel. I will start with a general search to see if I can find any information about Hieronymus. </thinking>
<search> Hieronymus Son Of Charles Martel </search>
<information> [Document 1] Hieronymus, son of Charles Martel Hieronymus, son of Charles Martel
Hieronymus (c. 722 - after 782), was the son of Charles Martel and his mistress Ruodhaid, and so was the brother of Bernard..... [Document 2] Carolingians descended from Charles Martel Carolingians descended from Charles Martel
This is a partial list of male descendants from Charles Martel (686–741) for fifteen generations..... [Document 3] Hieronymus, son of Charles Martel rad (d. 31 January 826), Abbé de Saint-Quentin and an imperial "missi of Charlemagne in 806.
Richarda, married Nithard..... </information>
<thinking> Now I have a basic information about Hieronymus, son of Charles Martel. He is a son of Charles Martel and Ruodhaid. I need to find out who Hieronymus' paternal grandmother is. Given that Hieronymus is a son of Charles Martel, his paternal grandmother would be the mother of Charles Martel. I need to find who Charles Martel's mother is. </thinking>
<search> paternal grandmother of charles martel </search>
<information> [Document 1] Martel Martel
Martel may refer to:
People.....
[Document 2]Carolingians descended from Charles Martel Carolingians descended from Charles Martel
This is a partial list of male descendants from Charles Martel (686–741) for fifteen generations..... [Document 3] Charles Martel (disambiguation) Charles Martel (disambiguation)
Charles Martel (c. 688–741) was a Frankish military and political leader.
Charles Martel may also refer to:..... </information>
<thinking> There is no specific information about the mother of Charles Martel in these documents. Given that Hieronymus is the son of Charles Martel and Ruodhaid, and we need to find Hieronymus' paternal grandmother, we already know that the paternal grandmother of Hieronymus is Ruodhaid. Now, we need to find who Hieronymus' paternal grandmother's father is, which is the same as finding who is the father of Ruodhaid. However, we don't have any information about Ruodhaid's father. Given the available information, we can conclude that Hieronymus' paternal grandmother is Ruodhaid. The question asks for Hieronymus, Son Of Charles Martel's paternal grandmother. Given that Hieronymus is the son of Charles Martel and Ruodhaid, and we need to find the paternal grandmother of Hieronymus, the answer is Ruodhaid. </thinking>
<answer> Ruodhaid </answer>

Table 17: Case Study: Failed search trajectory generated by Qwen2.5-7B-Instruct-BranPO from 2WikiMultihopQA.