UMCTS-GRPO: UNCERTAINTY-AWARE MONTE CARLO TREE SEARCH FOR LLM AGENT REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Tool-using language agents make sequences of decisions but are typically rewarded only for the final outcome. Tree-structured rollouts expose alternative decisions at shared histories, turning terminal rewards into step-level credit, yet existing tree-based agent RL expands branches at random and trusts every comparison equally. We introduce UMCTS-GRPO, which uses a single backed-up branch uncertainty to decide both where to collect rollout evidence and how much to trust it. During collection, uncertainty-aware Monte Carlo tree search steers simulations toward unresolved branches; during optimization, the same statistic weights sibling comparisons through an uncertainty-weighted baseline and a confidence gate, yielding step-level advantages without a critic, process reward model, or step-level annotation. In short: explore what you doubt; learn from what you have resolved. Experiments on seven retrieval-based QA benchmarks and the interactive ALFWorld environment, across five backbones from the Qwen2.5 and Llama3.2 families, show that UMCTS-GRPO consistently improves over strong agent RL baselines and remains effective under tight rollout budgets.

1 INTRODUCTION

A large language model (LLM) agent that solves a task through tool use makes a sequence of intermediate decisions: which query to issue, how to use the evidence it receives, and when to stop searching. Yet the entire interaction is judged only by its final outcome (Yao et al., 2022; Jin et al., 2025). Group Relative Policy Optimization (GRPO) learns from such outcomes without a value model by comparing trajectories sampled for the same prompt (Shao et al., 2024), but it assigns a single trajectory-level advantage to every step. An answer may contain a wasted search, while a wrong answer may follow a useful query whose evidence is later misused. Sampling more trajectories yields more outcomes, but does not reveal which decisions made the difference.

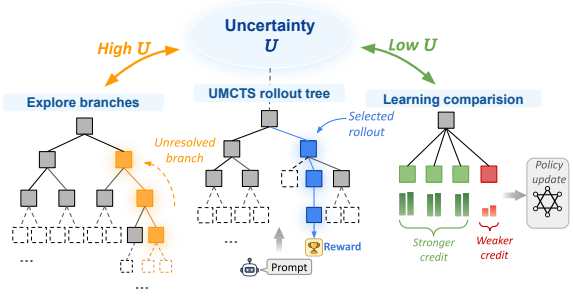


Figure 1: Uncertainty-guided search and credit assignment in UMCTS-GRPO.

Tree-structured rollouts turn interaction structure into a source of this missing supervision. Tree-GRPO branches at complete agent steps and reuses shared prefixes, allowing alternative decisions from the same interaction history to be compared without regenerating it (Ji et al., 2026). Differences in outcome below a branching point induce implicit process preferences from terminal rewards, turning the rollout tree into a scaffold for intermediate credit assignment. Yet the tree is built and read without accounting for branch confidence: branching points are chosen at random, and the resulting process preferences are not weighted by how well their evidence has been resolved.

Once rollouts are organized as a tree, two questions naturally arise: *which branches deserve more evidence, and which comparisons deserve more trust?* Some branches quickly settle into consistent

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

outcomes, while others remain unresolved. Unresolved branches are where additional rollouts are most informative, and also where current comparisons are least reliable. The same branch is thus worth exploring precisely because its comparison is not yet worth trusting. This observation suggests a single principle connecting collection and optimization: *uncertainty should attract exploration but temper learning*. During collection, uncertainty marks where evidence should be gathered; during optimization, it determines how strongly that evidence shapes the policy. In short, *explore what you doubt; learn from what you have resolved*.

We instantiate this principle as **UMCTS-GRPO**, as shown in Figure 1, which makes branch uncertainty the shared interface between tree search and process-level credit assignment. During collection, each rollout backs up return and uncertainty along its search path, and the backed-up uncertainty steers simulations toward unresolved branches. During optimization, the same statistics shape sibling comparisons through two components: an uncertainty-weighted baseline, which determines *what* an action is compared against, and a confidence gate, which determines *how strongly* that comparison counts. Combined with prompt-level outcome advantages, these comparisons transform terminal outcomes into explicit step-level credit, without a critic, process reward model, or step-level annotation. The tree thus becomes a closed feedback loop in which *outcomes update uncertainty, uncertainty redirects exploration, and confidence determines credit*.

We evaluate UMCTS-GRPO on seven retrieval-based QA benchmarks and ALFWorld using five Qwen2.5 and Llama3.2 backbones. It improves over trajectory-level, tree-based, and step-level agent RL baselines and remains effective under tight rollout budgets. Ablations show that uncertainty-guided search provides the main gain, while confidence-weighted credit improves learning. Training dynamics show that UMCTS-GRPO acquires search behavior earlier and sustains it. Our contributions are summarized as follows:

- We identify branch uncertainty as the shared quantity linking rollout collection and process-level credit assignment, and show that a single backed-up statistic can serve both roles asymmetrically: unresolved branches gain exploration priority during search, while their sibling comparisons lose weight during learning until the tree resolves them.
- We instantiate this principle as UMCTS-GRPO, which couples uncertainty-aware Monte Carlo tree search, backing up return and uncertainty to direct simulations toward unresolved branches, with confidence-weighted process advantages that separate *what* an action is compared against from *how strongly* the comparison counts.
- Across QA, ALFWorld, and five backbones, UMCTS-GRPO outperforms strong RL baselines, with ablations, budget studies, and training dynamics validating both components.

2 Related Work

Reinforcement learning for LLM agents. ReAct structures tool use as interleaved reasoning, actions, and observations (Yao et al., 2022), and Search-R1 trains retrieval agents in this format from outcome rewards with observation-token masking (Jin et al., 2025). Group-relative methods such as GRPO remove the critic but assign a single trajectory-level advantage to every step of a rollout (Shao et al., 2024). Finer credit has been sought from additional structure or supervision: process reward models score intermediate steps but require step-level labels or a separately trained verifier (Lightman et al., 2024; Wang et al., 2024); GiGPO groups recurring environment states across rollouts to form step-level advantages (Feng et al., 2026); RLVMR rewards tagged meta-reasoning behaviors (Zhang et al., 2026). UMCTS-GRPO combines local and global signals, but obtains the local signal differently: it actively collects alternatives at a shared prefix, needs no verifier, labels, or behavioral reward rules, and weights each comparison by how well the tree has resolved it.

Tree search for LLM training. Branching alternative continuations is a natural way to expose the consequences of intermediate decisions. VinePPO estimates step values by Monte Carlo rollouts (Kazemnejad et al., 2024), and Segment Policy Optimization shares estimation cost through tree-structured sampling at segment granularity (Guo et al., 2026). Closest to our setting, Tree-GRPO branches at complete agent steps and derives implicit process preferences from within-tree and across-tree reward normalization (Ji et al., 2026). Tree-GRPO, however, expands at randomly chosen nodes and treats every resulting comparison alike. UMCTS-GRPO keeps the agent-step tree

but replaces both decisions: search directs each new rollout by the evidence backed up from earlier ones, and sibling comparisons become explicit step-level advantages weighted by uncertainty. A separate line of work applies tree search at inference time to improve a single response (Yao et al., 2023; Zhou et al., 2023; Koh et al., 2026); UMCTS-GRPO searches only during training and evaluates the policy without test-time search.

Uncertainty-aware exploration and learning. Optimism in the face of uncertainty is a classical principle for allocating exploration. The Uncertainty Bellman Equation propagates value uncertainty to favor unresolved states (O'Donoghue et al., 2018), and Epistemic MCTS carries such estimates through model-based planning (Oren et al., 2025). TreeRL brings uncertainty-guided branching to LLM reinforcement learning (Hou et al., 2025). In all of these, uncertainty decides where to search; it does not decide how much to trust what the search returns. UMCTS-GRPO closes this gap: a single backed-up statistic sets the exploration bonus during collection and, through an uncertainty-weighted baseline and a confidence gate, discounts unresolved comparisons during optimization.

3 PRELIMINARIES

3.1 MULTI-TURN AGENT INTERACTION

We consider an agent that interacts with an environment through thought-action-observation steps (Yao et al., 2022). For a prompt x, let $s_t = (x, H_{<t})$ be the history available before step t. At each step, the policy generates an agent step a_t, consisting of a reasoning thought followed by a textual action, and the environment returns an observation o_t:

$$a_t \sim \pi_\theta(\cdot \mid s_t), \qquad o_t \sim P_{\text{env}}(\cdot \mid s_t, a_t), \qquad s_{t+1} = s_t \oplus (a_t, o_t). \tag{1}$$

An action requests a tool call or completes the task by answering. The trajectory $H = (a_0, o_0, \ldots, a_{T-1}, o_{T-1})$ receives a reward $R(H)$ upon completion, such as exact match or task success. Observations condition the policy but are masked from the loss (Jin et al., 2025).

3.2 GROUP-RELATIVE LEARNING ON ROLLOUT TREES

Given a group $\mathcal{G}_x$ of trajectories for the same prompt, GRPO estimates an outcome advantage by comparing each reward with the group's mean and standard deviation (Shao et al., 2024):

$$A^{\text{out}}(H_i) = \frac{R(H_i) - \mu_x}{\sigma_x + \epsilon}, \qquad \mu_x = \text{mean}_{H_j \in \mathcal{G}_x} R(H_j), \quad \sigma_x = \text{std}_{H_j \in \mathcal{G}_x} R(H_j). \tag{2}$$

Here $\epsilon > 0$ is a numerical stabilizer. In a trajectory-level update, this scalar is shared by all generated tokens of the interaction, so every decision in a trajectory receives the same credit.

An agent-step rollout tree organizes trajectories by their shared histories. A node stores a prefix s, and an outgoing edge (s, a) records a generated agent step together with its observation o, leading to the child prefix $s' = s \oplus (a, o)$. Children of the same node are alternative decisions made from identical preceding information, and the outcomes below them provide evidence about which decision was better. Tree-GRPO grows multiple such trees per prompt by expanding randomly chosen nodes, and combines reward normalization within each tree with normalization across all trees (Ji et al., 2026). Although its advantage remains a trajectory scalar, shared prefixes let the updates induce implicit intermediate preferences. UMCTS-GRPO retains the prompt-level outcome comparison and adds an explicit local advantage at every branching step; the next section describes how uncertainty determines where these branching steps are placed and how much each local advantage counts.

4 UMCTS-GRPO

Each agent-step tree in UMCTS-GRPO serves two purposes: deciding where to collect evidence and deciding how much to learn from it, as shown in Figure 2. Collection alternates between completing a rollout and updating the tree, so every new rollout is placed by the outcomes of all previous ones; optimization then turns the resulting sibling comparisons into confidence-weighted step-level advantages. Algorithm 1 gives the full training iteration.

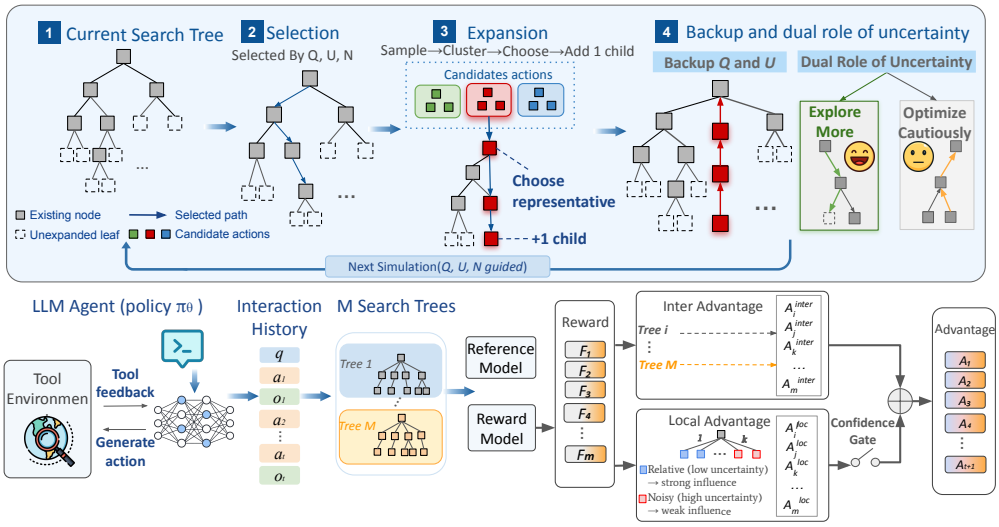


Figure 2: Overview of UMCTS-GRPO. The same backed-up uncertainty plays two complementary roles: it directs rollout budget toward unresolved branches during tree search and down-weights unresolved sibling comparisons during optimization. The resulting process advantages are combined with prompt-level outcome advantages for policy optimization.

4.1 Uncertainty-guided tree search for agent rollouts

Each edge (s, a) maintains a return estimate $Q(s, a)$, an uncertainty statistic $U(s, a) \geq 0$, a prior $P(s, a)$, and a count $N(s, a)$ of completed continuations. During collection, the behavior policy π_{old} is frozen, and each simulation performs selection, expansion, rollout, and backup in sequence.

Candidate actions. Because the language action space cannot be enumerated, each branching point samples a finite candidate pool of K_{cand} next steps from π_{old}, merges near-duplicate actions, and keeps one representative per cluster, whose frequency defines the prior $P(s, a)$. For each prompt we build M independent trees, each drawing its own root pool and initialized by executing its most frequent representative to termination (Appendix E.2).

Selection and expansion. Each subsequent simulation descends from the root. At a fully expanded node it ranks the nonterminal children by

$$S(s, a) = Q(s, a) + c_u \sqrt{U(s, a)} + c_p P(s, a) \frac{\sqrt{N(s)}}{1 + N(s, a)}, \tag{3}$$

where $N(s)$ counts completed continuations through s, priors are normalized over existing siblings, and $c_u, c_p \geq 0$ are exploration coefficients. The value term favors branches that have produced high returns, and the PUCT term (Silver et al., 2017) favors under-explored actions according to their prior and visit count. The uncertainty bonus instead keeps a branch competitive while its evidence remains unsettled, even when its current return estimate is not high: visit count measures how much a branch has been explored, whereas U measures how far the resulting evidence has settled. Descent stops at the first prefix with an untried candidate, executes it, and generates its continuation to termination, so each simulation adds exactly one trajectory.

Uncertainty Bellman backup. A branch can remain unresolved because the sampled action is surprising, uncertainty persists in its descendants, or backed-up returns disagree. Following the uncertainty Bellman equation, we fold these three sources into one statistic and propagate it alongside the return. The completed trajectory updates every edge on its path. Set $V(s_T) = R(H)$ and $U_V(s_T) = 0$ at the terminal state. On the ith backup through (s, a) leading to s', form the targets

$$z_i(s, a) = r(s, a) + \gamma V_i(s'), \qquad u_i(s, a) = u_{\text{local}}(s, a) + \gamma^2 U_{V,i}(s'), \tag{4}$$

where $r(s, a)$ is the immediate reward of the edge and u_{local} is the clipped mean surprisal of the action tokens. For n_Q incorporated targets the edge stores

$$Q(s,a) = \frac{1}{n_Q}\sum_i z_i(s,a), \qquad U(s,a) = \frac{1}{n_Q}\sum_i u_i(s,a) + \text{SampleVar}_i[z_i(s,a)]. \tag{5}$$

A parent's value is the prior-weighted average of its backed-up children, and its uncertainty adds their value disagreement to their propagated uncertainty (Appendix E.1). Repeated agreement drives U down, so U records what the tree has and has not yet settled.

4.2 Confidence-weighted process advantages

A rollout tree exposes alternative decisions at shared histories, but the value estimates of those alternatives are not equally reliable (Figure 3). GRPO assigns one trajectory-level credit to every step; Tree-GRPO lets shared prefixes induce implicit process preferences; UMCTS-GRPO makes the sibling comparison explicit and weights it by its confidence, since treating an unsettled comparison as final can reinforce a continuation that succeeded by chance. To do so, we separate two questions that a single advantage usually conflates: *what* an action is compared against, and *how strongly* that comparison should count.

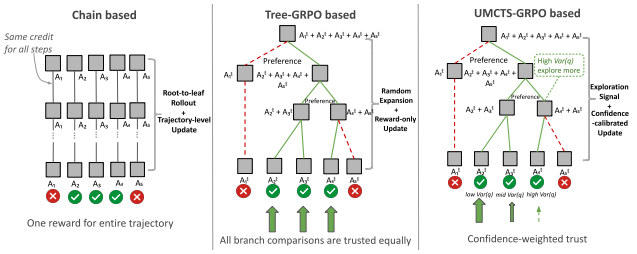


Figure 3: Credit assignment in chain-based GRPO, Tree-GRPO, and UMCTS-GRPO.

Uncertainty-weighted sibling baseline and confidence gate. For a state s with children $\mathcal{A}(s)$, $|\mathcal{A}(s)| \geq 2$, define

$$w(s,a) = \frac{1}{U(s,a)+\epsilon}, \qquad b(s) = \frac{\sum_{a\in\mathcal{A}(s)} w(s,a)Q(s,a)}{\sum_{a\in\mathcal{A}(s)} w(s,a)}. \tag{6}$$

The baseline answers the first question: confident siblings receive more weight, so an action is compared against the alternatives the tree has resolved most clearly. The gate answers the second:

$$g(s,a) = \frac{1}{1+\tau U(s,a)}, \qquad A^{\text{loc}}(s,a) = g(s,a)\frac{Q(s,a)-b(s)}{\sqrt{D(s)+\epsilon}}, \tag{7}$$

where $D(s)$ is the population variance of the sibling values and $\tau \geq 0$ controls attenuation. The denominator expresses the difference in units of the sibling spread, and the gate scales it by the confidence in the action's own outcome.

Combining process and outcome evidence. Sibling comparisons arise wherever the search branched; elsewhere, credit comes from the prompt-level outcome advantage of Equation (2). For step t of trajectory H_i, we use

$$A_{i,t} = \lambda_o A^{\text{out}}(H_i) + \lambda_l A^{\text{loc}}(s_{i,t}, a_{i,t}), \tag{8}$$

with $A^{\text{loc}} = 0$ at unbranched steps. The outcome term ranks trajectories across trees; the local term distinguishes decisions within them. All generated tokens of a step share $A_{i,t}$.

Policy optimization. Let $\rho_{i,\ell}(\theta) = \pi_\theta(y_{i,\ell} \mid x_i, H_{i,<\ell})/\pi_{\text{old}}(y_{i,\ell} \mid x_i, H_{i,<\ell})$, let $t(i,\ell)$ map token ℓ to its agent step, and let $\mathcal{B}_x$ be the batch of trajectories the search produces for prompt x. We optimize the clipped group-relative surrogate (Shao et al., 2024; Schulman et al., 2017)

$$\begin{aligned}\mathcal{J}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\Big[\big\langle \min\big(&\rho_{i,\ell}A_{i,t(i,\ell)},\\ &\text{clip}(\rho_{i,\ell}, 1-\delta, 1+\delta)A_{i,t(i,\ell)}\big)\big\rangle_{\text{valid tokens of } \mathcal{B}_x} - \beta K_{\text{ref}}\Big],\end{aligned} \tag{9}$$

where δ is the clipping threshold and K_{ref} is the KL penalty toward π_{ref} with weight β.

4.3 CONNECTING EXPLORATION AND PROCESS CREDIT

UMCTS-GRPO uses the same statistic U asymmetrically: larger U raises a branch in Equation (3) but lowers its gate in Equation (7), so a branch keeps attracting rollouts while its current preference is discounted, until accumulated evidence settles it. Proposition 1 makes the learning side precise.

Proposition 1 (Properties of confidence-weighted advantages)**.** *Fix a sibling set $\mathcal{A}(s)$ with $|\mathcal{A}(s)| \geq 2$, $\epsilon > 0$, and $\tau \geq 0$.*

(i) ***Sign preservation.*** *For fixed Q, b, and D, $\operatorname{sign}(A^{\mathrm{loc}}(s,a)) = \operatorname{sign}(Q(s,a) - b(s))$, and $|A^{\mathrm{loc}}(s,a)|$ is non-increasing in $U(s,a)$.*

(ii) ***No spurious preference.*** *If $Q(s,a) = q$ for all $a \in \mathcal{A}(s)$, then $A^{\mathrm{loc}}(s,a) = 0$ for all a, regardless of U.*

(iii) ***Consistency.*** *If $U(s,a) = u$ for all $a \in \mathcal{A}(s)$, then $A^{\mathrm{loc}}(s,a) = \big(Q(s,a) - \bar{Q}(s)\big)/\big((1+\tau u)\sqrt{D(s)+\epsilon}\big)$, which recovers the standardized sibling advantage when $\tau = 0$.*

Uncertainty thus changes only how much a comparison counts, never which action it favors, and the estimator reduces to standard sibling normalization when siblings are equally uncertain. Proposition 2 (Appendix C) further shows that, for a pair of siblings, the local term yields the gradient $\alpha^{+}\nabla_\theta \log \pi_\theta(a^{+} \mid s) - \alpha^{-}\nabla_\theta \log \pi_\theta(a^{-} \mid s)$ with confidence-dependent weights $\alpha^{\pm} > 0$, so sibling comparisons act as step-level preference pairs mined from terminal rewards.

5 EXPERIMENTS

5.1 SETTINGS

Datasets. We evaluate on seven QA benchmarks in two categories: single-hop QA with NQ, TriviaQA, and PopQA (Kwiatkowski et al., 2019; Joshi et al., 2017; Mallen et al., 2023), and multi-hop QA with HotpotQA, 2WikiMultiHopQA, MuSiQue, and Bamboogle (Yang et al., 2018; Ho et al., 2020; Trivedi et al., 2022; Press et al., 2023). Single-hop questions are typically solved with one retrieval, whereas multi-hop questions require the agent to coordinate successive searches. Following Ji et al. (2026), single-hop models are trained on NQ and multi-hop models on HotpotQA, so the remaining five benchmarks evaluate out-of-domain generalization. We report exact match (EM) on each benchmark and the macro-average within each category. To evaluate the method beyond retrieval-based QA, we further consider text-based ALFWorld (Shridhar et al., 2020), where actions alter the environment state. Following the GiGPO protocol (Feng et al., 2026), we report success rate for each task category and overall.

Baselines. For QA, we compare with direct inference, ReAct (Yao et al., 2022), Search-o1 (Li et al., 2025), and three RL baselines: chain-based GRPO (Shao et al., 2024), GSPO (Zheng et al., 2025), and Tree-GRPO (Ji et al., 2026), our closest tree-based baseline. We use the same data, retriever, and evaluation protocol as Ji et al. (2026). For ALFWorld, we compare with ReAct, GRPO, GiGPO (Feng et al., 2026), a step-level agent RL method that groups recurring states across trajectories rather than constructing sibling comparisons, and Tree-GRPO. All RL methods use the same outcome reward and interaction interface and are compared under matched rollout budgets.

Implementation. The QA implementation builds on Search-R1 (Jin et al., 2025) with an E5-based Wikipedia retriever (Wang et al., 2022). We evaluate Qwen2.5-1.5B/3B/7B/14B (Hui et al., 2024) and Llama3.2-3B (Grattafiori et al., 2024). Unless otherwise specified, the default QA rollout budget corresponds to four complete trajectories per prompt. ALFWorld experiments use Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct with the GiGPO interaction interface and episode limit. Training data, hyperparameters, and tree configurations are provided in Appendix.

5.2 RESULTS

QA performance. UMCTS-GRPO achieves the highest single-hop macro-average among the compared methods on all five backbones (Table 1). Relative to Tree-GRPO, the gains span every Qwen2.5 scale and Llama3.2-3B, with the largest increase on Llama3.2-3B, from 50.0 to 52.6. On

Table 1: Exact match (%) on single-hop and multi-hop QA. Avg. is the macro-average within each category; $\Delta^{\%}_{\text{rel}}$ is the relative change from GRPO on the same backbone. Bold marks the best result within a backbone, and blue highlights UMCTS-GRPO.

	Method	Single-Hop QA				Multi-Hop QA				
		NQ	Trivia	PopQA	Avg./$\Delta^{\%}_{\text{rel}}$	Hotpot	2wiki	Musiq	Bamb	Avg./$\Delta^{\%}_{\text{rel}}$
Qwen2.5-1.5b	Direct Inference	7.1	22.4	9.9	13.1	5.9	4.3	2.6	8.0	5.2
	Search-o1	10.2	30.9	15.0	18.7	11.6	12.2	3.1	13.0	10.0
	ReAct	9.5	22.1	13.8	15.1	7.3	8.0	1.9	11.2	7.1
	+ GRPO	39.4	51.0	39.7	43.4$_{\Delta_{\text{base}}}$	14.6	24.4	2.2	4.0	11.3$_{\Delta_{\text{base}}}$
	+ GSPO	36.8	48.9	37.3	41.0$_{-5.5\%}$	15.8	23.7	2.5	4.8	11.7$_{+3.5\%}$
	+ Tree-GRPO	43.6	57.3	41.6	47.5$_{+9.5\%}$	**29.5**	**26.8**	6.6	13.6	19.1$_{+69.0\%}$
	+ UMCTS-GRPO	**45.7**	**58.2**	**43.5**	**49.1**$_{+13.2\%}$	**29.5**	26.5	**7.7**	**18.4**	**20.5**$_{+81.6\%}$
Qwen2.5-3b	Direct Inference	10.6	28.8	10.8	16.7	14.9	24.4	2.0	2.4	10.9
	Search-o1	15.1	44.3	13.1	24.2	18.7	17.6	5.8	29.6	17.9
	ReAct	21.1	43.5	28.3	31.0	19.2	19.1	4.8	20.0	15.8
	+ GRPO	44.4	58.0	42.0	48.1$_{\Delta_{\text{base}}}$	39.0	36.3	15.2	36.8	31.8$_{\Delta_{\text{base}}}$
	+ GSPO	43.0	58.8	42.5	48.1$_{+0.0\%}$	40.2	39.8	17.0	36.8	33.5$_{+5.2\%}$
	+ Tree-GRPO	**46.8**	59.7	**43.6**	50.0$_{+4.0\%}$	42.4	**43.7**	17.8	**43.2**	36.8$_{+15.7\%}$
	+ UMCTS-GRPO	45.8	**61.7**	**43.6**	**50.4**$_{+4.8\%}$	**46.8**	**43.7**	**18.4**	42.4	**37.8**$_{+18.9\%}$
Llama3.2-3b	Direct Inference	16.2	29.6	7.4	17.7	12.6	9.2	2.0	8.0	8.0
	Search-o1	24.2	48.4	8.8	27.1	19.4	17.4	6.0	32.0	18.7
	ReAct	23.9	42.4	21.7	29.3	16.2	10.4	3.5	23.2	13.3
	+ GRPO	45.5	58.2	42.4	48.7$_{\Delta_{\text{base}}}$	36.0	26.9	11.8	32.0	26.7$_{\Delta_{\text{base}}}$
	+ GSPO	41.2	57.8	40.8	46.6$_{-4.3\%}$	28.1	24.5	8.6	32.0	23.3$_{-13\%}$
	+ Tree-GRPO	47.7	59.9	42.3	50.0$_{+2.7\%}$	44.6	38.4	17.6	**46.4**	36.8$_{+38.0\%}$
	+ UMCTS-GRPO	**48.8**	**62.7**	**46.4**	**52.6**$_{+8.0\%}$	**47.5**	**44.3**	**19.8**	40.4	**38.0**$_{+42.3\%}$
Qwen2.5-7b	Direct Inference	13.4	40.8	14.0	22.7	18.3	25.0	3.1	12.0	14.6
	Search-o1	23.8	47.2	26.2	32.4	22.1	21.8	5.4	32.0	20.3
	ReAct	30.6	56.3	34.6	40.5	27.9	25.3	11.3	28.8	23.3
	+ GRPO	45.8	61.5	44.3	50.5$_{\Delta_{\text{base}}}$	42.5	40.7	19.1	43.2	36.4$_{\Delta_{\text{base}}}$
	+ GSPO	47.0	64.5	46.1	52.5$_{+4.0\%}$	40.0	38.2	19.2	44.0	35.4$_{-2.8\%}$
	+ Tree-GRPO	48.1	63.3	45.2	52.2$_{+3.4\%}$	44.6	42.3	**20.2**	44.0	37.8$_{+3.8\%}$
	+ UMCTS-GRPO	**49.5**	**65.3**	**46.5**	**53.8**$_{+6.5\%}$	**45.7**	**42.6**	18.7	**44.8**	**38.0**$_{+4.3\%}$
Qwen2.5-14b	Direct Inference	19.8	53.1	18.4	30.4	21.7	25.3	4.5	16.0	16.9
	Search-o1	34.7	63.5	24.1	40.8	26.8	16.1	9.9	41.6	23.6
	ReAct	36.1	64.2	39.3	46.5	39.1	33.8	15.0	43.2	32.8
	+ GRPO	51.3	67.2	46.7	55.1$_{\Delta_{\text{base}}}$	47.7	42.6	23.2	53.6	41.8$_{\Delta_{\text{base}}}$
	+ GSPO	50.7	67.4	47.1	55.1$_{+0.0\%}$	50.1	50.2	23.8	52.8	44.2$_{+5.7\%}$
	+ Tree-GRPO	51.7	68.1	47.3	55.7$_{+1.1\%}$	50.2	50.5	25.9	54.4	45.3$_{+8.4\%}$
	+ UMCTS-GRPO	**52.6**	**68.7**	**47.9**	**56.4**$_{+2.4\%}$	**52.5**	**52.5**	**27.2**	**56.8**	**47.2**$_{+13.0\%}$

Table 2: ALFWorld success rates (%) with Qwen2.5-Instruct backbones under the GiGPO evaluation protocol (Feng et al., 2026): the step-150 checkpoint is evaluated for 8 passes (128 episodes each) and results are averaged over 3 seeds (mean $\pm$ std). All is the episode-weighted overall score.

	Method	ALFWorld						
		Pick	Look	Clean	Heat	Cool	Pick2	All
Qwen2.5-1.5b	ReAct	17.4	20.5	15.7	6.2	7.7	2.0	12.8
	+ GRPO	85.3	53.7	84.5	78.2	59.7	53.5	72.8 $\pm$ 3.6
	+ GiGPO	94.4	67.5	**94.8**	**94.4**	79.8	76.4	86.7 $\pm$ 1.7
	+ Tree-GRPO	88.6	41.7	76.0	80.0	70.0	61.9	73.4 $\pm$ 3.3
	+ UMCTS-GRPO	**94.9**	**76.1**	90.1	86.0	**90.3**	**79.4**	**88.0** $\pm$ 0.8
Qwen2.5-7b	ReAct	48.5	35.4	34.3	13.2	18.2	17.6	31.2
	+ GRPO	90.8	66.1	89.3	74.7	72.5	64.7	77.6 $\pm$ 5.2
	+ GiGPO	97.7	82.7	**98.8**	83.7	**89.3**	79.2	90.8 $\pm$ 1.3
	+ Tree-GRPO	87.4	63.6	84.6	69.0	74.1	64.1	76.5 $\pm$ 2.5
	+ UMCTS-GRPO	**99.7**	**90.3**	**98.8**	**95.7**	87.2	**90.0**	**94.2** $\pm$ 1.9

multi-hop QA, where each intermediate query shapes the evidence available to the next, UMCTS-GRPO improves the macro-average over Tree-GRPO at every Qwen2.5 scale. At 14B, UMCTS-GRPO improves all four multi-hop benchmarks and raises the average from 45.3 to 47.2, and at 1.5B, Bamboogle rises from 13.6 to 18.4. These results support our central hypothesis: allocating search toward unresolved branches and learning more strongly from resolved comparisons improves credit assignment from terminal rewards alone.

ALFWorld. Table 2 extends the evaluation to state-changing interaction. UMCTS-GRPO achieves the highest overall success rate on both backbones, reaching 88.0 at 1.5B and 94.2 at 7B,

Table 3: Rollout-budget study on Qwen2.5-3B (EM, %). Budget is the rollout cost per prompt in complete-trajectory equivalents: each of the M trees performs one full rollout and $B_{\text{sim}} = LN_{\text{sim}}$ continuations, each counted as half a trajectory, giving $M(1 + B_{\text{sim}}/2)$. Δ_{rel} is relative to chain-based GRPO at the same budget, and the best entry per column and budget is in **bold**.

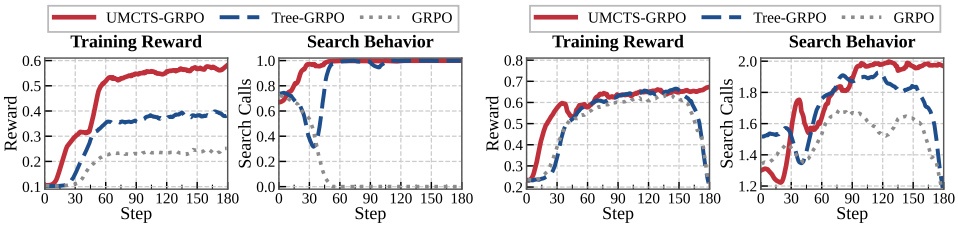
Method	Single-Hop QA				Multi-Hop QA				
	NQ	Trivia	PopQA	Avg./$\Delta^{\%}_{\text{rel}}$	Hotpot	2wiki	Musiq	Bamb	Avg./$\Delta^{\%}_{\text{rel}}$
Rollout token/tool budget ≈ 2 complete trajectories per prompt									
Chain-based (GRPO)	42.0	56.7	40.8	46.5$_{\Delta_{\text{base}}}$	17.9	25.6	3.3	12.8	14.9$_{\Delta_{\text{base}}}$
Tree-GRPO (M=1, N_{sim}=2, L=1)	46.1	59.4	43.6	49.7$_{+6.9\%}$	39.5	**40.2**	13.7	32.8	31.6$_{+112.1\%}$
UMCTS-GRPO (M=1, N_{sim}=2, L=1)	**46.6**	**61.8**	**44.2**	**50.9**$_{+9.5\%}$	**39.6**	38.2	**14.7**	**34.4**	**31.7**$_{+112.8\%}$
Rollout token/tool budget ≈ 4 complete trajectories per prompt									
Chain-based (GRPO)	44.4	58.0	42.0	48.1$_{\Delta_{\text{base}}}$	39.0	36.3	15.2	36.8	31.8$_{\Delta_{\text{base}}}$
Tree-GRPO (M=2, N_{sim}=2, L=1)	**46.8**	59.7	**43.6**	50.0$_{+4.0\%}$	42.4	**43.7**	17.8	43.2	36.8$_{+15.7\%}$
UMCTS-GRPO (M=2, N_{sim}=2, L=1)	45.8	**61.7**	**43.6**	**50.4**$_{+4.8\%}$	**46.8**	**43.7**	**18.4**	42.4	**37.8**$_{+18.9\%}$
Rollout token/tool budget ≈ 8 complete trajectories per prompt									
Chain-based (GRPO)	46.5	59.2	44.3	50.0$_{\Delta_{\text{base}}}$	39.4	36.4	16.1	33.6	31.4$_{\Delta_{\text{base}}}$
Tree-GRPO (M=4, N_{sim}=2, L=1)	47.6	60.8	44.2	50.8$_{+1.6\%}$	42.0	**42.9**	**19.5**	36.0	**35.1**$_{+11.8\%}$
UMCTS-GRPO (M=4, N_{sim}=2, L=1)	**49.4**	**62.3**	**45.2**	**52.3**$_{+4.6\%}$	**42.3**	39.4	17.6	**36.8**	34.0$_{+8.3\%}$

compared with 86.7 and 90.8 for GiGPO, and attains the lowest variance across seeds at 1.5B. The contrast with Tree-GRPO is especially clear: with random expansion, Tree-GRPO brings little benefit over GRPO in this setting, whereas UMCTS-GRPO lifts overall success over Tree-GRPO by 14.6 and 17.7 points at 1.5B and 7B. The gains are particularly pronounced on several task categories: at 7B, Heat improves from 83.7 to 95.7 and Pick2 from 79.2 to 90.0 relative to GiGPO; at 1.5B, Look improves from 67.5 to 76.1 and Cool from 79.8 to 90.3. Whereas GiGPO groups recurring states across independently sampled trajectories, UMCTS-GRPO actively constructs sibling comparisons at shared prefixes and allocates additional rollouts to unresolved branches, extending its benefits from retrieval-based QA to state-changing agent interaction.

Rollout budget. Table 3 varies the rollout budget over 2, 4, and 8 complete-trajectory equivalents per prompt on Qwen2.5-3B. Each of the M trees performs one full rollout followed by $B_{\text{sim}} = LN_{\text{sim}}$ simulations, organized as L rounds of N_{sim}; since each simulation resumes from an intermediate prefix, it is counted as half a trajectory, giving a budget of $M(1 + B_{\text{sim}}/2)$. We set $N_{\text{sim}} = 2$ and $L = 1$ ($B_{\text{sim}} = 2$), retain $K_{\text{ret}} = 3$ leaves per tree, and grow $M = 1, 2, 4$; Tree-GRPO uses the same tree configuration, and chain-based GRPO samples the corresponding number of complete trajectories. UMCTS-GRPO is the strongest single-hop method at every budget. At the tightest budget, it already surpasses chain-based GRPO trained with four times the budget on both single-hop (50.9 vs. 50.0) and multi-hop QA (31.7 vs. 31.4), and its single-hop margin over Tree-GRPO widens to 1.5 points at budget 8.

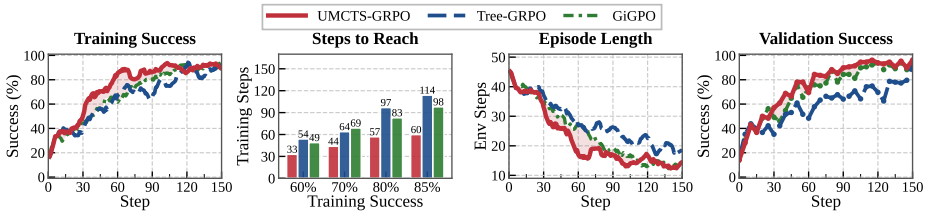
Training dynamics. Figure 4(a–b) tracks rollout reward and search calls per trajectory during multi-hop training. At 1.5B, UMCTS-GRPO settles at about one search per trajectory within the first 30 steps and reaches a final rollout reward of 0.57, compared with 0.38 for Tree-GRPO and 0.24 for GRPO. At 7B, it stabilizes around two searches per trajectory and remains stable through step 180, whereas the baseline runs deteriorate late in training. This behavior is consistent with the search mechanism: by concentrating rollouts on unresolved decision points, such as whether to issue a further query, the tree exposes contrasting outcomes early, and confidence-weighted credit turns them into a stable learning signal. Figure 4(c) compares training dynamics with GiGPO on Qwen2.5-7B-Instruct, with both methods run in the official verl-agent environment under the same evaluation protocol. UMCTS-GRPO learns faster: it reaches every training-success level earlier, attaining 85% success in 60 steps versus 98 for GiGPO, and shortens episodes earlier in training.

Ablations. UMCTS-GRPO uses uncertainty in two complementary roles: guiding tree search and refining step-level credit. Table 4 isolates their contributions on Qwen2.5-1.5B multi-hop QA under the same rollout budget and training configuration. Search plays the dominant role: replacing adaptive selection with random branching reduces average EM from 20.5 to 11.0, and removing only the uncertainty bonus ($c_u = 0$) lowers it to 12.2. Confidence-weighted credit provides consistent further gains: removing local credit ($\lambda_l = 0$) or uncertainty weighting ($w = 1, \tau = 0$) lowers the



(a) Qwen2.5-1.5B on multi-hop QA

(b) Qwen2.5-7B on multi-hop QA



(c) Qwen2.5-7B on ALFWorld

Figure 4: Training dynamics across multi-hop QA and ALFWorld. (a–b) On multi-hop QA, UMCTS-GRPO learns high-reward search behavior earlier and sustains it more consistently throughout training. (c) On ALFWorld with Qwen2.5-7B-Instruct, we show training and validation success, the training steps needed to first reach each success level, and episode length; shaded regions indicate where UMCTS-GRPO is ahead.

Table 4: Ablations on Qwen2.5-1.5B multi-hop QA (EM, %). Subscripts give the relative change from the full model.

Variant	Hotpot	2Wiki	MuSiQue	Bamb.	Avg.
UMCTS-GRPO	29.5	26.5	7.7	18.4	**20.5**
Search ablations					
w/o Adaptive Search (random)	14.7	21.7	2.1	5.6	$11.0_{-46.3\%}$
w/o Uncertainty Bonus ($c_u = 0$)	14.9	27.7	2.1	4.0	$12.2_{-40.5\%}$
Credit ablations					
w/o Local Credit ($\lambda_l = 0$)	29.3	26.4	7.0	14.4	$19.3_{-5.9\%}$
w/o Uncertainty Weighting ($w = 1, \tau = 0$)	28.7	27.4	6.4	15.2	$19.4_{-5.4\%}$
w/o Outcome Credit ($\lambda_o = 0$)	†collapse	collapse	collapse	collapse	-
Baselines					
Tree-GRPO	29.5	26.8	6.6	13.6	$19.1_{-6.8\%}$
Chain-based GRPO	14.6	24.4	2.2	4.0	$11.3_{-44.9\%}$

† Training collapses without the outcome term.

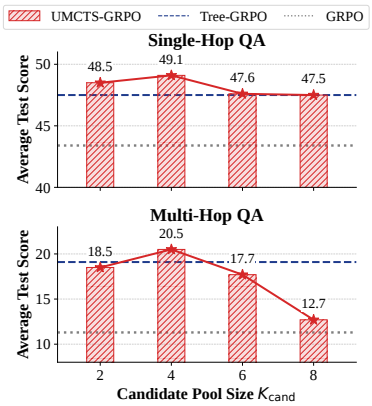


Figure 5: Effect of candidate pool size K_{cand} on Qwen2.5-1.5B.

average to 19.3 and 19.4, with the largest effect on Bamboogle (18.4 versus 14.4 and 15.2). The two components thus divide the labor: uncertainty-guided search determines where branch evidence is collected, while confidence-weighted credit determines how strongly that evidence shapes learning. Figure 5 varies the candidate-pool size under a fixed rollout budget. Single-hop QA is robust to K_{cand}, with all settings scoring between 47.5 and 49.1, at or above Tree-GRPO. On multi-hop QA, performance peaks at $K_{\text{cand}} = 4$, which balances the breadth of alternatives at each branching point against the depth the fixed budget can reach. We use $K_{\text{cand}} = 4$ in all experiments.

6 Conclusion

We introduced UMCTS-GRPO, which uses the same uncertainty signal to guide tree-search exploration and weight process-level credit. Backed-up uncertainty directs rollouts toward unresolved branches and calibrates sibling comparisons during optimization, yielding step-level supervision from terminal rewards alone. Across retrieval-based QA and ALFWorld, UMCTS-GRPO consistently improves over strong tree- and trajectory-based baselines, while ablations show that adaptive search provides the main gain and confidence-weighted local credit further improves learning.

AI USE STATEMENT

In this work, we used generative AI tools, including OpenAI's ChatGPT, for feedback on research methodology and experimental design, literature search and summarization, interpretation and presentation of experimental results, and manuscript editing, including grammar, clarity, and alternative phrasings. Generative AI tools were not used to conduct experiments or produce experimental results. All methodological decisions, implementations, experimental results, analyses, and claims were reviewed and verified by the authors. We take full responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide the complete UMCTS-GRPO training algorithm and detailed implementation of uncertainty backup, candidate construction, tree search, retention, and advantage estimation in the appendix. Experimental settings, rollout budgets, evaluation protocols, and hyperparameters are provided in the main paper and supplementary material. We additionally release an anonymized implementation, including the code and configurations for training and evaluation, at `https://anonymous.4open.science/r/UMCTS-GRPO-CCD5/README.md`.

REFERENCES

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. *Advances in Neural Information Processing Systems*, 38:46375–46408, 2026.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Yiran Guo, Lijie Xu, Jie Liu, Dan Ye, and Shuang Qiu. Segment policy optimization: Effective segment-level credit assignment in rl for large language models. *Advances in Neural Information Processing Systems*, 38:114399–114431, 2026.

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

Zhenyu Hou, Ziniu Hu, Yujiang Li, Rui Lu, Jie Tang, and Yuxiao Dong. Treerl: Llm reinforcement learning with on-policy tree search. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12355–12369, 2025.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

Yuxiang Ji, Ziyu Ma, Yong Wang, Guanhua Chen, Xiangxiang Chu, and Liaoni Wu. Tree search for llm agent reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 87362–87388, 2026.

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

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, 2017.

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

Jing Yu Koh, Stephen McAleer, Daniel Fried, and Ruslan Salakhutdinov. Tree search for language model agents, 2026. URL https://arxiv.org/abs/2407.01476.

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

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

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

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

Yaniv Oren, Viliam Vadocz, Matthijs TJ Spaan, and Wendelin Böhmer. Epistemic monte carlo tree search. In *International Conference on Learning Representations*, volume 2025, pp. 75835–75861, 2025.

Brendan O'Donoghue, Ian Osband, Remi Munos, and Volodymyr Mnih. The uncertainty bellman equation and exploration. In *International conference on machine learning*, pp. 3836–3845, 2018.

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

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

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

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

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

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

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, 2024.

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

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

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Zijing Zhang, Ziyang Chen, Mingxiao Li, Zhaopeng Tu, and Xiaolong Li. Rlvmr: Reinforcement learning with verifiable meta-reasoning rewards for robust long-horizon agents. In *International Conference on Learning Representations*, volume 2026, pp. 114676–114700, 2026.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization, 2025. URL https://arxiv.org/abs/2507.18071.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning acting and planning in language models. *arXiv preprint arXiv:2310.04406*, 2023.

A NOTATION

Table 5 summarizes the notation used in the main text and the appendix.

Symbol	Meaning
Agent interaction (Section 3)	
x	Prompt (question or task instruction)
$s_t = (x, H_{<t})$	State before step t: the prompt and the interaction history so far
a_t	Agent step (thought followed by action)
$o_t,\ P_{\mathrm{env}}$	Observation returned by the environment and its transition distribution
$\oplus$	Appending a step and its observation to the history
$H,\ T$	Trajectory $(a_0, o_0, \ldots, a_{T-1}, o_{T-1})$ and its number of steps
$R(H)$	Outcome reward of trajectory H
$\pi_\theta,\ \pi_{\mathrm{old}},\ \pi_{\mathrm{ref}}$	Current policy, frozen behavior policy used for collection, and reference policy
Group-relative outcome advantage (Section 3)	
$\mathcal{G}_x$	Group of trajectories for prompt x, collected from all M trees
$\mu_x,\ \sigma_x$	Mean and standard deviation of rewards in $\mathcal{G}_x$
$A^{\mathrm{out}}(H)$	Prompt-level outcome advantage, Equation (2)
ϵ	Small positive constant for numerical stability
Tree construction and budget (Section 4.1, Table 3)	
M	Number of independent trees per prompt
$L,\ N_{\mathrm{sim}}$	Rounds and simulations per round; each tree performs $B_{\mathrm{sim}} = LN_{\mathrm{sim}}$ simulations after its initial rollout, for a budget of $M(1 + B_{\mathrm{sim}}/2)$
K_{cand}	Number of next-step samples in a candidate pool
K_{ret}	Number of leaves retained per tree for training
$\mathcal{C}_b,\ a_b$	Candidate cluster and its representative action (Appendix E.2)
$(s, a),\ s'$	Edge from prefix s via action a, and the resulting child prefix
Edge and node statistics (Section 4.1)	
$Q(s, a)$	Return estimate of an edge, Equation (5)
$U(s, a)$	Uncertainty statistic of an edge, Equation (5)
$P(s, a)$	Prior of an edge from cluster frequency; $\bar{P}_+,\ \bar{P}_*$ are renormalized versions
$N(s),\ N(s, a)$	Number of completed continuations through a node and through an edge
$z_i,\ u_i$	Return and uncertainty targets of the ith backup, Equation (4)
$n_Q,\ s_z^2$	Number of backups incorporated into an edge and the sample variance of their return targets
$r(s, a),\ \gamma$	Immediate reward of an edge and discount factor
$V(s),\ U_V(s)$	Node value and node uncertainty, Equations (18)–(19)
$\mathcal{A}_+(s)$	Children of s that have received at least one backup
$\mathcal{I}(s, a),\ \bar{\ell}(s, a)$	Generated tokens of an action step and their mean log probability under π_{old}
$u_{\mathrm{local}}(s, a)$	Local uncertainty input, the clipped mean surprisal $\mathrm{clip}(-\bar{\ell}, 0, 10)$
$C(s, a),\ \eta$	Optional cost of an edge and its penalty weight ($\eta = 0$ in all experiments)
Selection (Section 4.1)	
$S(s, a)$	Selection score, Equation (3)
$c_u,\ c_p$	Coefficients of the uncertainty bonus and the PUCT term
Process advantages (Sections 4.2–4.3)	
$\mathcal{A}(s)$	Children of s used for local comparison (Appendix E.3)
$w(s, a),\ b(s)$	Inverse-uncertainty weight and uncertainty-weighted sibling baseline, Equation (6)
$g(s, a),\ \tau$	Confidence gate and its attenuation strength, Equation (7)
$D(s),\ \bar{Q}(s)$	Population variance and mean of the sibling values
$A^{\mathrm{loc}}(s, a)$	Confidence-weighted local advantage, Equation (7)
Z_a	Standardized sibling comparison before gating (Appendix C)
$\alpha^{\pm}$	Confidence-dependent preference weights of a sibling pair, Proposition 2
$\lambda_o,\ \lambda_l$	Weights of the outcome and local advantages
$A_{i,t}$	Combined advantage of step t of trajectory H_i, Equation (8)
Policy optimization (Section 4.2)	
$\rho_{i,\ell}(\theta)$	Token-level probability ratio between π_θ and π_{old}
$t(i, \ell)$	Agent step containing token ℓ of trajectory i
$\mathcal{B}_x$	Batch of trajectories produced by the search for prompt x
$\delta,\ c_{\mathrm{dual}}$	Clipping threshold and dual-clip constant
$\beta,\ K_{\mathrm{ref}}$	Weight of the reference KL penalty and the penalty itself
$\mathcal{J}(\theta)$	Training objective, Equation (9)

Table 5: Notation.

B ALGORITHM WORKFLOW

Algorithm 1 gives one training iteration. The behavior policy is frozen throughout collection, and each tree incorporates a completed continuation before selecting the next prefix; batching across prompts and trees preserves this within-tree ordering. Candidate-pool handling, including the single top-up of an exhausted pool, follows Appendix E.2.

Algorithm 1 UMCTS-GRPO: one training iteration

Require: policy π_θ, reference π_{ref}, prompt batch $\mathcal{X}$; trees M, simulations $B_{\text{sim}} = LN_{\text{sim}}$, candidates K_{cand}, retained leaves K_{ret}; coefficients $c_u, c_p, \gamma, \tau, \lambda_o, \lambda_l$

$\pi_{\text{old}} \leftarrow \pi_\theta$; $\mathcal{B} \leftarrow \emptyset$
for $x \in \mathcal{X}$ **do**
  for $m = 1, \dots, M$ **do**
    $\mathcal{T}_m \leftarrow$ root $s_0 = (x, \emptyset)$ with pool $\textsc{Pool}(s_0)$ {prior P from cluster frequency}
    for $j = 0, \dots, B_{\text{sim}}$ **do**
      $s \leftarrow s_0$ **if** $j = 0$ **else** $\textsc{Select}(\mathcal{T}_m)$ {Eq. (3)}
      $a \leftarrow$ most frequent untried representative in $\textsc{Pool}(s)$ {create pool if absent}
      $H \leftarrow \textsc{Rollout}(\pi_{\text{old}},\ s \oplus a)$; observe $R(H)$
      $\textsc{Backup}(\mathcal{T}_m, H)$ {update Q, U, N along the path}
    end for
    $\mathcal{L}_m \leftarrow$ top-K_{ret} leaves of $\mathcal{T}_m$ by incoming-edge score; keep Q, U
  end for
  $A^{\text{out}} \leftarrow$ Eq. (2) over $\bigcup_m \mathcal{L}_m$
  $A^{\text{loc}} \leftarrow$ Eqs. (6)–(7) over siblings in $\bigcup_m \mathcal{L}_m$
  $A_{i,t} \leftarrow \lambda_o A^{\text{out}} + \lambda_l A^{\text{loc}}$ {Eq. (8)}
  $\mathcal{B} \leftarrow \mathcal{B} \cup \{(H_i,\ \log \pi_{\text{old}},\ A_{i,\cdot},\ \text{masks})\}$
end for
Update π_θ on $\mathcal{B}$ by maximizing Eq. (9)

The search resumes generation from a stored interaction prefix. In retrieval-based QA, the stored history supplies the full policy context, so subsequent queries can be issued independently; in a state-changing environment such as ALFWorld, we restore the corresponding environment state before executing a new branch.

C Proofs of Propositions

Throughout, fix a sibling set $\mathcal{A}(s)$ with $|\mathcal{A}(s)| \geq 2$, $\epsilon > 0$, $\tau \geq 0$, and $U(s,a) \geq 0$. We write the standardized comparison as

$$Z_a = \frac{Q(s,a) - b(s)}{\sqrt{D(s) + \epsilon}}, \qquad \text{so that} \qquad A^{\text{loc}}(s,a) = g(s,a)\, Z_a \tag{10}$$

by Equation (7). All statements concern the local advantage before it is combined with the outcome term.

Proof of Proposition 1. *(i) Sign preservation.* Since $U(s,a) \geq 0$ and $\tau \geq 0$, the gate satisfies $0 < g(s,a) \leq 1$, so multiplying Z_a by it preserves the sign. For fixed Q, b, and D (recall that $b(s)$ depends on the sibling uncertainties through Equation (6)), the comparison Z_a is fixed while $g(s,a)$ is non-increasing in $U(s,a)$; hence

$$|A^{\text{loc}}(s,a)| = g(s,a)\,|Z_a| \leq |Z_a|, \tag{11}$$

with strict decrease in $U(s,a)$ when $\tau > 0$ and $Z_a \neq 0$.

(ii) No spurious preference. All weights $w(s,a)$ are positive, so if $Q(s,a) = q$ for every sibling, then $b(s) = q$ and every numerator $Q(s,a) - b(s)$ vanishes.

(iii) Consistency. Equal uncertainties give equal weights and a common gate, so $b(s) = \bar{Q}(s)$ and $g(s,a) = 1/(1 + \tau u)$. Substituting into Equation (7) yields the stated expression, which for $\tau = 0$ is the standardized sibling advantage. □

Weighted baseline and gating. The baseline is the confidence-weighted center of the sibling values:

$$\sum_{a \in \mathcal{A}(s)} w(s,a)\big(Q(s,a) - b(s)\big) = 0. \tag{12}$$

After gating, the local advantages are in general no longer centered: resolved siblings retain more of their deviation than unresolved ones, so the update is dominated by the comparisons the tree has settled most clearly.

Proposition 2 (Confidence-weighted step-level preference)**.** *Let s have two children a^+, a^- with $Q(s,a^+) > Q(s,a^-)$, each appearing in one trajectory of $\mathcal{B}_x$, and neglect ϵ in the denominator of Equation* (7)*. At $\rho = 1$, the local term of Equation* (8) *contributes, up to a positive normalization constant, the policy gradient*

$$\lambda_l \Big(\alpha^+ \nabla_\theta \log \pi_\theta(a^+ \mid s) - \alpha^- \nabla_\theta \log \pi_\theta(a^- \mid s) \Big), \qquad \alpha^\pm = \frac{2\, g(s,a^\pm)\, w(s,a^\mp)}{w(s,a^+) + w(s,a^-)} > 0. \quad (13)$$

If $U(s,a^+) = U(s,a^-) = u$, then $\alpha^+ = \alpha^- = 1/(1+\tau u)$, and the update is the gradient of the step-level preference $\log \pi_\theta(a^+ \mid s) - \log \pi_\theta(a^- \mid s)$ scaled by confidence.

Proof of Proposition 2. Let $\Delta = Q(s,a^+) - Q(s,a^-) > 0$ and $w^\pm = w(s,a^\pm)$. Equation (6) gives

$$Q(s,a^+) - b(s) = \frac{w^-\Delta}{w^+ + w^-}, \qquad Q(s,a^-) - b(s) = -\frac{w^+\Delta}{w^+ + w^-}. \quad (14)$$

With two siblings, $D(s) = \Delta^2/4$, so neglecting ϵ yields

$$A^{\mathrm{loc}}(s,a^\pm) = \pm \frac{2\, g(s,a^\pm)\, w^\mp}{w^+ + w^-} = \pm\, \alpha^\pm. \quad (15)$$

At $\rho = 1$, both the clip and the dual-clip guard are inactive, so the surrogate in Equation (9) has gradient $\sum A \nabla_\theta \log \pi_\theta$ over generated tokens, up to the token-averaging constant. The tokens of each step share one advantage and their log probabilities sum to $\log \pi_\theta(a^\pm \mid s)$, which gives the stated expression. Equal uncertainties give $w^+ = w^-$ and $g(s,a^+) = g(s,a^-) = 1/(1+\tau u)$, hence $\alpha^+ = \alpha^- = 1/(1+\tau u)$. □

D Experimental Details

D.1 Retrieval-based QA

Data and retrieval. Following Ji et al. (2026), single-hop models are trained on the NQ training set (79,168 questions) and multi-hop models on the HotpotQA training set (90,447 questions); the other five benchmarks are never seen during training. Evaluation uses the full test or development split of all seven benchmarks as released by Search-R1 (Jin et al., 2025), including 22,523 multi-hop questions (2WikiMultiHopQA 12,576, HotpotQA 7,405, MuSiQue 2,417, Bamboogle 125). The retriever is E5-base-v2 (Wang et al., 2022) over the 2018 Wikipedia dump used by Search-R1, and each search call returns the top-3 passages. The corpus, index, prompt template, and interaction format are identical for all methods.

Interaction and reward. The agent interleaves reasoning with `<search>` calls and ends with an `<answer>` tag; retrieved passages are inserted as observations inside `<information>` tags and masked from the policy loss. An episode allows at most 3 tool calls, each turn generates at most 500 tokens, retrieved observations are truncated to 500 tokens, and the full context is capped at 4,096 tokens. The reward follows the Search-R1 recipe used by Ji et al. (2026): exact match of the extracted answer after standard normalization (1 if correct), with a partial format score for well-formed but incorrect trajectories. The reward is binary at evaluation time: all reported numbers are exact match only.

Optimization. All RL methods share the hyperparameters in Table 6, which follow Ji et al. (2026). Training runs for 180 steps with 512 prompts per step, AdamW with learning rate 10^{-6}, warmup ratio 0.285, gradient clipping 1.0, one PPO epoch per batch, token-mean loss aggregation, entropy coefficient 10^{-3}, and the low-variance (K3) KL estimator toward the initial policy with $\beta = 10^{-3}$. Rollouts are sampled with temperature 1.0 and top-p 0.95 using vLLM in bfloat16. Qwen2.5 models are the base (non-instruct) checkpoints.

D.2 ALFWORLD

We use the verl-agent implementation and prompt of Feng et al. (2026) without modification of the environment, the observation format, or the action interface. The agent sees the task description and the two most recent observation–action pairs, may take at most 50 environment steps, and receives reward 10 for success and 0 otherwise, with a penalty of 0.1 for invalid actions. Each training step samples 16 tasks from the ALFWorld training split; the rollout budget matches the group size of 8 used by GiGPO. Training runs for 150 steps with learning rate 10^{-6}, KL coefficient 0.01 (K3), entropy coefficient 10^{-3}, discount $\gamma = 0.95$, mini-batch size 256, maximum prompt length 2,048, and maximum response length 512 tokens per step. Rollouts use temperature 1.0; evaluation uses temperature 0.4 on the 128 in-distribution evaluation games, and the step-150 checkpoint is evaluated for 8 passes.

D.3 UMCTS-GRPO CONFIGURATION

Table 7 lists the search and credit hyperparameters. They are shared across backbones and were not tuned per model or benchmark; the only swept parameter is K_{cand} (Figure 5). Candidate clustering embeds the normalized action text with all-mpnet-base-v2 and merges candidates of the same action type whose cosine similarity is at least 0.8. For ALFWorld, whose actions are drawn from a small admissible set, candidates are merged by exact match, each branching point keeps at most two representatives, each tree runs 8 simulations, and the trees of a task share their search statistics so that later simulations can branch from prefixes discovered by either tree.

Configuration	**QA (single- and multi-hop)**	**ALFWorld**
Optimizer / learning rate	AdamW / 1×10^{-6}	AdamW / 1×10^{-6}
Warmup ratio	0.285	0
KL estimator / coefficient β	K3 / 0.001	K3 / 0.01
Entropy coefficient	0.001	0.001
Training steps	180	150
Prompts per step	512	16
PPO mini-batch size	64	256
PPO epochs / gradient clip	1 / 1.0	1 / 1.0
Clip threshold δ / dual-clip c_{dual}	0.2 / 3	0.2 / 3
Tokens per turn / max context	500 / 4,096	512 / 2,048 (prompt)
Max observation length	500	–
Max tool calls / environment steps	3	50
Rollout temperature / top-p	1.0 / 0.95	1.0 / 1.0
Evaluation decoding	greedy	temperature 0.4
Discount γ	1.0	0.95
Reward	EM + format score	success (10) $-$ 0.1 invalid
Retriever / top-k passages	E5 / 3	–

Table 6: Training hyperparameters shared by all RL methods within a task family.

Symbol	**Meaning**	**Value**
M	Trees per prompt	2
$N_{\mathrm{sim}},\ L$	Tree-shape parameters	2, 1
K_{cand}	Candidate pool size	4
K_{ret}	Retained leaves per tree	3
$c_u,\ c_p$	Uncertainty-bonus and PUCT coefficients	1.0, 1.0
γ	Discount in the backup	1.0
τ	Confidence-gate attenuation	$\{1.0, 3.0\}$
$\lambda_o,\ \lambda_l$	Outcome and local advantage weights	1.0, 1.0
η	Cost penalty weight	0
–	Candidate merging	all-mpnet-base-v2, cosine ≥ 0.8

Table 7: UMCTS-GRPO hyperparameters for retrieval-based QA. The same values are used for every backbone.

Rollout budget accounting. All tree methods are compared at matched budgets in the sense of Ji et al. (2026): the number of complete-trajectory equivalents per prompt, $M(1 + LN_{\mathrm{sim}}/2)$, which is 4 in the default configuration. UMCTS-GRPO additionally samples K_{cand} next-step candidates only at prefixes that the search selects for branching; these are single-step generations, not full

trajectories, and candidate clustering reuses the tokens of the chosen representative, so no candidate is generated twice.

Compute. All experiments were run on a cluster of NVIDIA GH200 nodes with four GPUs each, and H200 nodes with 8 GPUs each. A single training run takes 8–30 hours for the 1.5B and 3B models and up to two days for the larger models.

E Additional Method Details

E.1 Return and uncertainty backup

Local input. For the generated tokens $\mathcal{I}(s,a)$ of an action step, define the mean sampled log probability

$$\bar{\ell}(s,a) = \frac{1}{|\mathcal{I}(s,a)|} \sum_{k \in \mathcal{I}(s,a)} \log \pi_{\text{old}}(y_k \mid s, y_{<k}), \qquad u_{\text{local}}(s,a) = \text{clip}\big(-\bar{\ell}(s,a), 0, 10\big). \tag{16}$$

The mean excludes padding. This surprisal of the sampled tokens requires only the log probabilities already computed during generation, so it adds no decoding cost. Nonfinite inputs use a fallback of 1. On the final edge of a completed rollout, the deterministic-evaluator convention sets this local input to zero and initializes the downstream pair to $(R(H), 0)$, so the terminal reward is supplied exactly once through this boundary value.

Edge statistics. Each backup contributes a target pair (z_i, u_i) from Equation (4). For n_Q incorporated pairs, let

$$\bar{z} = \frac{1}{n_Q} \sum_{i=1}^{n_Q} z_i, \qquad \bar{u} = \frac{1}{n_Q} \sum_{i=1}^{n_Q} u_i, \qquad s_z^2 = \begin{cases} \dfrac{1}{n_Q - 1} \displaystyle\sum_{i=1}^{n_Q} (z_i - \bar{z})^2, & n_Q > 1, \\ 0, & n_Q = 1. \end{cases} \tag{17}$$

The edge stores $Q = \bar{z}$ and $U = \bar{u} + s_z^2$, maintained by online mean and second-moment updates. Because s_z^2 measures the disagreement among backup targets directly, a branch whose continuations genuinely conflict remains uncertain however often it is visited, while a branch whose continuations agree is driven toward resolution.

Node aggregation. Let $\mathcal{A}_+(s)$ contain only children that have received at least one backup, and let $\bar{P}_+(s,a)$ be their normalized priors. After updating a child edge, recompute

$$V(s) = \sum_{a \in \mathcal{A}_+(s)} \bar{P}_+(s,a) Q(s,a), \tag{18}$$

$$U_V(s) = \sum_{a \in \mathcal{A}_+(s)} \bar{P}_+(s,a)^2 U(s,a) + \sum_{a \in \mathcal{A}_+(s)} \bar{P}_+(s,a) \big(Q(s,a) - V(s)\big)^2. \tag{19}$$

The first uncertainty term propagates child statistics; the second records disagreement across child values. The updated $(V(s), U_V(s))$ supplies the next upstream target. Excluding unbacked children prevents their default zeros from being treated as observed returns. For a nonterminal node without backed-up children, the implementation falls back to its incoming-edge value and zero node uncertainty; the evaluated terminal boundary is handled as above. Because every target is refreshed as the tree grows, U tracks the current state of the search and directly ranks how settled each branch is.

E.2 Candidate pools and action priors

Finite action pools. Each pool is formed by sampling next-step generations, not complete trajectories. Candidates are parsed into an action type and normalized action text. Exact matching merges identical normalized actions; embedding-based matching merges candidates of the same type when their similarity exceeds the configured threshold. Unparseable outputs retain a separate raw-text type.

A cluster $\mathcal{C}_b$ stores its frequency and one representative generation a_b; the edge created from it receives the creation prior $P(s, a_b) = |\mathcal{C}_b| / \sum_j |\mathcal{C}_j|$. Expansion claims one highest-frequency untried representative and reuses its tokens, so each expansion costs a single continuation. Subsequent steps follow a single rollout continuation, and candidate pools at later prefixes are created only when search selects them for branching. A selected exhausted pool can receive one additional sampling pass over its lifetime; if no new representative is found, the attempt is skipped.

Prior normalization. An expanded candidate edge stores its cluster-frequency fraction at creation, and a later top-up does not rewrite previously stored edge priors. For any supplied sibling subset $\mathcal{A}_*$, priors are renormalized over that subset. If every edge has a stored cluster prior, these fractions are used. If any edge lacks one, as can occur for an ordinary continuation edge, the entire subset instead uses the length-normalized score

$$\bar{P}_*(s, a) = \frac{\exp\big(\bar{\ell}(s, a)\big)}{\sum_{b \in \mathcal{A}_*} \exp\big(\bar{\ell}(s, b)\big)}. \tag{20}$$

Cluster fractions and log-probability scores are never mixed within one normalization, and equal fallback scores give uniform priors. Selection uses existing siblings; node aggregation uses only backed-up siblings.

Counts and costs. The implementation supports an optional cost penalty, ranking children by $S(s, a) - \eta C(s, a)$ with S from Equation (3), where $C(s, a)$ is the number of generated action tokens plus the number of retrieved documents; all experiments use $\eta = 0$. For a collected tree, $N(s)$ is the number of completed leaves below s, clamped below by one, and $N(s, a)$ is the corresponding count for the child reached by a. Prefix sharing avoids regenerating and re-executing shared histories across branches.

E.3 Retention and advantage construction

Retained comparisons. Before pruning, leaves are ranked by Equation (3) applied to their incoming edge. Up to K_{ret} leaves are kept per tree together with their shared prefixes, while the stored Q and U continue to summarize the evidence of the full collection phase. The local estimator uses the children of each retained prefix. If fewer than K_{ret} leaves are available, sampling with replacement fills the training group; these copies affect training multiplicity but do not create new observations for backup.

Normalization and token assignment. The outcome group pools retained trajectories across all M trees for the same prompt, including any replacement copies, and its standard deviation uses the sample convention. For a singleton group, the implementation uses a zero baseline and unit scale. The local denominator instead uses the population variance

$$D(s) = \frac{1}{|\mathcal{A}(s)|} \sum_{a \in \mathcal{A}(s)} \big(Q(s, a) - \bar{Q}(s)\big)^2, \qquad \bar{Q}(s) = \frac{1}{|\mathcal{A}(s)|} \sum_{a \in \mathcal{A}(s)} Q(s, a). \tag{21}$$

The baseline and gate in Equations (6)–(7) are computed once from the frozen collection statistics. The resulting step advantage is assigned to generated thought and action tokens. Observations and padding are masked, and shared prefixes that appear in multiple retained trajectories are weighted according to that multiplicity.

Clipped implementation. Equation (9) presents the core policy surrogate. The token-level implementation clamps log ratios to $[-20, 20]$ before exponentiation and includes a dual-clip guard for negative advantages. Writing $\ell_{\text{clip}}(\rho, A) = \min(\rho A, \text{clip}(\rho, 1 - \delta, 1 + \delta)A)$, the guarded surrogate is

$$\ell_{\text{guard}}(\rho, A) = \begin{cases} \max\big(\ell_{\text{clip}}(\rho, A), c_{\text{dual}} A\big), & A < 0, \\ \ell_{\text{clip}}(\rho, A), & A \geq 0, \end{cases} \qquad c_{\text{dual}} > 1. \tag{22}$$

The actor also supports entropy regularization and the configured reference-policy KL penalty. These implementation terms leave the construction of tree-based advantages unchanged.

F Qualitative Examples

We compare test trajectories of UMCTS-GRPO and Tree-GRPO trained on the same backbone and evaluated at the same step (step 120). Retrieved passages are abridged to the decisive evidence; queries and answers are shown verbatim. Figures 6 and 7 show one multi-hop and one single-hop case.

Question: Where did the director of the film *The Shout* graduate from? 2WikiMultiHopQA
Gold: National Film School in Łódź

UMCTS-GRPO (ours) ✓

HOP 1 Identify the director

Search `director of film The Shout`

The Shout: "…a 1978 British horror film directed by **Jerzy Skolimowski** …"

HOP 2 Retrieve his education

Search `Jerzy Skolimowski university`

Jerzy Skolimowski: "…a graduate of the prestigious **National Film School in Łódź** …"

Answer **National Film School in Łódź** ✓

Tree-GRPO ✗

HOP 1 Identify the director

Search `Director of film The Shout`

The Shout: "…a 1978 British horror film directed by **Jerzy Skolimowski** …"

HOP 2 Retrieve his education

Search `University or school Jerzy Skolimowski graduated from`

***Henryk** Skolimowski*: "…received his Ph.D. in philosophy from **Oxford University** …"

Answer **University of Oxford** ✗

Figure 6: **Multi-hop case (Qwen2.5-3B).** Both agents resolve the bridge entity in the first hop. UMCTS-GRPO issues a focused second query and retrieves evidence about the director himself; Tree-GRPO's query drifts to a different Skolimowski, and the entity mismatch propagates to the final answer.

Question: Who was the producer of *9*? PopQA
Gold: Jim Lemley; Timur Bekmambetov

UMCTS-GRPO (ours) ✓

Search `who was the producer of 9`

9 (2009 animated film): "…produced by **Jim Lemley**, Tim Burton, **Timur Bekmambetov**, and Dana Ginsburg …"

Answer **Jim Lemley** ✓

Tree-GRPO ✗

Invalid `search`

Environment: action not wrapped in a valid tag; the model is asked to retry.

Invalid actions repeat until the turn budget is exhausted.

Answer *none* ✗

Figure 7: **Single-hop case (Qwen2.5-1.5B).** UMCTS-GRPO completes the search–answer loop with one well-formed query that retrieves the answer directly; Tree-GRPO never issues a valid search and exhausts its interaction budget without answering.

G Limitations and Future Work

Our experiments focus on retrieval-based QA and text-based interactive environments; extending UMCTS-GRPO to web browsing, code execution, and multimodal agents is a natural next step. In addition, the uncertainty statistic currently combines its three sources with fixed coefficients shared across tasks, and learning to adapt these coefficients or the candidate-pool size to each task could further improve the allocation of rollout budget.