REGPO: LEVERAGING REGRET-BASED FEEDBACK FOR AGENTIC POST-TRAINING WITH TREE-STRUCTURED SAMPLING

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce **REGPO**, a policy optimization method that uses *regret-based* feedback for step-level credit assignment in post-training language models (LMs) for agentic tasks. Specifically, motivated by the decomposition of regret for learning in Markov decision processes (MDPs) into action suboptimality gaps, **REGPO** develops sampled-search proxies for *regret-based advantages* at selected prefixes, leveraging tree-structured sampling. These per-step regret proxies are constructed from sampled rollout subtrees by comparing each action's estimated value with the highest estimate at the same prefix. We show that, in finite deterministic tree-structured MDPs, existing Tree-GRPO's return-based advantages can decrease the probability of selecting the optimal action during learning. This occurs when the action's value under the current policy is lower than that of a suboptimal action. Experiments on these MDPs then show how the different advantage constructions affect the learning dynamics of policy optimization, e.g., which actions remain likely to be sampled during training. An ALFWorld analysis further examines how action-value rankings relate to action selection, task-relevant intermediate-state reachability, and continuation execution in agentic settings. We also evaluate **REGPO** extensively on four agentic benchmarks spanning robotic manipulation, web interaction, browser control, and embodied reasoning. Under the benchmark-specific reporting protocol, the evaluated **REGPO** configurations attain the highest scores among the compared methods.

1 INTRODUCTION

Large language models (LLMs) have demonstrated strong performance in reasoning and interactive tasks (Wei et al., 2022; Shao et al., 2024; Yao et al., 2023). However, agentic tasks with long horizons remain challenging for LMs, because a trajectory return provides only a coarse learning signal for a long sequence of actions. When an agent fails after many tool calls or environment interactions, using a single trajectory outcome to construct the learning signal for every generated action, as in *group-based* post-training (Shao et al., 2024; Wei et al., 2025), does not identify which action first made task success unreachable. Conversely, a successful trajectory may contain an early action that should be reinforced even if the current policy rarely completes a successful continuation following that action. These issues raise the pressing question of proper *credit assignment* in post-training LM agents, especially with trajectory-based feedback.

In this paper, we focus on *step-level* credit assignment: assigning distinct credit to actions taken at different agent steps of a trajectory. In LM agentic environments, an agent step corresponds to one LM output, parsed into an action for execution in the environment. More importantly, *tree-structured* sampling is oftentimes available and exploited in such environments, where the agents can sample alternative actions from a shared prefix, and roll out their counterfactual continuations for data collection. For example, Tree-GRPO (Ji et al., 2026), a recent variant of GRPO, precisely leverages this structure in agentic post-training. For the comparison of relevance here, we focus on Tree-GRPO's intra-tree component. Tree-GRPO assigns its intra-tree relative advantage estimate to each complete trajectory. These estimates induce a comparison among actions at a fixed prefix, based on their sample mean continuation returns under the old policy.

However, Tree-GRPO's induced mean-return comparison can rank an action with a better achievable continuation below an alternative, when the old policy rarely selects the subsequent actions required

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.

to realize that higher return. The resulting update may thus decrease the probability of selecting that action, thereby reducing how often the training process reaches and learns at later prefixes (see Section 3.2 and Theorem 2 for formal justifications). To mitigate this issue, we take a *regret-based* perspective on credit assignment: an action is evaluated by how much it reduces the *best achievable return* from the current prefix. This motivates assigning step-level credit using action suboptimality gaps estimated from sampled rollout subtrees. More specifically, we introduce Regret-feedback Policy Optimization (**REGPO**). At each selected prefix, **REGPO** searches over sampled continuations and estimates the value of each sampled action. It compares each action's estimated value with the highest estimate at that prefix. These comparisons yield sampled-search proxies for the actions' regret-based advantages.

Indeed, the two feedback signals can induce different update directions for the policies, affecting how often the training process reaches later prefixes that may lead to higher returns. We analyze how these action comparisons affect the selection of actions with better achievable continuations, the probabilities of reaching later prefixes along those continuations, and the learning dynamics starting from those prefixes. We test these effects in two families of finite tree-structured Markov decision processes (MDPs): the *Hard-Safe tree* we newly construct, as a canonical model for agentic tasks (see Figure 1 (a)), and *random trees* (see Figure 1 (b)). In ALFWorld, we further examine whether action-value disagreement is accompanied by the aforementioned differences in action selection, task-relevant intermediate-state reachability, and continuation execution.

Under the benchmark-specific reporting protocol, across RoboTwin 2.0 (Beat Block Hammer) (Chen et al., 2026) for robotic manipulation, WebShop (Yao et al., 2022) for web interaction, MiniWoB++ (Liu et al., 2018) for browser control, and ALFWorld (Shridhar et al., 2021) for embodied reasoning, the evaluated **REGPO** configurations attain the highest scores compared to the existing methods. Relative to the strongest baseline without regret-based feedback, the margins are 14.8 and 6.2 points on Beat Block Hammer in-distribution (ID) and out-of-distribution (OOD) evaluation, 3.9 points on WebShop, 2.1 points on MiniWoB++, and 20.2 and 29.1 points on ALFWorld ID and OOD evaluations. Due to space constraints, we defer a detailed review of related work to Section C.

2 PRELIMINARIES

Problem setup. Agentic tasks often require multi-step, long-horizon interactions with an external system or environment, which can be modeled as *sequential decision-making* problems. Let p denote a task instance, including its instruction and, when needed, an environment seed or other fixed configuration, and let o_t be the observation before step t, with o_0 being the initial observation. Before agent step $t \geq 0$, let $\xi_t = (p, o_0, a_0, r_0^{\text{step}}, o_1, a_1, r_1^{\text{step}}, \ldots, o_t)$ be the complete task-conditioned interaction prefix. The LM receives $h_t = \text{Prompt}(\xi_t)$, a possibly truncated rendering of the task and interaction prefix. Conditioned on h_t, the LM output distribution $p_\theta^{\text{LM}}(\cdot \mid h_t)$ generates $y_t = (\ell_{t,i})_{i=1}^{L_t}$. We parse this output as $a_t = \text{parse}(y_t)$, treating malformed outputs as a designated invalid action in the action space. The LM output distribution and parser induce the parsed-action policy $\pi_{\theta,t}(a \mid \xi_t) := \mathbb{P}_{y \sim p_\theta^{\text{LM}}(\cdot \mid h_t)}[\text{parse}(y) = a]$. We write π_θ for the resulting sequence of step-indexed parsed-action policies. The action is then executed in the underlying environment. Let ρ denote the environment law, where the next observation and step reward are sampled from $(o_{t+1}, r_t^{\text{step}}) \sim \rho(\cdot \mid \xi_t, a_t)$. This reward may be sparse and mostly terminal, and may also include intermediate penalties such as invalid-action or formatting penalties. The interaction continues until termination or a maximum horizon. Let T_τ be the realized number of agent steps and $\tau := \xi_{T_\tau}$ be the trajectory. Its realized discounted return is $R^{\text{env}}(\tau) := \sum_{t=0}^{T_\tau - 1} \gamma^t r_t^{\text{step}}$, where $\gamma \in [0, 1]$.

Policy objective and advantage estimation. For a task instance p, let $G(p) = \{\tau_n\}_{n=1}^N$ be trajectories sampled from the old policy $\pi_{\theta_{\text{old}}}$. The methods below use the same clipped token-level surrogate form as in proximal policy optimization (PPO) (Schulman et al., 2017). With T_n steps in τ_n, $L_{n,t}$ tokens at step t, $Z_n = \sum_{t=0}^{T_n - 1} L_{n,t}$, and aggregation weights $\alpha_n > 0$, the objective is

$$\widehat{\mathcal{J}}_{G(p)}(\theta) = \frac{1}{\sum_{m=1}^N \alpha_m} \sum_{n=1}^N \frac{\alpha_n}{Z_n} \sum_{t,i} \left[\min\Big(\varrho_{n,t,i}(\theta) \widehat{A}_{n,t,i}, \text{clip}(\varrho_{n,t,i}(\theta), 1-\epsilon, 1+\epsilon) \widehat{A}_{n,t,i} \Big) - \beta d_{n,t,i}^{\text{KL}}(\theta) \right]. \tag{1}$$

Here, $\sum_{t,i}$ abbreviates $\sum_{t=0}^{T_n - 1} \sum_{i=1}^{L_{n,t}}$; α_n determines the loss aggregation and is specified for each reported configuration in Section J; $\varrho_{n,t,i}$ is the new-to-old likelihood ratio for token $\ell_{n,t,i}$; $d_{n,t,i}^{\text{KL}}$

is its KL estimator against the reference model; and ϵ and β are the clipping and KL coefficients. The key in policy optimization is to estimate the *advantage* $\widehat{A}_{n,t,i}$, which varies across different methods: GRPO (Shao et al., 2024) normalizes each trajectory's return within $G(p)$ and uses that same trajectory-level scalar for every token (t, i) in τ_n. GiGPO (Feng et al., 2025) adds a step-level relative return-to-go estimate computed among trajectory-step pairs to the GRPO trajectory-level scalar, and uses the sum for every token generated at step t. Tree-GRPO (Ji et al., 2026) samples root-to-leaf trajectories from rollout trees and uses, for every token in τ_n, a weighted combination of trajectory-return advantages normalized across the pooled trees and within the tree containing τ_n. The advantage estimators for these methods are presented in Section I for completeness.

3 Regret-based Feedback for Agentic Tasks

3.1 From Regret to Step-Level Credit

Fix a task instance p and a common horizon T for analysis. If the agent-environment interaction terminates before T, we extend it through T using zero-reward absorbing transitions and a singleton no-op action at each absorbing prefix. Let Ξ_t be the set of feasible task-conditioned prefixes at step t. Let $\mathcal{A}(\xi_t)$ be the feasible actions at prefix ξ_t, and let Π_T be the class of finite-horizon, nonstationary policy sequences $\nu = (\nu_t)_{t=0}^{T-1}$ with $\nu_t(\cdot \mid \xi_t) \in \Delta(\mathcal{A}(\xi_t))$ for every $\xi_t \in \Xi_t$, where $\Delta(\mathcal{A}(\xi_t))$ denotes the probability simplex over $\mathcal{A}(\xi_t)$. Let $P_\rho^\nu(\cdot \mid \xi_t)$ denote the law of the continuation trajectory induced jointly by ν and the environment law ρ from ξ_t, and let $P_\rho^\nu(\cdot \mid \xi_t, a)$ denote the corresponding law when action a is taken first and ν is followed thereafter. Define $G_t := \sum_{k=t}^{T-1} \gamma^{k-t} r_k^{\text{step}}$. For any policy sequence $\nu \in \Pi_T$, its value, action value, and advantage functions are defined as:

$$V_{\rho,t}^\nu(\xi_t) := \mathbb{E}_{\tau \sim P_\rho^\nu(\cdot|\xi_t)}[G_t], \quad Q_{\rho,t}^\nu(\xi_t, a) := \mathbb{E}_{\tau \sim P_\rho^\nu(\cdot|\xi_t,a)}[G_t], \quad A_{\rho,t}^\nu(\xi_t, a) := Q_{\rho,t}^\nu(\xi_t, a) - V_{\rho,t}^\nu(\xi_t).$$

Define the optimal value and action-value functions by

$$V_{\rho,t}^\star(\xi_t) := \sup_{\mu \in \Pi_T} V_{\rho,t}^\mu(\xi_t), \qquad Q_{\rho,t}^\star(\xi_t, a) := \mathbb{E}_\rho\left[r_t^{\text{step}} + \gamma V_{\rho,t+1}^\star(\xi_{t+1}) \mid \xi_t, a\right],$$

with $V_{\rho,T}^\star(\xi_T) = 0$ and $V_{\rho,t}^\star(\xi_t) = \sup_{a \in \mathcal{A}(\xi_t)} Q_{\rho,t}^\star(\xi_t, a)$. When the supremum at ξ_0 is attained, let $\pi^\star \in \arg\max_{\mu \in \Pi_T} V_{\rho,0}^\mu(\xi_0)$ denote an optimal feasible policy from ξ_0. For any $\pi \in \Pi_T$, define its per-episode regret relative to the full class of feasible policies in the induced finite-horizon history-based MDP by $\mathrm{PReg}_\rho(\pi; \Pi_T \mid \xi_0) := V_{\rho,0}^\star(\xi_0) - V_{\rho,0}^\pi(\xi_0)$.

Immediate regret. For an action a at prefix ξ_t, define its **immediate regret** as $\mathrm{IReg}_{\rho,t}(\xi_t, a) := V_{\rho,t}^\star(\xi_t) - Q_{\rho,t}^\star(\xi_t, a)$. This nonnegative quantity is the standard *action suboptimality gap*: whenever the action supremum is attained, it is zero exactly for actions in $\arg\max_{a \in \mathcal{A}(\xi_t)} Q_{\rho,t}^\star(\xi_t, a)$. We use its negative as the optimal advantage of action a at prefix ξ_t:

$$A_{\rho,t}^\star(\xi_t, a) := -\mathrm{IReg}_{\rho,t}(\xi_t, a) = Q_{\rho,t}^\star(\xi_t, a) - V_{\rho,t}^\star(\xi_t). \tag{2}$$

Proposition 1. *Assume rewards are bounded. For any $\pi \in \Pi_T$,*

$$\mathrm{PReg}_\rho(\pi; \Pi_T \mid \xi_0) = \mathbb{E}_{\tau \sim P_\rho^\pi(\cdot|\xi_0)}\left[\sum_{t=0}^{T-1} \gamma^t \mathrm{IReg}_{\rho,t}(\xi_t, a_t)\right] = -\mathbb{E}_{\tau \sim P_\rho^\pi(\cdot|\xi_0)}\left[\sum_{t=0}^{T-1} \gamma^t A_{\rho,t}^\star(\xi_t, a_t)\right].$$

By Proposition 1, per-episode regret is exactly the expected discounted sum of the immediate regrets incurred along the policy's trajectories, which identifies $A_{\rho,t}^\star$ as a step-level regret signal. Separately, whenever the action maxima are attained, Bellman optimality implies that a policy selecting only $A_{\rho,t}^\star$-maximizing actions at every prefix reachable from ξ_0 is optimal from ξ_0. The proof of Proposition 1 is given in Section D.2 for completeness.

An auxiliary objective for step-level feedback. Motivated by Proposition 1, and suppressing ρ in the notation, we may treat $A_t^\star(\xi, a)$ as an auxiliary reward for taking action a at prefix ξ. Define the *task loss* $\mathcal{L}_1(\theta) := V_0^\star(\xi_0) - V_0^{\pi_\theta}(\xi_0)$, whose minimization is equivalent to maximizing the expected task return. Let $d_t^{\text{old}}(\xi)$ be the probability that $\pi_{\theta_{\text{old}}}$ reaches prefix ξ at step t. We then define an auxiliary loss

$$\mathcal{L}_2(\theta; \theta_{\text{old}}) := -\sum_{t,\xi} \gamma^t d_t^{\text{old}}(\xi) \sum_a \pi_{\theta,t}(a \mid \xi) A_t^\star(\xi, a), \tag{3}$$

which is the negative of the expected $A_t^\star$ of actions selected by π_θ at prefixes weighted by their visitation under the old policy. With these prefix weights fixed, the negative gradient of $\mathcal{L}_2$ is the REINFORCE term (Williams, 1992) that weights $\nabla_\theta \log \pi_{\theta,t}(a \mid \xi)$ by $A_t^\star(\xi, a)$.

Approximating immediate regret through sampled search. To approximate immediate regret, we form a sampled rollout subtree $\widehat{\mathcal{T}}(\xi_t)$ from continuations generated under $\pi_{\theta_{\text{old}}}$. Let $\widehat{\mathcal{A}}(\xi_t)$ be its sampled child actions, $\widehat{\mathcal{L}}(\xi_t, a)$ the terminal leaves below action a, and $R_t(\ell \mid \xi_t) := \sum_{k=t}^{T-1} \gamma^{k-t} r_k^{\text{step}}$ the return-to-go to ℓ. When $\rho(\cdot \mid \xi, a)$ is a point mass at every feasible task-conditioned complete prefix–action pair, define the sampled-maximum estimator $\widehat{Q}_t^{\max}(\xi_t, a) := \max_{\ell \in \widehat{\mathcal{L}}(\xi_t, a)} R_t(\ell \mid \xi_t)$. In this deterministic setting, $\widehat{Q}_t^{\max}$ is the best sampled continuation return below action a and can equivalently be computed by recursive maximum backups over the sampled rollout subtree. The resulting action-value estimates provide a finite-search policy-improvement signal over the sampled actions. The optimization below uses this feedback to update π_θ. For our language-agent experiments, we also use the pseudo-posterior predictive estimator $\widehat{Q}_t^{\text{post}}$, which incorporates predicted returns for unsampled continuation leaves when estimating action values, as detailed in Section H.4. Writing $\widehat{Q}_t$ for the estimator used in a given experiment, define

$$\widehat{V}_t(\xi_t) := \max_{a \in \widehat{\mathcal{A}}(\xi_t)} \widehat{Q}_t(\xi_t, a), \qquad \widehat{A}_t(\xi_t, a) := \widehat{Q}_t(\xi_t, a) - \widehat{V}_t(\xi_t) = -\widehat{\text{IReg}}_t(\xi_t, a).$$

Thus, $\widehat{A}_t$ compares each sampled action's value estimate with the prefix-wise maximum. The reliability of this comparison depends on the underlying action-value estimates. Proposition 14 provides a finite-sample coverage bound for $\widehat{Q}_t^{\max}$ for deterministic environments; Sections H.6 and H.7 provide evidence in both synthetic and ALFWorld environments on the quality of $\widehat{Q}_t^{\text{post}}$ as an estimator, and discuss conditions under which it is a good one.

Policy optimization. For trajectory n, let $\widehat{A}_{n,t}^{\text{reg}}$ denote the regret-based component derived from $\widehat{A}_t$ for the action taken at agent step t. If no valid sampled-subtree comparison is available at that prefix, we set this component to zero. It is broadcast to the generated tokens at that agent step and added to the base advantage estimate to form $\widehat{A}_{n,t,i}^{\text{RegPO}} := \widehat{A}_{n,t,i}^{\text{base}} + w_{\text{reg}} \widehat{A}_{n,t}^{\text{reg}}$, where $w_{\text{reg}} > 0$. This combined advantage estimate is used in the clipped surrogate of Equation (1). After each update, we collect new rollout subtrees under the updated policy and repeat the procedure. $\widehat{A}^{\text{RegPO}}$ serves as an estimator of $A_t^{\pi_{\theta_{\text{old}}}} + w_{\text{reg}} A_t^\star$, whose rationale can be justified by the following theorem.

Theorem 1. *Under the representation and differentiability conditions in Sections E.1 and E.2, $\pi^\star$ globally minimizes both $\mathcal{L}_1$ and $\mathcal{L}_2(\cdot; \theta_{\text{old}})$. Moreover, at $\theta = \theta_{\text{old}}$,*

$$-\nabla_\theta(\mathcal{L}_1 + w_{\text{reg}}\mathcal{L}_2) = \mathbb{E}_{\rho, \pi_{\theta_{\text{old}}}}\left[\sum_t \gamma^t \nabla_\theta \log \pi_{\theta,t}(a_t \mid \xi_t)\left(A_t^{\pi_{\theta_{\text{old}}}} + w_{\text{reg}} A_t^\star\right)\right].$$

Thus, the on-policy policy-gradient estimator using the exact combined advantage is unbiased for the negative gradient of $\mathcal{L}_1 + w_{\text{reg}}\mathcal{L}_2$ evaluated at the old policy.

The proofs are deferred to Section E.1 and Section E.2. Sharing a global minimizer concerns the optimal solution; a descent step for the combined loss need not decrease $\mathcal{L}_1$. For tabular policies, Proposition 7 further gives an $O(N^{-1/2})$ bound on average task loss when $\mathcal{L}_2$ is updated using exact $A_t^\star$ and the old-policy prefix weights are refreshed after each update.

3.2 Credit-Assignment Differences among Advantage Estimates

We compare how GRPO, Tree-GRPO, and **RegPO** assign credit to actions at individual steps. For analysis, we use a finite-horizon history-based MDP with terminal rewards in $[0, 1]$ and $\gamma = 1$, where $\rho(\cdot \mid \xi, a)$ is a point mass at every feasible task-conditioned complete prefix–action pair. The evaluated agentic environments satisfy the same determinism condition. We work with the parsed-action policy π_θ from Section 2.

GRPO's trajectory-level credit assignment. GRPO samples complete trajectories from the initial prefix and broadcasts each trajectory-level relative advantage estimate to every token in that trajectory. It therefore does not deliberately compare alternative actions using multiple continuations from a fixed intermediate prefix. Proposition 2 formalizes a related pointwise limitation: one scalar cannot generally match the distinct advantages of actions taken at different prefixes.

Tree-GRPO and regret-based feedback at a fixed prefix. Tree-GRPO assigns its intra-tree relative advantage estimate to each complete trajectory. Under the parsed-action and balanced-sampling conditions in Section D.4, at a two-action root ξ with actions a, b, its intra-tree update favors the action with the higher sample mean continuation return. With independent continuations sampled under the rollout policy π, the sample-mean difference converges to $Q^\pi(\xi, a) - Q^\pi(\xi, b)$. **RegPO**

instead constructs a sampled-search proxy for the regret-based advantage, whose ideal pairwise signal is $Q^\star(\xi, a) - Q^\star(\xi, b)$. Thus, the two signals can favor different actions when the Q^π and $Q^\star$ differences have opposite signs. Section D.4 details the exact weighted contribution, its assumptions, and its relation to the token-level update. To study how this disagreement affects reaching and learning at later prefixes, we introduce the Hard-Safe tree, as a canonical model for agentic tasks with long horizons.

Definition 1 (Hard-Safe tree)**.** *Fix an integer $H \geq 1$ and $c \in (0, 1)$. At a root prefix ξ, the action set is* $\{\text{Safe}, \text{Hard}\}$. Safe *terminates with return c. Choosing* Hard *starts a sequence of H binary* Hard steps*. At Hard step k, the available actions are $a_k^\star$ and $\bar{a}_k$. Selecting $a_k^\star$ at every Hard step yields return one after step H; selecting any $\bar{a}_k$ terminates immediately with return zero. We write ξ_k for the prefix at Hard step k, reached after choosing* Hard *and then selecting $a_1^\star, \ldots, a_{k-1}^\star$.*
Under the current policy π, let $q_k := \pi(a_k^\star \mid \xi_k)$ and $p_\pi := \prod_{k=1}^{H} q_k$. By construction,

$$Q^\star(\xi, \text{Hard}) = 1, \quad Q^\pi(\xi, \text{Hard}) = p_\pi, \quad Q^\star(\xi, \text{Safe}) = Q^\pi(\xi, \text{Safe}) = c. \tag{4}$$

We study the regime $0 < p_\pi < c < 1$, in which the rankings induced by Q^π and $Q^\star$ disagree. Figure 1(a) illustrates this construction. In this regime, the two action-value differences are

$$Q^\pi(\xi, \text{Hard}) - Q^\pi(\xi, \text{Safe}) = p_\pi - c < 0, \quad Q^\star(\xi, \text{Hard}) - Q^\star(\xi, \text{Safe}) = 1 - c > 0.$$

Tree-GRPO's mean-return comparison favors Safe; the exact regret-based advantage favors Hard. On a fixed sampled rollout subtree, a single successful Hard continuation among m samples yields a sample mean of $1/m$ but a sampled maximum of 1. If $1/m < c$, these comparisons favor Safe and Hard, respectively, under the conditions above; Proposition 3 gives the finite-sample analysis.

Agentic interpretation in ALFWorld. Hard models a branch whose higher achievable return requires reliable downstream execution, whereas Safe offers an easier moderate return. A related pattern appears in the ALFWorld task "put a hot mug in cabinet." At a prefix where the agent holds an unheated mug at the cabinet, immediate placement is plausible but leaves the heating requirement unsatisfied; going first to the microwave appears to move away from the destination but enables the correct heat-return-place sequence. The early-policy mean action-value comparison favors placement, whereas the stronger-policy proxy for $Q^\star$ in Section 4.2.2 favors the microwave-first branch because premature placement must be undone.

The exact combined advantage in the Hard-Safe tree. To analyze the exact counterpart of the combined advantage used in Equation (1), we use the normalized form $\lambda A^\pi + (1 - \lambda)A^\star$, where $\lambda \in [0, 1]$. At the root, this mixture favors Hard over Safe exactly when $\lambda(p_\pi - c) + (1-\lambda)(1-c) > 0$. For $\lambda > 0$, this mixture is proportional to the additive convention $A^\pi + w_{\text{reg}} A^\star$, with $w_{\text{reg}} = (1 - \lambda)/\lambda$; the factor λ rescales the advantage contribution without changing its direction. In the sampled comparison above, the sampled mean and sampled maximum replace p_π and 1, respectively (Remark 2). Section F.2 gives the two Hard-Safe losses and derives the corresponding gradients at the root and Hard steps.

Learning dynamics in the Hard-Safe tree. Let π_τ denote the policy at continuous learning time τ. Under a tabular parameterization with a separate logit at each prefix, let $p(\tau) := \pi_\tau(\text{Hard} \mid \xi) = \sigma(z(\tau))$ and $q_k(\tau) := \pi_\tau(a_k^\star \mid \xi_k) = \sigma(\phi_k(\tau))$. We suppress the dependence on τ, with dots denoting τ-derivatives. Using the exact combined advantage $\lambda A^\pi + (1 - \lambda)A^\star$ gives

$$\dot{z} = p(1-p)\big[\lambda(p_\pi - c) + (1-\lambda)(1-c)\big], \qquad \dot{\phi}_k = p \left(\prod_{j<k} q_j \right) q_k(1 - q_k) \left[\lambda \prod_{j>k} q_j + 1 - \lambda \right]. \tag{5}$$

To analyze estimated feedback, we compare continuous expected updates using A^π alone with those using $\lambda A^\pi + (1 - \lambda)\widehat{A}$, from the common initialization $p(0) = p_0 \in (0, 1)$ and $q_k(0) = q_0 \in (0, 1)$ for all k. At each decision, let $\Delta_\pi, \Delta_\star, \widehat{\Delta}$ denote the action-value gaps induced by $Q^\pi, Q^\star, \widehat{Q}$, respectively: Hard minus Safe at the root and $a_k^\star$ minus $\bar{a}_k$ at Hard step k.

Theorem 2. *If $\widehat{\Delta} \geq \Delta_\pi$ at every decision and learning time, then every Hard step is reached with at least as high probability as under updates using only the current-policy advantage A^π, and every expected-return threshold above c is attained no later. If, in addition, for some fixed $\beta \in (0, 1]$, $\widehat{\Delta} \geq (1 - \beta)\Delta_\pi + \beta\Delta_\star$ at every decision and learning time and $(1 - \lambda)\beta \geq c$, then reaching $V^\pi \geq 1 - \epsilon$ takes $O(H^2/\epsilon)$ continuous time for any $0 < \epsilon < 1 - c$.*

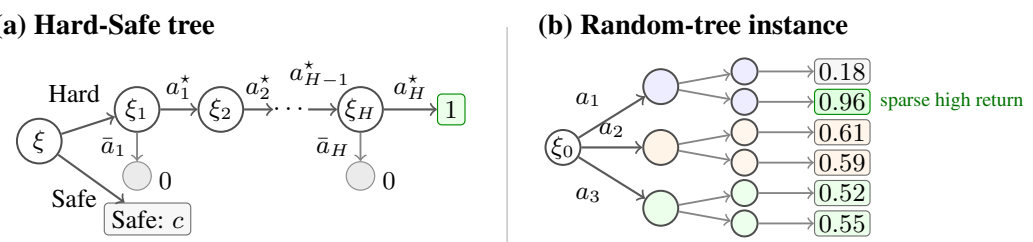

Figure 1: **Finite tree-structured MDPs. (a)** Safe returns c; Hard returns 1 only after $(a_1^\star, \ldots, a_H^\star)$, and 0 after any $\bar{a}_k$. Thus, $Q^\star(\xi, \mathrm{Hard}) = 1$, whereas $Q^\pi(\xi, \mathrm{Hard})$ may remain below c. **(b)** Random-tree instances are complete depth-four ternary trees with leaf returns generated once and held fixed. These returns can induce different action rankings under Q^π and $Q^\star$.

The first conclusion applies in particular when $\Delta_\pi \le \widehat{\Delta} \le \Delta_\star$ throughout learning. The dynamics in Equation (5) correspond to $\widehat{A} = A^\star$ and $\beta = 1$, so $\lambda \le 1 - c$ is sufficient for the polynomial bound. In contrast, when $q_0^H < c$, Proposition 10 shows that reaching the same threshold using A^π alone takes time exponential in H. Section F.3 gives the complete assumptions and proof. Section 4.1.1 evaluates the sampled end-to-end procedures across the normalized mixture weights.

4 Experiments

We evaluate **RegPO** on two families of finite tree-structured MDPs and four agentic benchmarks. The tree-structured MDP experiments isolate the effects of different credit-assignment rules in the controlled deterministic-MDP setting of Section 3.2, while the agentic benchmarks assess **RegPO** against baseline reinforcement-learning methods.

4.1 Credit Assignment in Finite Tree-Structured MDPs

We evaluate the methods on two families of finite tree-structured MDPs with deterministic transitions and rewards: the Hard-Safe and random-tree families. These controlled environments test how different credit-assignment rules affect action selection, the probability of reaching later prefixes, and learning efficiency. The Hard-Safe family isolates a prescribed disagreement between the rankings induced by Q^π and $Q^\star$, whereas random trees test whether the effects extend beyond this construction. Random-tree instances are randomly generated depth-four ternary trees (Figure 1). In the sampled finite-tree experiments, Tree-GRPO uses the recursive mean-value estimator analyzed in Sections 3.2 and D.4; we refer to this condition as Tree-GRPO (recursive mean).

For the finite-tree comparisons, we evaluate $\lambda \in \{0, 1/4, 1/2, 3/4, 1\}$ using the exact mixture $\lambda A^\pi + (1-\lambda)A^\star$ and its sampled counterpart $\lambda \widehat{A}^{\mathrm{intra}} + (1-\lambda)\widehat{A}^{\mathrm{reg}}$. The endpoints recover the regret-based and current-policy advantages, respectively; GRPO is a separate baseline.

4.1.1 Sampled End-to-End Hard-Safe Training

At each training step, each mixture condition samples a separate recursively branched rollout tree: at every visited nonterminal prefix, $b = 10$ actions are sampled from the old policy, and each sampled nonterminal child is recursively expanded. We fix GRPO's group size to match the expected number of generated actions under the initial policy π_0. Details are provided in Section J.1 and Table 5.

We initialize with $p_0 = 0.5$ at the root and $q_0 = 0.72$ for $a_k^\star$ at every Hard step; we set $c = 0.6$ and train for 1,000 steps. Complete configurations are reported in Table 5, and Section K.2 provides additional results from a sweep over depth H with a lower initial optimal-action probability at each Hard step. At $H = 20$, Figure 2 shows that increasing the regret-based weight preserves the probability of selecting Hard, raises the probability of reaching the final Hard step, and improves the Hard continuation value earlier in training, despite $Q^\pi(\xi, \mathrm{Hard}) < c < Q^\star(\xi, \mathrm{Hard})$ initially.

4.1.2 Random-Tree Family

Each random-tree instance is a complete depth-four ternary tree with a separate tabular softmax policy at every internal node. At generation, we assign returns to its terminal nodes, including 1, 2, 4, or 8 randomly placed high-return leaves, and hold them fixed during training. We vary the initial action probabilities to produce different degrees of disagreement between the rankings induced by $Q^{\pi_{\theta_0}}$ and $Q^\star$. At initialization, 66.7% of instances contain root actions a_1, a_2 such that $Q^\star(\xi_0, a_1) > Q^\star(\xi_0, a_2)$ but $Q^{\pi_{\theta_0}}(\xi_0, a_1) < Q^{\pi_{\theta_0}}(\xi_0, a_2)$. Training efficiency is the first

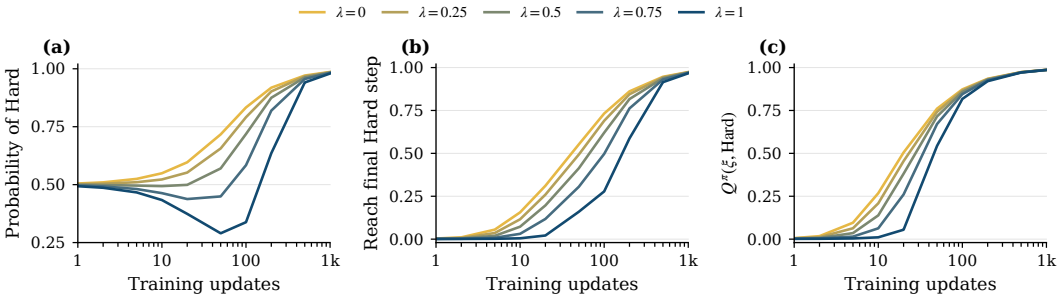

Figure 2: **Sampled end-to-end Hard-Safe training at $H = 20$. (a)** Probability of selecting Hard at the root. **(b)** Probability of reaching the final Hard step under the learned policy. **(c)** Probability of completing the Hard continuation after selecting Hard. Curves show means; shading shows ± 1 standard deviation over 500 seeds.

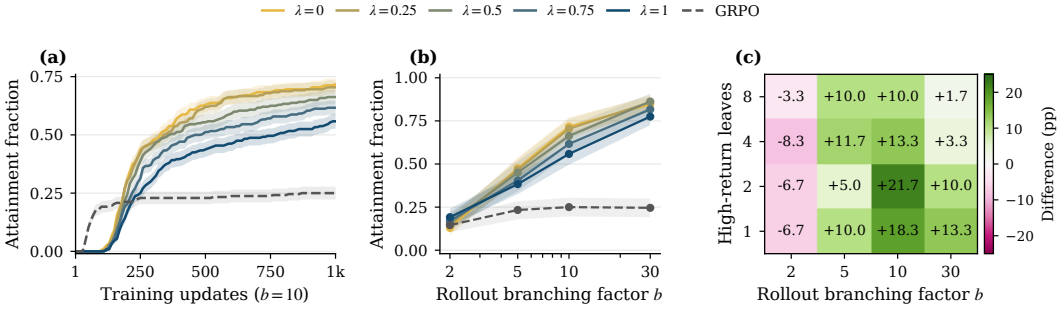

Figure 3: **Performance on the random-tree family.** Attainment denotes closing 95% of the initial-to-optimal expected-root-return gap. **(a)** Attainment over training at $b = 10$; shading shows ± 1 bootstrap standard deviation across 240 instances. **(b)** Attainment after 1,000 updates by rollout branching factor; shading shows instance-bootstrap 95% confidence intervals. **(c)** The $\lambda = 0$-minus-$\lambda = 1$ attainment difference (percentage points) by b and high-return-leaf count.

step at which the expected root return closes 95% of its initial-to-optimal gap. For each rollout branching factor b, each mixture condition samples one recursively branched rollout tree per step. We choose GRPO's group size to generate approximately the same number of actions and train all conditions for 1,000 steps. Complete configurations and the GRPO matching calculation are in Table 5 and section J.1.

Figure 3 shows that performance across mixture weights depends on rollout coverage. At $b = 10$, $\lambda = 1/2$ exceeds Tree-GRPO ($\lambda = 1$) by 10.4 percentage points (paired bootstrap 95% CI: $[5.8, 15.0]$); at $b = 2$, Tree-GRPO has the highest point estimate (19.2% versus 17.1% for $\lambda = 3/4$), but the paired CI includes zero. Figure 3**(c)** compares updates using the sampled regret-based advantage ($\lambda = 0$) with those using Tree-GRPO's intra-tree advantage estimate ($\lambda = 1$), grouped by the number of high-return leaves. The former-minus-latter difference is negative for every leaf count at $b = 2$ and positive for every leaf count at $b \geq 5$. A representative instance in Section K.1 links an initial action-ranking disagreement to root-action selection, the probability of reaching a depth-three prefix, and the corresponding action value. Section G shows that this separation persists with correct rankings because current-policy gaps are attenuated by continuation success probability.

4.2 Agentic Benchmark Experiments

4.2.1 Main Results

We evaluate **RegPO** against GRPO, GiGPO, and Tree-GRPO on four interactive benchmarks: Beat Block Hammer (Chen et al., 2026), WebShop (Yao et al., 2022), MiniWoB++ (Liu et al., 2018), and ALFWorld (Shridhar et al., 2021). We report success rate throughout; benchmark descriptions, evaluation splits, and protocols are provided in Section J.2.1. We match rollout budgets across GRPO, Tree-GRPO, and **RegPO** within each benchmark. Tree-GRPO and **RegPO** use Tree-GRPO's rollout-tree construction (Ji et al., 2026) and the same branching configuration. **RegPO** adds its estimated prefix-wise centered regret-based component $\widetilde{A}^{\text{reg}}$, with $w_{\text{reg}} = 1$; see Section H.3. Other experimental details are reported in Section J.

(a) Main results

Benchmark	Split	Initial	GRPO	GiGPO	Tree-GRPO	REGPO
RoboTwin 2.0 (Beat Block Hammer)	ID	31.2	64.1	—	59.4	**78.9**
	OOD	28.9	67.2	—	43.8	**73.4**
WebShop	Test	1.0	57.6 ±0.1	62.0 ±0.0	48.1 ±0.1	**65.9** ±0.1
MiniWoB++	Eval	1.2	26.0 ±0.9	30.7 ±0.9	25.0 ±0.0	32.8 ±1.6
ALFWorld	ID	0.0	28.6 ±0.3	45.5 ±5.4	23.1 ±1.1	**65.7** ±1.5
	OOD	0.0	21.3 ±0.5	33.3 ±0.9	19.9 ±0.7	**62.4** ±0.4

(b) ALFWorld (3 training seeds)

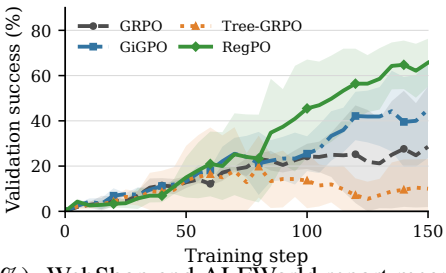

Table 1: **Agentic benchmark results. (a)** Success rate (%). WebShop and ALFWorld report mean ± standard deviation across three training seeds; MiniWoB++ reports mean ± standard deviation across three evaluation repetitions of one training seed; Beat Block Hammer uses one training seed with greedy decoding and therefore reports no deviation. Checkpoint selection and evaluation protocols are detailed in Section J.2.1. **(b)** ALFWorld ID validation success during training, with mean and standard-deviation shading across three training seeds.

The dash for GiGPO on Beat Block Hammer indicates non-applicability: its step-level relative advantage requires trajectory–step pairs matched by environment-state identifiers, which are unavailable for this benchmark. We reimplement all applicable baselines under matched benchmark-specific training horizons, rollout budgets, and evaluation sets. Baseline configurations follow the corresponding prior work where compatible; necessary adaptations and the hyperparameter-search protocol are detailed in Section J.2.2.

Under the reporting protocol in Section J.2.1, the evaluated **REGPO** configuration attains the highest point estimate in every row. The largest point-estimate margins occur on ALFWorld: 20.2 percentage points ID and 29.1 points OOD over the strongest baseline. Panel (b) provides learning-curve context for these results. **REGPO** also has the highest point estimates on both Beat Block Hammer splits and WebShop. The point-estimate margin on MiniWoB++ is more modest, at 2.1 points.

4.2.2 MECHANISTIC POLICY COMPARISON IN ALFWORLD

To better understand these results, we examine how training changes policy behavior in ALFWorld. We first test whether the action-ranking disagreement analyzed in Section 3.2 and observed in the finite-tree experiments of Section 4.1 also appears in an agentic setting, using a stronger-policy proxy for $Q^\star$. We then examine whether the trained policies more reliably select actions with useful continuations, reach task-relevant intermediate states, execute those continuations successfully, and complete the task. We compare the Update-10 policy with the trained Tree-GRPO and **REGPO** policies. The initial policy produces valid actions at only 0.6% of validation steps, leaving too few informative continuations; by update 10, this rate reaches 98.2%, providing an early-stage policy that reliably produces valid actions. Analogous MiniWoB++ analyses are reported in Section M.

Update 10	Tree-GRPO	REGPO
33.0 ±3.4	43.1 ±4.1	**50.7** ±4.1

Table 2: **Agreement between estimated action rankings in ALFWorld (%).** ± denotes one standard deviation from a task-level bootstrap.

Action-ranking disagreement. We compare the action-value rankings of an early-stage policy and the trained Tree-GRPO and **REGPO** policies with rankings induced by a stronger-policy proxy for $Q^\star$. We freeze 96 selected prefixes from trajectories sampled from the trained **REGPO** policy on 32 held-out ID tasks in ALFWorld. At each prefix ξ, we evaluate the **REGPO** trajectory action and three distinct feasible alternative actions proposed by GPT-5.5, giving 384 prefix–action pairs shared by every evaluated policy. For each learned policy π and action a, we estimate $\widehat{Q}^{\pi}_{\text{mean}}(\xi, a)$ by the mean discounted return of eight independent continuations. One GPT-5.5 continuation from the same prefix and forced action provides $\widehat{Q}_{\text{GPT}}(\xi, a)$, a stronger-policy proxy for $Q^\star(\xi, a)$. For each unordered action pair (a_1, a_2), the two estimated action rankings agree strictly when

$$\big(\widehat{Q}^{\pi}_{\text{mean}}(\xi, a_1) - \widehat{Q}^{\pi}_{\text{mean}}(\xi, a_2)\big)\big(\widehat{Q}_{\text{GPT}}(\xi, a_1) - \widehat{Q}_{\text{GPT}}(\xi, a_2)\big) > 0.$$

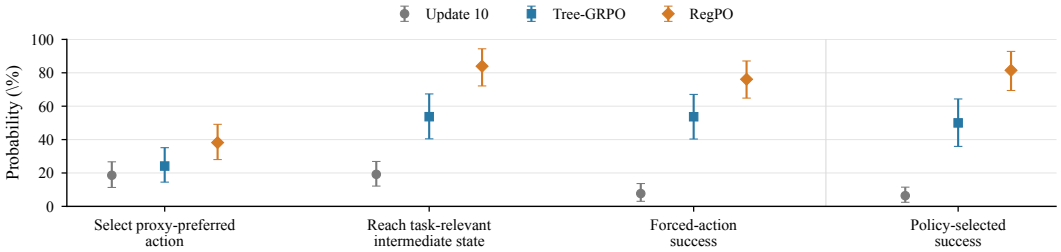

Figure 4: **Policy behavior from frozen selected prefixes in ALFWorld.** Points show task-level means and error bars show task-bootstrap 95% confidence intervals. Measurement cohorts and sample sizes are detailed in Section L.

Table 2 reports the strict agreement rates. The rankings induced by $\widehat{Q}^{\pi}_{\text{mean}}$ and $\widehat{Q}_{\text{GPT}}$ have low strict agreement for the early policy, consistent with the differing action rankings in Section 4.1.2. Agreement is higher after training and highest for **REGPO**.

Action selection, intermediate-state reach, and continuation execution. For the Update-10 policy, write $\widehat{Q}^{10}_{\text{mean}}$ for its action-value estimate. We call (a^+, a^-) a *strict disagreement pair* when

$$\widehat{Q}^{10}_{\text{mean}}(\xi, a^+) < \widehat{Q}^{10}_{\text{mean}}(\xi, a^-), \qquad \widehat{Q}_{\text{GPT}}(\xi, a^+) > \widehat{Q}_{\text{GPT}}(\xi, a^-).$$

We identify 153 such pairs, and every a^+ has a successful GPT-5.5 continuation, although $\widehat{Q}^{10}_{\text{mean}}$ ranks it below its paired alternative. For each pair, define the estimated current-policy and pseudo-posterior gaps as

$$\widehat{\Delta}_{\pi} := \widehat{Q}^{10}_{\text{mean}}(\xi, a^+) - \widehat{Q}^{10}_{\text{mean}}(\xi, a^-), \qquad \widehat{\Delta} := \widehat{Q}^{\text{post},10}(\xi, a^+) - \widehat{Q}^{\text{post},10}(\xi, a^-),$$

where $\widehat{Q}^{\text{post},10}$ is the leave-one-task-out pseudo-posterior estimator from Section H.7. For 92.2% of the selected disagreement pairs, the pseudo-posterior gap is at least as large as the Update-10 gap, $\widehat{\Delta} \geq \widehat{\Delta}_{\pi}$, matching the local gap-ordering condition in Theorem 2.

For each task, fixed task-family rules (Table 12) derive candidate intermediate-state conditions from its symbolic goal parameters, such as reaching the target location while holding the required object. We determine whether each condition holds from ALFWorld's symbolic state, including object locations and properties. At each selected prefix ξ, we retain only conditions M that are false at ξ but become true at a nonterminal state along a fixed successful GPT-5.5 continuation. Let $S(\tau)$ indicate task success. For each evaluated policy π, we measure action selection as $\pi(a^+ \mid \xi)$ and intermediate-state reach as $\Pr_{\tau \sim P^{\pi}_{\rho}(\cdot|\xi)}[\exists \xi' \in \tau : M(\xi') = 1]$. We measure forced-action success for a candidate action a as $\Pr_{\tau \sim P^{\pi}_{\rho}(\cdot|\xi,a)}[S(\tau) = 1]$ and policy-selected success as $\Pr_{\tau \sim P^{\pi}_{\rho}(\cdot|\xi)}[S(\tau) = 1]$. Section L gives the sampling and aggregation protocols.

These measurements parallel the Hard-Safe construction and experiment in Sections 3.2 and 4.1.1: a strict disagreement pair identifies an action with a useful continuation that the Update-10 mean action-value estimate ranks below an alternative; action selection measures whether the policy chooses it; intermediate-state reach measures whether the policy reaches a task-relevant intermediate state; and forced-action success measures whether the continuation succeeds after fixing the action. Policy-selected success additionally reports end-to-end success when the policy chooses the next action itself.

Figure 4 reports the four behavioral measurements. Compared with Tree-GRPO, **REGPO** has point estimates higher by 14.0 percentage points for selection of a^+, 30.2 points for intermediate-state reach, 22.5 points for forced-action success, and 31.5 points for policy-selected success. Across these related cohorts, the action-ranking disagreement at Update 10 is accompanied by differences in action selection, task-relevant intermediate-state reach, and continuation execution.

For example, on the task "put some cellphone on bed," the Update-10 mean action-value estimate ranks `go to bed 1` below `inventory`, the command that reports the objects currently held by the agent, whereas the stronger-policy proxy gives the opposite ranking. The Update-10 policy selects `go to bed 1` in $2/8$ trials, whereas both trained policies select it in $8/8$. Correspondingly, intermediate-state reach increases from 74.6% to 100%, and forced-action success from $5/8$ to $8/8$.

AI USAGE

OpenAI Codex and Anthropic Claude Code were used to assist with grammar checking, shortening drafts, and writing code for the experiments. All AI-assisted output was reviewed and verified by the authors, who take full responsibility for the content, correctness, and conclusions of this work.

REPRODUCIBILITY STATEMENT

The appendices provide the complete theoretical assumptions and proofs, the sampled action-value estimators, experimental configurations, evaluation protocols, and additional results. Public benchmarks and initialization checkpoints are cited in the paper.

REFERENCES

Alekh Agarwal, Sham M. Kakade, Jason D. Lee, and Gaurav Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *Journal of Machine Learning Research*, 22(98):1–76, 2021. URL https://www.jmlr.org/papers/v22/19-736.html.

Pierre Alquier, James Ridgway, and Nicolas Chopin. On the properties of variational approximations of Gibbs posteriors. *Journal of Machine Learning Research*, 17(236):1–41, 2016. URL https://www.jmlr.org/papers/v17/15-290.html.

Raman Arora, Ofer Dekel, and Ambuj Tewari. Online bandit learning against an adaptive adversary: From regret to policy regret. In *Proceedings of the 29th International Conference on Machine Learning*, pp. 1503–1510, 2012. URL https://arxiv.org/abs/1206.6400.

Raman Arora, Michael Dinitz, Teodor V. Marinov, and Mehryar Mohri. Policy regret in repeated games. In *Advances in Neural Information Processing Systems*, volume 31, pp. 6733–6742, 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/hash/8643c8e2107ba86c47371e037059c4b7-Abstract.html.

Pier Giovanni Bissiri, Chris C. Holmes, and Stephen G. Walker. A general framework for updating belief distributions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(5):1103–1130, 2016. doi: 10.1111/rssb.12158. URL https://doi.org/10.1111/rssb.12158.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, Kailun Su, Tianling Xu, Guodong Liu, Mengkang Hu, Huan-ang Gao, Kaixuan Wang, Zhixuan Liang, Yusen Qin, Xiaokang Yang, Ping Luo, and Yao Mu. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. In *International Conference on Machine Learning*, 2026. URL https://icml.cc/virtual/2026/poster/62192.

Thibault Le Sellier de Chezelles, Maxime Gasse, Alexandre Lacoste, Massimo Caccia, Alexandre Drouin, Léo Boisvert, Megh Thakkar, Tom Marty, Rim Assouel, Sahar Omidi Shayegan, Lawrence Keunho Jang, Xing Han Lù, Ori Yoran, Dehan Kong, Frank F. Xu, Siva Reddy, Graham Neubig, Quentin Cappart, Russ Salakhutdinov, and Nicolas Chapados. The browsergym ecosystem for web agent research. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=5298fKGmv3. Expert Certification.

Yonathan Efroni, Gal Dalal, Bruno Scherrer, and Shie Mannor. Beyond the one-step greedy approach in reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1387–1396, 2018. URL https://proceedings.mlr.press/v80/efroni18a.html.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *Advances in Neural Information Processing Systems*, volume 38, pp. 46375–46408. Curran Associates, Inc., 2025. doi: 10.52202/085713-1544. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/420c9f777c0b4f78d515e53cf74d58b2-Abstract-Conference.html.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. WebVoyager: Building an end-to-end web agent with large multimodal models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6864–6890, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.371. URL https://aclanthology.org/2024.acl-long.371/.

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. URL

https://proceedings.iclr.cc/paper_files/paper/2026/file/8c7304e77c832ddc70075dfee081ca6c-Paper-Conference.pdf.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025a. doi: 10.15607/RSS.2025.XXI.017. URL https://www.roboticsproceedings.org/rss21/p017.html.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P. Foster, Pannag R. Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *Proceedings of the 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025b. URL https://proceedings.mlr.press/v270/kim25c.html.

Suhwan Kim, Taehyun Cho, Geonhyeong Kim, Yujin Kim, Youngsoo Jang, Moontae Lee, and Jungwoo Lee. A regret minimization framework on preference learning in large language models. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Tor Lattimore and Csaba Szepesvári. *Bandit Algorithms*. Cambridge University Press, 2020. doi: 10.1017/9781108571401. URL https://doi.org/10.1017/9781108571401.

Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaohui Yang, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, Dehui Wang, Dingxiang Luo, Yuchen Fan, Youbang Sun, Jia Zeng, Jiangmiao Pang, Shanghang Zhang, Yu Wang, Yao Mu, Bowen Zhou, and Ning Ding. SimpleVLA-RL: Scaling VLA training via reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 125846–125869, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/cbfbcb4da14235bd69b134070898ae9d-Paper-Conference.pdf.

Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. Reinforcement learning on web interfaces using workflow-guided exploration. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ry_ZRP5lX.

Chanwoo Park and Asuman E. Ozdaglar. Training with (Swap) regret loss in a single-layer self-attention model: A case study on the probability simplex, 2026. URL https://arxiv.org/abs/2607.23333.

Chanwoo Park, Xiangyu Liu, Asuman Ozdaglar, and Kaiqing Zhang. Do LLM agents have regret? a case study in online learning and games. In *International Conference on Learning Representations*, volume 2025, pp. 41214–41234, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/65ea878cb90b440e8b4cd34fe0959914-Paper-Conference.pdf.

Chanwoo Park, Ziyang Chen, Asuman Ozdaglar, and Kaiqing Zhang. Post-training LLMs as better decision-making agents: A regret-minimization approach. In *International Conference on Machine Learning*, 2026. URL https://icml.cc/virtual/2026/poster/62760.

Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. Agent Q: Advanced reasoning and learning for autonomous AI agents, 2024. URL https://arxiv.org/abs/2408.07199.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Jiadai Sun, Xinyue Yang, Yu Yang, Shuntian Yao, Wei Xu, Jie Tang, and Yuxiao Dong. WebRL: Training LLM web agents via self-evolving online curriculum reinforcement learning. In *International Conference on Learning Representations*, volume 2025, pp. 79791–79821, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/c66e1fcc9691aae706250638f36f681b-Paper-Conference.pdf.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Amrith Setlur, Matthew Yang, Charlie Snell, Jeremiah Greer, Ian Wu, Virginia Smith, Max Simchowitz, and Aviral Kumar. e3: Learning to explore enables extrapolation of test-time compute for LLMs. In *International Conference on Learning Representations*, volume 2026, pp. 127323–127361, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/ced46a50befedcb884ccf0cbe8c3ad23-Paper-Conference.pdf.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Junhong Shen, Hao Bai, Lunjun Zhang, Yifei Zhou, Amrith Setlur, Peter Tong, Diego Caples, Nan Jiang, Tong Zhang, Ameet Talwalkar, and Aviral Kumar. Thinking vs. doing: Improving agent reasoning by scaling test-time interaction. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 169640–169681. Curran Associates, Inc., 2025. doi: 10.52202/085713-5647. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/f7c4783621a20f5f11a316cd249e0252-Paper-Conference.pdf.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652. Curran Associates, Inc., 2023. doi: 10.52202/075280-0377. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=0IOX0YcCdTn.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359, 2017. doi: 10.1038/nature24270. URL https://doi.org/10.1038/nature24270.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419):1140–1144, 2018. doi: 10.1126/science.aar6404. URL https://doi.org/10.1126/science.aar6404.

Yueqi Song, Ketan Ramaneti, Zaid Sheikh, Ziru Chen, Boyu Gou, Tianbao Xie, Yiheng Xu, Danyang Zhang, Apurva Gandhi, Fan Yang, Joseph Liu, Tianyue Ou, Zhihao Yuan, Frank F. Xu, Shuyan Zhou, Xingyao Wang, Xiang Yue, Tao Yu, Huan Sun, Yu Su, and Graham Neubig. Agent data protocol: Unifying datasets for diverse, effective fine-tuning of LLM agents. In *International Conference on Learning Representations*, volume 2026, pp. 143153–143173, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/e79c09f39aca8507f527daad3601b3aa-Paper-Conference.pdf.

Wen Sun, Arun Venkatraman, Geoffrey J. Gordon, Byron Boots, and J. Andrew Bagnell. Deeply AggreVaTeD: Differentiable imitation learning for sequential prediction. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3309–3318, 2017. URL https://proceedings.mlr.press/v70/sun17d.html.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837. Curran Associates, Inc., 2022. doi: 10.52202/068431-1800. URL

https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

Zhepei Wei, Wenlin Yao, Yao Liu, Weizhi Zhang, Qin Lu, Liang Qiu, Changlong Yu, Puyang Xu, Chao Zhang, Bing Yin, Hyokun Yun, and Lihong Li. WebAgent-R1: Training web agents via end-to-end multi-turn reinforcement learning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 7909–7928. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.401. URL https://aclanthology.org/2025.emnlp-main.401/.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992. doi: 10.1007/BF00992696.

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, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2024. URL https://arxiv.org/abs/2412.15115.

Kunhe Yang, Lin F. Yang, and Simon S. Du. Q-learning with logarithmic regret. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 1576–1584. PMLR, 2021. URL https://proceedings.mlr.press/v130/yang21b.html.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, volume 35, pp. 20744–20757. Curran Associates, Inc., 2022. doi: 10.52202/068431-1508. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/82ad13ec01f9fe44c01cb91814fd7b8c-Abstract-Conference.html.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Hanchen Zhang, Xiao Liu, Bowen Lv, Xueqiao Sun, Bohao Jing, Iat Long Iong, Zhenyu Hou, Zehan Qi, Hanyu Lai, Yifan Xu, Rui Lu, Hongning Wang, Jie Tang, and Yuxiao Dong. AgentRL: Scaling agentic reinforcement learning with a multi-turn, multi-task framework, 2025. URL https://arxiv.org/abs/2510.04206.

Kai Zhang, Xiangchao Chen, Bo Liu, Tianci Xue, Zeyi Liao, Zhihan Liu, Xiyao Wang, Yuting Ning, Zhaorun Chen, Xiaohan Fu, Jian Xie, Yuxuan Sun, Boyu Gou, Qi Qi, Zihang Meng, Jianwei Yang, Ning Zhang, Xian Li, Ashish Shah, Dat Huynh, Hengduo Li, Zi Yang, Sara Cao, Lawrence Jang, Shuyan Zhou, Jiacheng Zhu, Huan Sun, Jason Weston, Yu Su, and Yifan Wu. Agent learning via early experience. In *International Conference on Machine Learning*, 2026. URL https://icml.cc/virtual/2026/poster/64488.

A Deferred Limitations

RegPO approximates regret-based advantages through sampled search and therefore inherits coverage and support limitations: under deterministic environment outcomes, its sampled maximum optimizes only over observed continuations and cannot recover sequences outside the old policy's support, while predictive extrapolation depends on model accuracy; under stochastic outcomes, rollout maxima do not consistently estimate optimal expected continuation values without repeated sampling or an environment model. Moreover, identifying a favorable early action does not improve expected return unless the policy also learns the successful downstream choices. Our theoretical results assume finite-horizon history-based MDPs together with explicit representation, coverage, and optimization conditions, including determinism for the sampled-maximum coverage and Hard-Safe visitation analyses, and do not establish convergence or monotonic improvement for LLM training with estimated advantages. Finally, branched rollouts may introduce runtime overhead, and our experiments are limited to the studied model scales, interaction horizons, and task families, leaving evaluation at larger scales and longer horizons to future work.

B DEFERRED CONCLUSION

We introduced **REGPO**, which uses sampled rollout subtrees to estimate step-level regret-based advantages for agentic post-training. Under the sampling and weighting conditions analyzed in the paper, Tree-GRPO instead induces an action comparison based on sample mean continuation returns. We prove that, in the Hard-Safe tree, updates using only Tree-GRPO's intra-tree advantage can require exponential learning time, whereas sufficiently informative regret-based advantages makes it polynomial. Our finite-tree MDP experiments show that, compared with Tree-GRPO, **REGPO** assigns more probability to the optimal root action, reaches later prefixes more often, and completes the remaining continuation more reliably. Across four LM agentic benchmarks, the evaluated **REGPO** configurations have the highest point estimate on every reported split. In ALFWorld, we find that the mean-return action rankings of an early-stage policy (Update 10) strictly agree with a stronger proxy for $Q^\star$ on only a third of action pairs, while the trained **REGPO** policy shows the highest agreement among the evaluated policies. From selected prefixes, **REGPO** also has higher point estimates than Tree-GRPO for selecting actions with useful continuations, reaching task-relevant intermediate states, and executing those continuations. **REGPO** uses rollout trees only during training and deploys the learned policy without tree expansion. Scaling to larger models and longer horizons requires broader rollout coverage and more accurate regret-based advantage estimates.

C Extended Related Work

LLMs for Agentic Applications. Recent work studies LLM agents for software engineering, tool use, and web interaction, following early systems such as ReAct and Reflexion (Yao et al., 2023; Shinn et al., 2023). Song et al. (2026) provide standardized protocols for sharing agent trajectories. Prompt engineering alone is often insufficient for complex agentic tasks, motivating post-training methods that improve decision-making through interaction (Zhang et al., 2025; Putta et al., 2024). Qi et al. (2025) train web agents with a self-evolving task curriculum, whereas Shen et al. (2025) use a curriculum that gradually increases the interaction horizon. Zhang et al. (2026) introduce an early-experience framework that samples alternative agent actions at states from expert demonstrations and executes them in the environment. The resulting next states supervise either an auxiliary next-state-prediction objective (implicit world modeling) or self-reflection that compares the alternatives with the expert actions.

Tree Search and Policy Improvement. AlphaGo Zero and AlphaZero use Monte Carlo tree search (MCTS) to construct a search policy: search visit counts supervise the policy, while outcomes of search-guided self-play supervise the value prediction (Silver et al., 2017; 2018). Within each search, the network parameters remain fixed while action selection and visit statistics evolve. The updated network then guides subsequent searches. This use of search to improve future action choices connects to multi-step policy improvement (Efroni et al., 2018).

RegPO also uses sampled continuations to guide policy improvement. Given a sampled rollout subtree, its sampled-maximum estimator evaluates each action by the best sampled continuation below that action. When environment outcomes are deterministic conditional on the complete prefix and action, this is equivalent to recursive maximum backups through the subtree. Comparing these action-value estimates at the same prefix gives sampled-search proxies for the regret-based advantages used for the next policy update, after which new rollout subtrees are collected. The optimal action values in our regret decomposition define the ideal target, whereas the sampled maximum optimizes only over the continuations represented in the sampled subtree. Thus, for a fixed sampled subtree, averaging continuation returns reflects the downstream action choices of the old policy, whereas maximum backups give feedback based on the best downstream continuation represented in that same subtree. The pseudo-posterior predictive estimator can additionally extrapolate beyond the sampled leaves, with accuracy depending on the hierarchical predictive model (Section H.5).

The distinction from MCTS is in how the continuations are collected and used. AlphaZero's MCTS adaptively selects simulations and averages their backed-up evaluations. **RegPO** constructs sampled-search proxies for regret-based advantages from already sampled rollout subtrees, without additional adaptive rollouts during this computation, and deploys the learned policy without tree search.

An additional distinction arises from action granularity. In AlphaZero, each search edge directly represents an environment action, so updated visit counts and backed-up evaluations can guide the next simulation within the same search. In our LM agent setup, by contrast, a parsed action $a_t = \mathrm{parse}(y_t)$ at interaction prefix ξ_t is induced by a variable-length LLM output $y_t = (\ell_{t,i})_{i=1}^{L_t}$ consisting of generated tokens. The inference engine operates on these tokens, whereas adaptive tree-search allocation operates on completed parsed actions and their successor interaction prefixes. Implementing this allocation therefore requires interleaving LLM generation, parsing, environment execution, backup, and selection of the next simulation. Our rollout-tree construction instead uses a prescribed branching rule and rollout budget under $\pi_{\theta_{\mathrm{old}}}$; sampled returns do not redirect additional rollouts within the same sampled rollout subtree. Regret-based advantage feedback is constructed after the subtree is collected, avoiding this additional within-subtree feedback loop while retaining the rollout-then-update interface of the PPO-style objective.

Tree-GRPO: Sampling a Tree vs. Optimizing Its Continuations. Tree-GRPO shares prefixes to collect trajectories efficiently and combines within-tree and across-tree return normalization; its analysis also establishes a connection to step-level preference learning (Ji et al., 2026). Our distinction concerns the continuation target, rather than whether Tree-GRPO can provide useful step-level preferences. Under the parsed-action and balanced-sampling conditions in Section D.4, grouping its trajectory-level advantage estimates by the action at a fixed prefix gives an induced comparison based on sample mean continuation returns, whose population targets are $Q^{\pi_{\theta_{\mathrm{old}}}}$ values. Our

sampled-maximum estimator instead selects among discovered continuations, allowing a successful suffix to raise its ancestors' credit even when that suffix has low probability under $\pi_{\theta_{\text{old}}}$. Averaging is appropriate for estimating the old policy's expected return; the difference is whether downstream actions are averaged under the old policy or optimized within the sampled tree. This is separate from averaging stochastic environment outcomes, which an expected-return planner must retain. The Hard-Safe analysis isolates when the current-policy and optimal-continuation targets disagree and how that disagreement affects subsequent visitation; it does not establish universal superiority of sampled-maximum updates.

Expert-Guided Policy Updates. **REGPO** is also related in spirit to the expert-guided policy optimization in Sun et al. (2017), where the comparator trajectories in our case may be viewed as some expert data. Specifically, under the corresponding reward formulation, Deeply AggreVaTeD updates a policy using the expert action value $Q_t^{\pi^{\mathrm{E}}}(\xi, a)$ at prefixes visited by a policy that mixes the policy being optimized with the expert policy π^{E}. When $\pi^{\mathrm{E}} = \pi^\star$, this becomes $Q_t^\star(\xi, a)$; subtracting the action-independent baseline $V_t^\star(\xi)$ yields $A_t^\star(\xi, a)$ and hence the same policy-gradient form as the ideal auxiliary objective $\mathcal{L}_2$, up to discounting and prefix-weighting conventions. The key distinction is how this signal is obtained: Deeply AggreVaTeD assumes access to an expert action-value oracle during training, whereas **REGPO** constructs sampled-search proxies for regret-based advantages from sampled rollout subtrees collected under $\pi_{\theta_{\text{old}}}$ and combines them with the base advantage estimate.

Regret-Based Training. Regret has recently been used as both an evaluation lens and a training principle for language models. Park et al. (2025) study whether LLM agents exhibit low regret in online learning and game-like interaction settings, while Park et al. (2026) use regret minimization to post-train LLM decision-making agents. Other work analyzes regret-based objectives theoretically. Park & Ozdaglar (2026) study regret and swap-regret losses in a single-layer self-attention model over probability-simplex policies, showing that regret-trained attention can realize classical no-regret and no-swap-regret dynamics such as smoothed fictitious play and Blum–Mansour-style updates. RePO (Kim et al., 2026) also uses regret terminology for LLM post-training, but in a different setting: it interprets offline preference labels as behavior-conditioned assessments of relative suboptimality and derives a preference objective containing a sequential policy-deviation term. Our setting instead begins with history-dependent environment rewards and a comparator class of feasible continuation policies. We use the resulting immediate regret to derive a regret-based advantage target for credit at individual agent steps, and construct sampled proxies for that target in an agent's action-history tree.

Policy Regret. Policy regret was introduced to address adaptive environments in which the learner's past actions change the future rewards it faces (Arora et al., 2012). Subsequent work distinguishes policy regret from external regret in repeated games with adaptive opponents (Arora et al., 2018). The distinction between external regret and policy regret is relevant to LM agent tasks: the reward associated with an action can depend on the interaction history, including previous observations, actions, and environment transitions. Our main formulation defines policy regret over continuation policies in a stochastic sequential process and recovers its action-sequence form as the deterministic-environment specialization.

Immediate Regret and Action Gaps. In stochastic bandits, the difference between the best arm mean and the mean of a selected arm is called its suboptimality gap, action gap, or immediate regret (Lattimore & Szepesvári, 2020). The sequential analogue in a finite-horizon MDP is the action suboptimality gap $\Delta_t(s, a) = V_t^\star(s) - Q_t^\star(s, a)$, and expected regret can be written as an expected sum of these gaps (Yang et al., 2021). Our immediate regret uses this same mathematical quantity on a finite action-history tree and interprets it as a single-step contribution to regret. We construct sampled-search proxies for these gaps from rollout subtrees and use the resulting regret-based feedback for agentic post-training.

D Theoretical Details and Proofs

D.1 Relation to Policy Regret and Expected-Return Maximization

Fix T and the initial task-conditioned prefix $\xi_0 = (p, o_0)$, and let $\mathcal{C}_T^{\Pi} \subseteq \Pi_T$ be a nonempty comparator class fixed independently of π. For any $\pi \in \Pi_T$, define its regret against this class by

$$\mathrm{PReg}_\rho\big(\pi; \mathcal{C}_T^{\Pi} \mid \xi_0\big) := \sup_{\mu \in \mathcal{C}_T^{\Pi}} \sum_{t=0}^{T-1} \gamma^t \mathbb{E}_{\tau \sim P_\rho^{\mu}(\cdot|\xi_0)}\big[r_t^{\mathrm{step}}\big] - \sum_{t=0}^{T-1} \gamma^t \mathbb{E}_{\tau \sim P_\rho^{\pi}(\cdot|\xi_0)}\big[r_t^{\mathrm{step}}\big].$$

Equivalently,

$$\mathrm{PReg}_\rho\big(\pi; \mathcal{C}_T^{\Pi} \mid \xi_0\big) = \sup_{\mu \in \mathcal{C}_T^{\Pi}} V_{\rho,0}^{\mu}(\xi_0) - V_{\rho,0}^{\pi}(\xi_0).$$

Each comparator policy μ is evaluated under its own trajectory law P_ρ^{μ}: changing its actions can also change later observations and rewards. This counterfactual evaluation supports interpreting the quantity as policy regret: it parallels the defining motivation for policy regret in adaptive online environments (Arora et al., 2012; 2018).

For the full comparator class Π_T,

$$\mathrm{PReg}_\rho(\pi; \Pi_T \mid \xi_0) = V_{\rho,0}^{\star}(\xi_0) - V_{\rho,0}^{\pi}(\xi_0).$$

Thus, in the fixed history-based MDP considered here, the full-class per-episode regret is also the standard optimal-return gap. Because Π_T is fixed, $V_{\rho,0}^{\star}(\xi_0)$ is independent of the learned policy, so minimizing this gap is equivalent to maximizing expected discounted return. The policy-regret interpretation therefore does not introduce a different global optimum; the per-episode regret motivates the decomposition into action suboptimality gaps given in Proposition 1.

D.2 Per-Episode Regret Decomposition

Proposition 1. *Assume rewards are bounded. For any $\pi \in \Pi_T$,*

$$\mathrm{PReg}_\rho(\pi; \Pi_T \mid \xi_0) = \mathbb{E}_{\tau \sim P_\rho^{\pi}(\cdot|\xi_0)}\left[\sum_{t=0}^{T-1} \gamma^t \, \mathrm{IReg}_{\rho,t}(\xi_t, a_t)\right] = -\mathbb{E}_{\tau \sim P_\rho^{\pi}(\cdot|\xi_0)}\left[\sum_{t=0}^{T-1} \gamma^t A_{\rho,t}^{\star}(\xi_t, a_t)\right].$$

Proof. For any nonterminal prefix ξ_t and action a_t, the Bellman equation gives

$$Q_{\rho,t}^{\star}(\xi_t, a_t) = \mathbb{E}_\rho\big[r_t^{\mathrm{step}} + \gamma V_{\rho,t+1}^{\star}(\xi_{t+1}) \mid \xi_t, a_t\big].$$

Since $A_{\rho,t}^{\star} = Q_{\rho,t}^{\star} - V_{\rho,t}^{\star}$ and $\mathrm{IReg}_{\rho,t} = -A_{\rho,t}^{\star}$, taking expectation under the trajectory law induced jointly by ρ and π gives

$$\begin{aligned} -\mathbb{E}_{\rho,\pi}[\gamma^t A_{\rho,t}^{\star}(\xi_t, a_t)] &= \mathbb{E}_{\rho,\pi}[\gamma^t V_{\rho,t}^{\star}(\xi_t)] - \mathbb{E}_{\rho,\pi}[\gamma^t r_t^{\mathrm{step}}] \\ &\quad - \mathbb{E}_{\rho,\pi}[\gamma^{t+1} V_{\rho,t+1}^{\star}(\xi_{t+1})]. \end{aligned}$$

Summing over $t = 0, \ldots, T-1$ telescopes in expectation. Since $V_{\rho,T}^{\star} = 0$,

$$-\mathbb{E}_{\tau \sim P_\rho^{\pi}}\left[\sum_{t=0}^{T-1} \gamma^t A_{\rho,t}^{\star}(\xi_t, a_t)\right] = V_{\rho,0}^{\star}(\xi_0) - V_{\rho,0}^{\pi}(\xi_0) = \mathrm{PReg}_\rho(\pi; \Pi_T \mid \xi_0).$$

Substituting $\mathrm{IReg}_{\rho,t} = -A_{\rho,t}^{\star}$ yields the claimed decomposition. □

Action-space form under deterministic environment outcomes. Suppose the observation and reward process is deterministic conditional on the task-conditioned complete prefix and action. Let $\mathcal{T}_T^{\mathrm{full}}(\xi_0)$ be the full feasible action-history tree rooted at ξ_0, $\mathcal{C}_T(\xi_0)$ its length-T action sequences, and $r_t(a_{0:t}) := \gamma^t r_t^{\mathrm{step}}(a_{0:t})$. For a realized sequence, define

$$\mathrm{PReg}(a_{0:T-1}; \mathcal{C}_T(\xi_0)) := \max_{a'_{0:T-1} \in \mathcal{C}_T(\xi_0)} \sum_{t=0}^{T-1} r_t(a'_{0:t}) - \sum_{t=0}^{T-1} r_t(a_{0:t}).$$

For any $\pi_\theta \in \Pi_T$, let $P_\rho^{\pi_\theta}(\cdot \mid \xi_0)$ denote the trajectory law induced jointly by the environment law ρ and the parsed-action policy π_θ, conditional on the initial prefix ξ_0. The policy-space and expected action-sequence forms are equivalent:

$$\begin{aligned} \mathrm{PReg}_\rho(\pi_\theta; \Pi_T \mid \xi_0) &= \mathrm{PReg}(\pi_\theta; \mathcal{C}_T(\xi_0)) \\ &:= \mathbb{E}_{\tau \sim P_\rho^{\pi_\theta}(\cdot|\xi_0)}[\mathrm{PReg}(a_{0:T-1}; \mathcal{C}_T(\xi_0))] \\ &= V_0^\star(\xi_0) - V_0^{\pi_\theta}(\xi_0). \end{aligned} \tag{6}$$

Proof. For the deterministic specialization, for any nonterminal prefix ξ_t, the deterministic transition induced by action a_t gives a next prefix ξ_{t+1} and immediate step reward $r_t^{\mathrm{step}}(a_{0:t})$. By definition of the discounted subtree value,

$$Q_t^\star(\xi_t, a_t) = r_t^{\mathrm{step}}(a_{0:t}) + \gamma V_{t+1}^\star(\xi_{t+1}).$$

Since $A_t^\star(\xi_t, a_t) = Q_t^\star(\xi_t, a_t) - V_t^\star(\xi_t)$, we also have $\mathrm{IReg}_t(\xi_t, a_t) = -A_t^\star(\xi_t, a_t)$. Hence

$$-\gamma^t A_t^\star(\xi_t, a_t) = \gamma^t V_t^\star(\xi_t) - \gamma^t r_t^{\mathrm{step}}(a_{0:t}) - \gamma^{t+1} V_{t+1}^\star(\xi_{t+1}).$$

Summing over $t = 0, \ldots, T-1$ telescopes:

$$-\sum_{t=0}^{T-1} \gamma^t A_t^\star(\xi_t, a_t) = V_0^\star(\xi_0) - \sum_{t=0}^{T-1} \gamma^t r_t^{\mathrm{step}}(a_{0:t}) - \gamma^T V_T^\star(\xi_T).$$

Since $V_T^\star(\xi_T) = 0$ and $\sum_{t=0}^{T-1} \gamma^t r_t^{\mathrm{step}}(a_{0:t}) = \sum_{t=0}^{T-1} r_t(a_{0:t})$, the right side is

$$V_0^\star(\xi_0) - \sum_{t=0}^{T-1} r_t(a_{0:t}).$$

By construction,

$$V_0^\star(\xi_0) = \max_{a'_{0:T-1} \in \mathcal{C}_T(\xi_0)} \sum_{t=0}^{T-1} r_t(a'_{0:t}),$$

so this equals $\mathrm{PReg}(a_{0:T-1}; \mathcal{C}_T(\xi_0))$. Substituting $\mathrm{IReg}_t = -A_t^\star$ gives the pathwise decomposition; taking expectation under π_θ gives Equation (6). □

Remark 1 (Interpretation and scope)**.** *In the deterministic case, Equation (6) identifies the full-class per-episode regret with expected action-sequence policy regret. The equivalence with expected-return maximization relies on the optimal comparator value being independent of the learned policy. If the comparator class depends on π_θ, its optimal value may also depend on π_θ, in which case minimizing comparator-relative regret need not be equivalent to maximizing expected return. In our formulation, policy dependence enters through the sampled estimator, not through the fixed comparator class. A reward-maximization method using the same $Q^\star$-based action-value recursion would likewise produce the same per-action signal $A_t^\star = Q_t^\star - V_t^\star$. Proposition 1 provides a stepwise decomposition of per-episode regret; it is not a policy-gradient theorem. Separately, Bellman optimality shows that a policy supported on zero-immediate-regret actions at every reachable prefix is optimal.* **REGPO** *uses sampled estimates of $A_t^\star = Q_t^\star - V_t^\star$ as auxiliary feedback for actions at selected prefixes, rather than as an unbiased estimator of the global per-episode-regret gradient. Our distinction from trajectory-level return-based advantage estimates and mean-return comparisons lies in how this signal is estimated from sampled rollout subtrees and used for credit assignment.*

D.3 Trajectory-Level Credit and Advantages of Actions at Prefixes

Proposition 2 (Trajectory broadcast cannot match advantages of actions at distinct prefixes pointwise)**.** *Consider any trajectory-broadcast rule that assigns the same scalar $B(\tau)$ to every prefix–action pair on a realized trajectory τ. There exist a depth-two history-based MDP with deterministic next-prefix and reward outcomes, a current policy π, and a trajectory $\tau = (\xi_0, a_0, \xi_1, a_1, \xi_2)$ such that τ has positive probability under π and*

$$A_0^\star(\xi_0, a_0) = 0, \qquad A_1^\star(\xi_1, a_1) = -1, \qquad A_0^\pi(\xi_0, a_0) = \frac{1}{3}, \qquad A_1^\pi(\xi_1, a_1) = -\frac{2}{3}.$$

Consequently, for every $B(\tau)$,

$$\max_{t\in\{0,1\}} |B(\tau) - A_t^\star(\xi_t, a_t)| \geq \frac{1}{2}, \qquad \max_{t\in\{0,1\}} |B(\tau) - A_t^\pi(\xi_t, a_t)| \geq \frac{1}{2},$$

and the average squared error over the two prefix–action pairs is at least $1/4$ for either target. These bounds hold even with noiseless rewards and arbitrarily many repeated samples of the same trajectory.

Proof. At the root node ξ_0, let action a_0 lead to a second prefix node ξ_1, and let the other root action terminate with return zero. At ξ_1, let one terminal action return one and let the realized action a_1 return zero. Then

$$A_0^\star(\xi_0, a_0) = 0, \qquad A_1^\star(\xi_1, a_1) = -1.$$

Choose a current policy satisfying

$$\pi(a_0 \mid \xi_0) = \frac{1}{2}, \qquad \pi(a_1 \mid \xi_1) = \frac{1}{3}.$$

The trajectory $(\xi_0, a_0, \xi_1, a_1, \xi_2)$ then has probability $1/6$. Moreover,

$$Q_0^\pi(\xi_0, a_0) = \frac{2}{3}, \qquad V_0^\pi(\xi_0) = \frac{1}{3}, \qquad Q_1^\pi(\xi_1, a_1) = 0, \qquad V_1^\pi(\xi_1) = \frac{2}{3},$$

so

$$A_0^\pi(\xi_0, a_0) = \frac{1}{3}, \qquad A_1^\pi(\xi_1, a_1) = -\frac{2}{3}.$$

Both target pairs differ by one. For either pair (x, y) and any scalar B, the triangle inequality gives

$$1 = |x - y| \leq |B - x| + |B - y|,$$

so at least one of the two absolute errors is at least $1/2$. Moreover,

$$\frac{1}{2}\left((B-x)^2 + (B-y)^2\right) = \left(B - \frac{x+y}{2}\right)^2 + \frac{1}{4} \geq \frac{1}{4}.$$

Repeatedly sampling the same trajectory changes neither fact, because the broadcast advantage estimate remains constant across its two prefix–action pairs. □

D.4 Tree-GRPO Comparisons among Actions at Prefixes

Weighted contribution at a prefix. For each terminal leaf ℓ of a sampled rollout tree, write R_ℓ for the return of the corresponding root-to-leaf trajectory. Tree-GRPO assigns that trajectory the intra-tree advantage estimate

$$\widehat{A}_\ell^{\text{intra}} = \frac{R_\ell - \overline{R}_{\widehat{\mathcal{T}}}}{s_{\widehat{\mathcal{T}}}},$$

where $\overline{R}_{\widehat{\mathcal{T}}}$ and $s_{\widehat{\mathcal{T}}}$ are the mean and standard deviation across the root-to-leaf trajectories in the sampled rollout tree. Tree-GRPO broadcasts $\widehat{A}_\ell^{\text{intra}}$ along the corresponding trajectory (Ji et al., 2026). The full estimator also includes an inter-tree advantage estimate; here we isolate the intra-tree component used for comparisons within a rollout tree.

Consider the intra-tree component's policy-gradient term at $\theta = \theta_{\text{old}}$, where clipping is locally inactive. Under the parsed-action abstraction, trajectories taking action a at the same prefix ξ share $\nabla_\theta \log \pi_\theta(a \mid \xi)$. Write Z_ℓ for the total number of generated tokens in the trajectory corresponding to leaf ℓ. Retaining the factor $1/Z_\ell$ from Equation (1), the contribution is

$$\left(\sum_{\ell \in \widehat{\mathcal{L}}(\xi,a)} \frac{\widehat{A}_\ell^{\text{intra}}}{Z_\ell} \right) \nabla_\theta \log \pi_\theta(a \mid \xi). \tag{7}$$

The coefficient is a weighted sum of intra-tree relative advantage estimates, or equivalently of returns after the within-tree normalization defining $\widehat{A}_\ell^{\text{intra}}$. Summing Equation (7) over the sampled actions gives the full contribution at the prefix. This expression accommodates different trajectory lengths and different numbers of sampled continuations. If ξ is the root of the sampled rollout tree used for within-tree normalization, every action has the same number of sampled continuations, and all Z_ℓ are equal, the difference between the grouped coefficients of any two actions is a positive multiple of the difference between their sample mean continuation returns. At a binary prefix, the update direction has the same sign as this difference. With independent continuations sampled under $\pi_{\theta_{\text{old}}}$, these sample means converge to the corresponding $Q^{\pi_{\theta_{\text{old}}}}$ values.

Relation to token-level optimization. The factorization above treats an environment action as one choice under the parsed-action policy. In the token-level PPO-style surrogate, each LM output contributes its own token-level log-policy gradients. LM outputs mapped to the same parsed action may use different token sequences, and the same parsed action may occur at different complete interaction prefixes. Their contributions therefore remain trajectory-specific unless they share the same log-policy gradient. The main analysis uses the parsed-action abstraction, while the implemented surrogate retains these token-level contributions.

Recursive mean and sampled-maximum backups. Early termination can produce different numbers of terminal leaves below the sampled actions at a prefix. A flat average over terminal leaves then weights actions according to their numbers of sampled descendants. For a prefix ξ with sampled children $\xi a_1, \ldots, \xi a_m$, the recursive mean-value estimator is

$$\widehat{V}_{\text{mean}}(\xi) = \frac{1}{m} \sum_{j=1}^{m} \widehat{V}_{\text{mean}}(\xi a_j), \qquad \widehat{V}_{\text{mean}}(\ell) = R_\ell$$

for a terminal leaf ℓ. The corresponding $\widehat{Q}_{\text{mean}}(\xi, a)$ averages the values of sampled children reached by action a, so each sampled prefix–action pair is counted once. When the sampled children are drawn under $\pi_{\theta_{\text{old}}}$, the estimator has expectation $Q^{\pi_{\theta_{\text{old}}}}$ by backward induction; consistency follows as the relevant numbers of sampled continuations grow.

The sampled-maximum estimator instead uses recursive maximum backups:

$$\widehat{A}_{\max}(\xi, a) = \underbrace{\max_{\ell \in \widehat{\mathcal{L}}(\xi,a)} R_\ell}_{\widehat{Q}_{\max}(\xi,a)} - \underbrace{\max_{\ell \in \widehat{\mathcal{L}}(\xi)} R_\ell}_{\widehat{V}_{\max}(\xi)}.$$

The distinction is therefore between prefix and action values computed by recursive averaging and those computed by recursive maximization.

In the Hard-Safe tree, before a reward-one leaf is sampled, every sampled Hard continuation has return zero. The recursive mean and sampled-maximum action values then agree at every Hard prefix, while the corresponding advantage estimates differ only by an action-independent shift at that prefix. The mean–max distinction becomes operative after a successful Hard continuation is discovered; earlier differences between implementations arise from other weighting or normalization choices.

D.5 Mean–Max Separation

Root update directions in the Hard-Safe tree. Let $s := \pi(\text{Hard} \mid \xi) \in (0, 1)$ and $z := \log(s/(1 - s))$. For fixed scalar update weights G_H and G_S, the population first-order direction of the PPO-

style surrogate at the old policy is

$$\mathbb{E}_{a\sim\pi}\left[G_a \frac{\partial}{\partial z}\log\pi(a \mid \xi)\right] = s(1-s)(G_H - G_S).$$

If the update weights are a positive affine normalization of $Q^\pi(\xi,\cdot)$, then $G_H - G_S = \alpha(p_\pi - c) < 0$ for some $\alpha > 0$ by Equation (4). Hence the direction decreases z and the probability of Hard.

For the exact regret-based advantage, the optimal parent value is one, so Equations (2) and (4) gives $A^\star(\xi, \text{Hard}) = 0$ and $A^\star(\xi, \text{Safe}) = c - 1$. Their difference is $1 - c > 0$, so the same calculation shows that the direction increases z and the probability of Hard. □

Proposition 3 (Finite-sample mean–max comparison)**.** *In the Hard-Safe tree, write $p := p_\pi < c$ and draw m independent Hard returns $X_1, \ldots, X_m \overset{\text{i.i.d.}}{\sim} \text{Bernoulli}(p)$. Because every Safe return equals $c \in (0,1)$,*

$$\mathbb{P}\left(\max_{1\le i\le m} X_i > c\right) = 1 - (1-p)^m,$$
$$\mathbb{P}\left(\frac{1}{m}\sum_{i=1}^m X_i > c\right) = \mathbb{P}(\text{Binomial}(m,p) > mc) \le \exp\left[-2m(c-p)^2\right].$$

Thus the recursive-maximum comparison ranks Hard above Safe with probability at least $1-\delta$ whenever

$$m \ge \frac{\log(1/\delta)}{-\log(1-p)} = O\left(\frac{1}{p}\log\frac{1}{\delta}\right),$$

whereas the probability that the sample-mean comparison ranks Hard above Safe decreases exponentially in m.

Proof of the finite-sample mean–max comparison. Let $S_m := \sum_{i=1}^m X_i \sim \text{Binomial}(m,p)$. Since $c \in (0,1)$,

$$\max_i X_i > c \iff \max_i X_i = 1 \iff S_m \ge 1.$$

Therefore

$$\mathbb{P}\left(\max_i X_i > c\right) = 1 - \mathbb{P}(S_m = 0) = 1 - (1-p)^m.$$

Similarly,

$$\frac{1}{m}\sum_{i=1}^m X_i > c \iff S_m/m > c.$$

Because $X_i \in [0,1]$ and $p < c$, Hoeffding's inequality gives

$$\mathbb{P}(S_m/m > c) \le \exp[-2m(c-p)^2].$$

Finally, $1-(1-p)^m \ge 1-\delta$ exactly when $(1-p)^m \le \delta$, which is guaranteed by

$$m \ge \frac{\log(1/\delta)}{-\log(1-p)}.$$

The stated asymptotic form follows from $-\log(1-p) = \Theta(p)$ as $p \downarrow 0$. □

Remark 2 (Condition for the Sampled Combined Advantage)**.** *In the Hard-Safe setting of Proposition 3, let $\overline{X}_m := m^{-1}\sum_{i=1}^m X_i$. The sampled mean and sampled maximum give the combined Hard-versus-Safe advantage difference*

$$\lambda(\overline{X}_m - c) + (1-\lambda)(\max_i X_i - c).$$

Conditional on sampling a successful Hard rollout, so that $\max_i X_i = 1$, the mixture favors Hard if and only if

$$\lambda < \frac{1-c}{1-\overline{X}_m}.$$

For $\lambda > 0$, using $w_{\text{reg}} = (1-\lambda)/\lambda$ makes this equivalent to $w_{\text{reg}} > (c - \overline{X}_m)/(1-c)$ under the additive convention. Because $\overline{X}_m \ge 0$, $\lambda < 1 - c$ is sufficient independently of the sampled mean. For $\lambda > 0$, the equivalent sufficient condition is $w_{\text{reg}} > c/(1-c)$.

E Regret as an Auxiliary Objective for Policy Optimization

Proposition 1 motivates an auxiliary loss that encourages actions with small immediate regret at each prefix. We show that this loss and the standard task loss admit a common optimal policy, and that their combination yields the exact combined advantage $A_t^{\pi_{\theta_{\text{old}}}} + w_{\text{reg}} A_t^\star$ in a gradient-descent update. Here $\pi_{\theta_{\text{old}}}$ is the old policy used to generate the rollout batch, as in Equation (1). This is the exact combined advantage corresponding to the estimate $\widehat{A}^{\text{RegPO}}$.

For the formal results below, use the finite-horizon history-based MDP and full comparator class Π_T from Proposition 1. Fix p, ρ, and $\xi_0 = (p, o_0)$, assume finite feasible prefix and action sets, bounded rewards, and $\gamma \in (0, 1]$, and let $\{\pi_\theta : \theta \in \Theta\} \subseteq \Pi_T$. Suppress ρ, and define Ξ_t as the set of prefixes reachable at step t with positive probability under the trajectory law induced jointly by ρ and at least one policy in Π_T. Also define $d_t^\theta(\xi) := \mathbb{P}_{\rho,\pi_\theta}(\xi_t = \xi \mid \xi_0)$ and $d_t^{\text{old}} := d_t^{\theta_{\text{old}}}$. Let $\mathcal{L}_1(\theta) := V_0^\star(\xi_0) - V_0^{\pi_\theta}(\xi_0)$, take $\mathcal{L}_2(\theta; \theta_{\text{old}})$ from Equation (3), and write $\mathcal{L}(\theta; \theta_{\text{old}}) := \mathcal{L}_1(\theta) + w_{\text{reg}} \mathcal{L}_2(\theta; \theta_{\text{old}})$.

E.1 A Common Optimum

Proposition 4 (Common minimizer of the task and auxiliary losses)**.** *Suppose $\pi^\star = \pi_{\theta^\star}$ belongs to the parameterized policy class and, at every feasible prefix $\xi \in \Xi_t$,*

$$\operatorname{supp} \pi_t^\star(\cdot \mid \xi) \subseteq \arg\max_a A_t^\star(\xi, a). \tag{8}$$

Then $\theta^\star$ simultaneously minimizes $\mathcal{L}_1$ and $\mathcal{L}_2(\cdot; \theta_{\text{old}})$, both with value zero. For every $w_{\text{reg}} > 0$,

$$\begin{aligned} \min_{\theta \in \Theta} \mathcal{L}(\theta; \theta_{\text{old}}) &= 0, \\ \arg\min_{\theta \in \Theta} \mathcal{L}(\theta; \theta_{\text{old}}) &= \{\theta \in \Theta : \mathcal{L}_1(\theta) = 0,\ \mathcal{L}_2(\theta; \theta_{\text{old}}) = 0\}. \end{aligned} \tag{9}$$

Consequently, every global minimizer of the combined loss attains optimal task return. If additionally $d_t^{\text{old}}(\xi) > 0$ at every feasible prefix, every global minimizer of the auxiliary loss alone also attains optimal task return.

Proof. Both losses are nonnegative. Under Equation (8), every selected action has zero immediate regret, so $\mathcal{L}_2(\theta^\star; \theta_{\text{old}}) = 0$. By Proposition 1, the policy is optimal and $\mathcal{L}_1(\theta^\star) = 0$. Their positive weighted sum therefore has minimum zero, attained exactly when both losses are zero. This proves Equation (9).

Under the additional coverage condition, zero auxiliary loss forces $\pi_{\theta,t}(a \mid \xi)\, \text{IReg}_t(\xi, a) = 0$ at every feasible prefix and action, since $\gamma^t d_t^{\text{old}}(\xi) > 0$. The policy selects only zero-regret actions at every prefix and is optimal by Proposition 1. □

The losses need not have identical sets of minimizers: an optimal policy may take suboptimal actions at prefixes it never visits, and the auxiliary loss may penalize those actions. For tabular softmax policies, deterministic-policy optima may be attained only in the closure of the policy class. The minimum-zero statement then applies to that closure, with infimum zero along a sequence of policies approaching the optimum.

E.2 The Gradient and Its Fixed-Batch Surrogates

Assume π_θ is differentiable near θ_{old} and $\pi_{\theta_{\text{old}},t}(a \mid \xi) > 0$ for every feasible action at prefixes with $d_t^{\text{old}}(\xi) > 0$. In the gradient identities below, derivatives are with respect to θ and evaluated at $\theta = \theta_{\text{old}}$. The old policy, prefix weights, and advantage functions remain fixed.

Proposition 5 (The exact combined advantage as a negative gradient of the combined loss)**.** *At $\theta = \theta_{\text{old}}$,*

$$-\nabla_\theta \mathcal{L}(\theta; \theta_{\text{old}}) = \mathbb{E}_{\rho,\pi_{\theta_{\text{old}}}} \left[\sum_{t=0}^{T-1} \gamma^t \nabla_\theta \log \pi_{\theta,t}(a_t \mid \xi_t) \left(A_t^{\pi_{\theta_{\text{old}}}} + w_{\text{reg}} A_t^\star \right) \right]. \tag{10}$$

Thus the policy-gradient estimator using the exact combined advantage is unbiased for the negative gradient of the combined loss at the old policy.

Proof. The policy-gradient identity gives, at $\theta = \theta_{\text{old}}$,

$$-\nabla_\theta \mathcal{L}_1(\theta) = \mathbb{E}_{\rho,\pi_{\theta_{\text{old}}}}\left[\sum_{t=0}^{T-1} \gamma^t \nabla_\theta \log \pi_{\theta,t}(a_t \mid \xi_t) A_t^{\pi_{\theta_{\text{old}}}}(\xi_t, a_t)\right].$$

This follows by differentiating the trajectory likelihood, weighting each log-policy gradient by its return-to-go, and taking the conditional expectation $Q_t^{\pi_{\theta_{\text{old}}}}$. Subtracting the action-independent baseline $V_t^{\pi_{\theta_{\text{old}}}}$ leaves the expectation unchanged. For the auxiliary term, direct differentiation gives

$$\begin{aligned}-\nabla_\theta \mathcal{L}_2(\theta;\theta_{\text{old}}) &= \sum_{t,\xi} \gamma^t d_t^{\text{old}}(\xi) \sum_a \nabla_\theta \pi_{\theta,t}(a \mid \xi) A_t^\star(\xi, a) \\ &= \mathbb{E}_{\rho,\pi_{\theta_{\text{old}}}}\left[\sum_{t=0}^{T-1} \gamma^t \nabla_\theta \log \pi_{\theta,t}(a_t \mid \xi_t) A_t^\star(\xi_t, a_t)\right],\end{aligned}$$

where the last equality is evaluated at $\theta = \theta_{\text{old}}$. Adding the two identities with weight w_{reg} proves Equation (10). □

Weighted log-probability surrogate. To implement this direction, maximize the surrogate

$$\mathcal{J}_{\log}(\theta;\theta_{\text{old}}) := \mathbb{E}_{\rho,\pi_{\theta_{\text{old}}}}\left[\sum_{t=0}^{T-1} \gamma^t \left(A_t^{\pi_{\theta_{\text{old}}}}(\xi_t, a_t) + w_{\text{reg}} A_t^\star(\xi_t, a_t)\right) \log \pi_{\theta,t}(a_t \mid \xi_t)\right].$$

With the advantage terms held fixed, at $\theta = \theta_{\text{old}}$ we have

$$\nabla_\theta \mathcal{J}_{\log}(\theta;\theta_{\text{old}}) = -\nabla_\theta \mathcal{L}(\theta;\theta_{\text{old}}).$$

Replacing the log probability by $\pi_{\theta,t}(a_t \mid \xi_t)/\pi_{\theta_{\text{old}},t}(a_t \mid \xi_t)$ gives the same derivative at the old policy. Under the support condition above, the auxiliary component of this unclipped ratio surrogate equals $-\mathcal{L}_2(\theta;\theta_{\text{old}})$ for every policy π_θ being optimized.

E.3 The Task-Loss Gradient Using Future Regret-Based Advantages

The distinction between $\mathcal{L}_1$ and $\mathcal{L}_2$ can also be made explicit without differentiating prefix weights. Continue to assume a finite horizon, differentiable policies with positive action probabilities, bounded rewards, and a fixed environment law and comparator. All log-policy gradients below are evaluated at $\theta = \theta_{\text{old}}$.

Proposition 6 (Future regret-based advantages in the return gradient)**.** *Under these assumptions,*

$$-\nabla_\theta \mathcal{L}_1(\theta) = \mathbb{E}_{\rho,\pi_{\theta_{\text{old}}}}\left[\sum_{t=0}^{T-1} \gamma^t \nabla_\theta \log \pi_{\theta,t}(a_t \mid \xi_t) \sum_{k=t}^{T-1} \gamma^{k-t} A_k^\star(\xi_k, a_k)\right],$$

$$\mathbb{E}_{\rho,\pi_{\theta_{\text{old}}}}\left[\sum_{k=t}^{T-1} \gamma^{k-t} A_k^\star(\xi_k, a_k) \mid \xi_t, a_t\right] = Q_t^{\pi_{\theta_{\text{old}}}}(\xi_t, a_t) - V_t^\star(\xi_t). \tag{11}$$

Consequently, replacing the immediate $A_t^\star$ term in the exact combined advantage with its complete discounted future sum gives $-(1+w_{\text{reg}})\nabla\mathcal{L}_1$ in expectation, rather than the combined-loss gradient.

Proof. The regret decomposition gives $-\mathcal{L}_1(\theta) = \mathbb{E}_{\rho,\pi_\theta} \sum_k \gamma^k A_k^\star$. The functions $A_k^\star$ are independent of θ, so differentiating the trajectory likelihood gives the log-policy gradient multiplied by this sum. Terms with $k < t$ vanish after conditioning on ξ_t, since the conditional mean log-policy gradient is zero. This proves the first identity. For the second, use $A_k^\star = \mathbb{E}_\rho[r_k^{\text{step}} + \gamma V_{k+1}^\star \mid \xi_k, a_k] - V_k^\star$ and telescope in conditional expectation from t onward. The result is $Q_t^{\pi_{\theta_{\text{old}}}} - V_t^\star$. Its difference from $A_t^{\pi_{\theta_{\text{old}}}}$ is the action-independent baseline $V_t^{\pi_{\theta_{\text{old}}}} - V_t^\star$, which leaves the policy-gradient expectation unchanged. Adding the base policy-gradient term therefore gives the stated factor $1 + w_{\text{reg}}$. □

Numerical verification. We evaluate Equation (11) using expected-update dynamics on the finite tree-structured MDPs studied in Sections 4.1.1 and 4.1.2. After accounting for the factor $1 + w_{\mathrm{reg}}$, updates using the complete discounted future sum agree with those using A^π alone. Across ten Hard-Safe depths, their censoring outcomes match in every case, and their uncensored times to $V^\pi(\xi_0) = 0.95$ agree to relative error at most 2.2×10^{-14}. Across all 240 random-tree instances, their censoring outcomes again match, and their uncensored times to the 95% criterion in Section 4.1.2 agree to relative error at most 1.0×10^{-10}. This agreement numerically verifies the distinction in Equation (11): the complete future sum recovers $-\nabla\mathcal{L}_1$, whereas the immediate $A_t^\star$ term contributes $-\nabla\mathcal{L}_2$.

E.4 Tabular Updates of the Auxiliary Loss

Continue with the finite-horizon history-based MDP, finite feasible prefix and action sets, and bounded rewards from Section E. In this subsection, the parameters are the action probabilities themselves: $\theta_{t,\xi,a} = \pi_{\theta,t}(a \mid \xi)$. Let Θ be the set of all feasible vectors of these probabilities, corresponding to Π_T. With the old-policy prefix weights fixed, $\mathcal{L}_2(\theta; \theta_{\mathrm{old}})$ is linear, hence convex, in θ, and each prefix contributes a separate term. In $\mathcal{L}_1(\theta)$, the prefix probabilities also depend on θ, so the task loss is generally not convex. This statement concerns action probabilities; it does not assert convexity in softmax logits or neural network parameters.

The next result relates gradient descent on $\mathcal{L}_2$ to task return. Write $\pi^{(k)} := \pi_{\theta^{(k)}}$ for the policy at update k; the superscript (k) counts gradient-descent updates, while the subscript t in $\pi_t^{(k)}(a \mid \xi)$ denotes the interaction step. At update k, use $\pi^{(k)}$ as the old policy and take a gradient-descent step on $\mathcal{L}_2(\theta; \theta^{(k)})$. Hold its prefix weights fixed while differentiating, and then refresh them after the update:

$$g^{(k)} := \nabla_\theta \mathcal{L}_2(\theta; \theta^{(k)})\Big|_{\theta=\theta^{(k)}}, \qquad \theta^{(k+1)} := \arg\min_{\vartheta \in \Theta} \|\vartheta - (\theta^{(k)} - \eta g^{(k)})\|_2^2. \tag{12}$$

Thus each gradient-descent step is followed by choosing the nearest feasible vector of action probabilities.

Proposition 7 (Task return under tabular auxiliary updates)**.** *Use exact $A_t^\star$ and exact prefix probabilities in Equation (12), starting from any $\theta^{(1)} \in \Theta$. Let $D, G > 0$ satisfy $\|\theta - \vartheta\|_2 \le D$ for all $\theta, \vartheta \in \Theta$ and $\|g^{(k)}\|_2 \le G$ for all k. Such finite bounds exist under the assumptions above. For any $N \ge 1$ and constant step size $\eta > 0$,*

$$\frac{1}{N}\sum_{k=1}^{N}\left[V_0^\star(\xi_0) - V_0^{\pi^{(k)}}(\xi_0)\right] \le \frac{D^2}{2\eta N} + \frac{\eta G^2}{2}. \tag{13}$$

In particular, choosing $\eta = D/(G\sqrt{N})$ makes the right-hand side $DG/\sqrt{N}$. The same bound holds for $V_0^\star(\xi_0) - \max_{1\le k\le N} V_0^{\pi^{(k)}}(\xi_0)$.

Proof. Choose $\theta^\star \in \Theta$ whose policy is optimal at every feasible prefix. Linearity, Proposition 1, and the zero auxiliary loss of this policy give

$$\begin{aligned}\langle g^{(k)}, \theta^{(k)} - \theta^\star\rangle &= \mathcal{L}_2(\theta^{(k)}; \theta^{(k)}) - \mathcal{L}_2(\theta^\star; \theta^{(k)}) \\ &= \mathcal{L}_1(\theta^{(k)}) = V_0^\star(\xi_0) - V_0^{\pi^{(k)}}(\xi_0).\end{aligned} \tag{14}$$

Since Θ is closed and convex and $\theta^\star \in \Theta$, choosing the nearest feasible vector in Equation (12) gives

$$\begin{aligned}\|\theta^{(k+1)} - \theta^\star\|_2^2 &\le \|\theta^{(k)} - \eta g^{(k)} - \theta^\star\|_2^2 \\ &\le \|\theta^{(k)} - \theta^\star\|_2^2 - 2\eta\mathcal{L}_1(\theta^{(k)}) + \eta^2 G^2.\end{aligned}$$

Summing from $k = 1$ to N, cancelling consecutive squared distances, and using $\|\theta^{(1)} - \theta^\star\|_2^2 \le D^2$ proves Equation (13). The smallest task loss among these policies is at most their average task loss. □

Comparison with task-loss updates. Although $\mathcal{L}_1$ is generally not convex, its gradient updates can still converge to an optimal policy. Agarwal, Kakade, Lee, and Mahajan (2021) establish global convergence of tabular policy-gradient methods under their stated assumptions. In particular, their ordinary softmax result assumes that the distribution of initial states used for optimization assigns positive probability to every state (Agarwal et al., 2021, Theorem 10). Here, linearity of the auxiliary loss gives the direct equality in Equation (14) and the short proof above. This argument is closely related to the analysis of repeated policy updates using optimal action values in Sun et al. (2017).

With $\eta = D/(G\sqrt{N})$, gradient descent on $\mathcal{L}_2$ with refreshed prefix weights gives average task loss at most $DG/\sqrt{N}$, so at least one of the first N policies satisfies the same bound. Individual updates can still decrease task return. Sections F and G provide the learning-time comparisons with task-loss updates.

E.5 Approximate Auxiliary Loss and Task Performance

The next result applies to the finite-horizon history-based MDP. Retain $\pi^\star = \pi_{\theta^\star}$ from Proposition 4, optimal at every feasible prefix. At every prefix with positive d_t^{old}, suppose estimates are defined for all feasible actions and satisfy

$$\max_a |\widehat{Q}_t(\xi, a) - Q_t^\star(\xi, a)| \le \varepsilon_t(\xi).$$

Set $\widehat{A}_t(\xi, a) = \widehat{Q}_t(\xi, a) - \widehat{V}_t(\xi)$, with any fixed action-independent baseline, and define $\widehat{\mathcal{L}}_2$ by replacing $A_t^\star$ with $\widehat{A}_t$ in Equation (3). Write

$$E_{\mathrm{old}} := \sum_{t,\xi} \gamma^t d_t^{\mathrm{old}}(\xi)\varepsilon_t(\xi), \qquad \widehat{\mathcal{L}} := \mathcal{L}_1 + w_{\mathrm{reg}}\widehat{\mathcal{L}}_2.$$

The estimates, their error bounds, and the old-policy prefix weights are fixed in these optimization problems.

Proposition 8 (Approximate auxiliary loss and optimal return)**.** *For any $\delta \ge 0$, the following conclusions hold. If $\widehat{\mathcal{L}}(\theta) \le \inf_{\vartheta\in\Theta} \widehat{\mathcal{L}}(\vartheta) + \delta$, then*

$$\mathcal{L}_1(\theta) + w_{\mathrm{reg}}\mathcal{L}_2(\theta; \theta_{\mathrm{old}}) \le \delta + 2w_{\mathrm{reg}}E_{\mathrm{old}}.$$

Alternatively, if $\widehat{\mathcal{L}}_2(\theta) \le \inf_{\vartheta\in\Theta} \widehat{\mathcal{L}}_2(\vartheta) + \delta$ and $d_t^\theta(\xi) \le C d_t^{\mathrm{old}}(\xi)$ at every feasible prefix for some finite $C \ge 1$, then

$$\mathcal{L}_1(\theta) \le C\,[\delta + 2E_{\mathrm{old}}].$$

Proof. Let $e_t(\xi, a) = \widehat{Q}_t(\xi, a) - Q_t^\star(\xi, a)$. Action-independent baselines cancel when comparing two policies under the same fixed prefix weights. Since $\mathcal{L}_2(\theta^\star; \theta_{\mathrm{old}}) = 0$,

$$\begin{aligned}\mathcal{L}_2(\theta; \theta_{\mathrm{old}}) &= \widehat{\mathcal{L}}_2(\theta) - \widehat{\mathcal{L}}_2(\theta^\star) \\ &\quad + \sum_{t,\xi} \gamma^t d_t^{\mathrm{old}}(\xi) \sum_a [\pi_{\theta,t}(a \mid \xi) - \pi_t^\star(a \mid \xi)] e_t(\xi, a) \\ &\le \widehat{\mathcal{L}}_2(\theta) - \widehat{\mathcal{L}}_2(\theta^\star) + 2E_{\mathrm{old}}.\end{aligned}$$

The final bound follows because the absolute difference of the expectations of a function bounded by $\varepsilon_t(\xi)$ is at most $2\varepsilon_t(\xi)$. For the combined loss, use $\mathcal{L}_1(\theta^\star) = 0$ and $\widehat{\mathcal{L}}(\theta) - \widehat{\mathcal{L}}(\theta^\star) \le \delta$. For the auxiliary loss alone, the same comparison gives $\mathcal{L}_2(\theta; \theta_{\mathrm{old}}) \le \delta + 2E_{\mathrm{old}}$. Finally, nonnegative immediate regrets and the assumed prefix-distribution bound imply $\mathcal{L}_1(\theta) \le C\mathcal{L}_2(\theta; \theta_{\mathrm{old}})$. □

F Learning Dynamics under Combined Feedback

F.1 Combined Update at a Binary Prefix

Proposition 9 (Policy-gradient contribution at a binary prefix)**.** *Fix a prefix with two actions a, b, a frozen current policy π, and $p = \pi(a \mid \xi) = \sigma(z) \in (0,1)$. Set $\Delta_\pi = Q^\pi(\xi,a) - Q^\pi(\xi,b)$, $\Delta_\star = Q^\star(\xi,a) - Q^\star(\xi,b)$, and $\widehat{\Delta} = \widehat{Q}(\xi,a) - \widehat{Q}(\xi,b)$. The conditional expected policy-gradient contribution to the logit z using $A^\pi + w_{\mathrm{reg}}\widehat{A}$ is $p(1-p)\big[\Delta_\pi + w_{\mathrm{reg}}\widehat{\Delta}\big]$. If $|\widehat{Q}(\xi,x) - Q^\star(\xi,x)| \le \varepsilon$ for $x = a, b$, then $|\widehat{\Delta} - \Delta_\star| \le 2\varepsilon$. Thus $\Delta_\pi + w_{\mathrm{reg}}(\Delta_\star - 2\varepsilon) > 0$ suffices for a positive logit update and an increase in p.*

Proof. For the binary logit, $\partial_z \log \pi(a \mid \xi) = 1 - p$ and $\partial_z \log \pi(b \mid \xi) = -p$. Taking the conditional expectation over the sampled action and canceling the action-independent baselines in the two advantages gives the stated contribution. The error bound follows from the triangle inequality, and hence $\widehat{\Delta} \ge \Delta_\star - 2\varepsilon$, which gives the sufficient condition. □

In the setting of Proposition 9, suppose $\Delta_\pi < 0 < \Delta_\star$ and $\Delta_\pi \le \widehat{\Delta} \le \Delta_\star$. At a matched coefficient scale, the policy-gradient contribution using $\lambda A^\pi + (1-\lambda)\widehat{A}$, $0 \le \lambda \le 1$, differs from the contribution using A^π alone by $p(1-p)(1-\lambda)(\widehat{\Delta} - \Delta_\pi) \ge 0$. Thus an isolated logit step gives at least as much probability to a as the same-size step using A^π alone, strictly more if $\lambda < 1$ and $\widehat{\Delta} > \Delta_\pi$. Even $\widehat{\Delta} \le 0$ can reduce suppression; an actual increase instead requires $\lambda\Delta_\pi + (1-\lambda)\widehat{\Delta} > 0$. Here λ is a normalized mixture weight. For $\lambda > 0$, this mixture has the same direction as $A^\pi + w_{\mathrm{reg}}\widehat{A}$, where $w_{\mathrm{reg}} = (1-\lambda)/\lambda$.

F.2 The Combined Loss in the Hard-Safe Tree

Use $p = \pi_\theta(\mathrm{Hard} \mid \xi)$ and $q_k = \pi_\theta(a_k^\star \mid \xi_k)$, and write $p_{\mathrm{old}}, q_{k,\mathrm{old}}$ for their old-policy values. The old-policy visitation probability of prefix ξ_k is $d_k^{\mathrm{old}}(\xi_k) = p_{\mathrm{old}} \prod_{i<k} q_{i,\mathrm{old}}$. Since the optimal return is one and $\gamma = 1$, the losses are

$$\mathcal{L}_1(\theta) = (1-p)(1-c) + p\left(1 - \prod_{k=1}^{H} q_k\right), \tag{15}$$

$$\mathcal{L}_2(\theta;\theta_{\mathrm{old}}) = (1-p)(1-c) + \sum_{k=1}^{H} d_k^{\mathrm{old}}(\xi_k)(1-q_k). \tag{16}$$

The first loss depends on completing the entire Hard sequence. The second assigns an auxiliary loss to each reached decision, with its old-policy prefix weight held fixed. These expressions agree at the old policy by telescoping the failure probabilities, but their derivatives differ.

For $\lambda \in [0,1]$ and independent logits $p = \sigma(z)$ and $q_k = \sigma(\phi_k)$, differentiation of the normalized combined loss at the old policy gives

$$-\partial_z\big(\lambda\mathcal{L}_1 + (1-\lambda)\mathcal{L}_2\big) = p(1-p)\left[\lambda(p_\pi - c) + (1-\lambda)(1-c)\right], \tag{17}$$

$$-\partial_{\phi_k}\big(\lambda\mathcal{L}_1 + (1-\lambda)\mathcal{L}_2\big) = d_k^{\mathrm{old}}(\xi_k)q_k(1-q_k)\left[\lambda\prod_{i>k} q_i + 1 - \lambda\right]. \tag{18}$$

An empty product is one. The auxiliary contribution at step k is not multiplied by the probability of completing the remaining suffix after step k, but it is still multiplied by the probability of reaching ξ_k and by the action-probability factor at step k. Its magnitude therefore still depends on the old-policy visitation probability of ξ_k. The root expression recovers the main-text condition for favoring Hard. For $\lambda > 0$, the normalized mixture is proportional to the additive convention with $w_{\mathrm{reg}} = (1-\lambda)/\lambda$.

The combined update need not improve current task return. For $H = 1$, write $q = q_1$ and $V^\pi = (1-p)c + pq$. Under continuous-time ordinary logit-gradient updates using the normalized mixture $\lambda A^\pi + (1-\lambda)A^\star$ and refreshed prefix weights, Equations (17) and (18) gives

$$\dot{p} = [p(1-p)]^2[\lambda(q-c) + (1-\lambda)(1-c)], \qquad \dot{q} = p[q(1-q)]^2.$$

Hence

$$\frac{d}{dt}V^\pi = (q-c)\dot{p} + p\dot{q}.$$

At $p = 0.5$, $q = 0.05$, $c = 0.6$, and $\lambda = 1/3$, both $\dot{p}$ and $\dot{q}$ are positive, but $dV^\pi/dt < 0$. The increased probability of entering the currently unsuccessful Hard branch initially lowers task return more than the downstream improvement raises it. This behavior is consistent with the common optimum.

F.3 Expected-Update Dynamics in the Hard-Safe Tree

The expected-update dynamics are defined in Equation (5). They use $\gamma = 1$ and the normalized mixture weight $\lambda \in [0, 1]$ from Section 3.2. At each learning time, the old policy determines the prefix weights and advantage estimates held fixed within that gradient evaluation. These dynamics are the negative gradients of $\lambda\mathcal{L}_1 + (1-\lambda)\mathcal{L}_2$ evaluated at the old policy, using Equations (15) and (16). The old policy and hence this loss are refreshed as learning proceeds. The results below establish the claims in Theorem 2 and give their explicit bounds.

Proposition 10 (Learning time with exact action-value differences)**.** *Suppose $p(0) = p_0 \in (0, 1)$ and $q_k(0) = q_0 \in (0, 1)$ for every k, where c, p_0, q_0, λ are fixed independently of H. For $0 < \epsilon < 1 - c$, let $T_\lambda(\epsilon)$ be the first learning time at which $V^\pi = (1-p)c + pp_\pi \geq 1 - \epsilon$. If $0 \leq \lambda \leq 1 - c$, then p and every q_k increase, and*

$$T_\lambda(\epsilon) \leq \frac{2H^2}{p_0(1-\lambda)(1-\epsilon/2)q_0^2\epsilon} + \frac{2}{p_0^2(1-c-\epsilon/2)\epsilon} = O(H^2/\epsilon). \tag{19}$$

In contrast, the expected updates using A^π alone ($\lambda = 1$) satisfy, whenever $q_0^H < c$,

$$T_1(\epsilon) \geq \frac{q_0^{-H} - c^{-1}}{H}.$$

The upper bound includes pure $A^\star$ feedback at $\lambda = 0$.

Proof. The action-value difference in the root update equals $\lambda p_\pi + 1 - \lambda - c$ and is positive when $\lambda \leq 1 - c$. Thus $p \geq p_0$ and every q_k increases. Converting the Hard-step dynamics in Equation (5) to probability coordinates gives

$$\dot{q}_k = p\left(\prod_{j<k} q_j\right) q_k^2(1-q_k)^2\left[\lambda\prod_{j>k} q_j + 1 - \lambda\right].$$

Set $b = 1 - \epsilon/(2H)$. Analyze successive times at which the first k optimal-action probabilities have all reached b; all coordinates continue to update simultaneously. Once the first $k-1$ coordinates reach b,

$$\prod_{j<k} q_j \geq b^{k-1} \geq 1 - \epsilon/2, \qquad \dot{q}_k \geq p_0(1-\lambda)(1-\epsilon/2)q_0^2(1-q_k)^2.$$

Integrating until $q_k \geq b$ takes at most $2H/[p_0(1-\lambda)(1-\epsilon/2)q_0^2\epsilon]$ additional time. If that coordinate has already reached b, the additional time is zero. Summing over H coordinates proves the first term in Equation (19). At that time $p_\pi \geq b^H \geq 1 - \epsilon/2$, and thereafter the root action-value difference is at least $1 - c - \lambda\epsilon/2 \geq 1 - c - \epsilon/2 > 0$. Hence

$$\dot{p} \geq p_0^2(1-c-\epsilon/2)(1-p)^2.$$

Reaching $p \geq 1 - \epsilon/2$ takes at most the second term in the upper bound, after which $V^\pi \geq pp_\pi \geq (1-\epsilon/2)^2 \geq 1 - \epsilon$.

For $\lambda = 1$, symmetric initialization preserves $q_k = q$ at all learning times, and $\dot{q} = pq^{H+1}(1-q)^2 \leq q^{H+1}$. Before q^H reaches c, expected return is at most $c < 1 - \epsilon$. Therefore

$$T_1(\epsilon) \geq \int_{q_0}^{c^{1/H}} q^{-H-1}\,dq = \frac{q_0^{-H} - c^{-1}}{H}.$$

□

Estimated gaps rather than accurate absolute values. For $Q = Q^\pi, Q^\star, \widehat{Q}$, respectively, define $\Delta_\pi, \Delta_\star, \widehat{\Delta}$ as $Q(\xi, \mathrm{Hard}) - Q(\xi, \mathrm{Safe})$ at the root and $Q(\xi_k, a_k^\star) - Q(\xi_k, \bar{a}_k)$ at Hard step k, consistently with Proposition 9. The next result requires progress in these comparisons, not a small absolute error for either action value.

Proposition 11 (Estimated-gap comparison throughout learning)**.** *Compare the A^π-only expected-update dynamics with the expected-update dynamics using $\lambda A^\pi + (1-\lambda)\widehat{A}$, $0 \le \lambda \le 1$, from the same initialization. Suppose the estimated-feedback dynamics exist for all finite learning times and, at every decision and learning time, $\widehat{\Delta} \ge \Delta_\pi$. Then, at every common learning time, using superscripts A^π and* est *for the A^π-only and estimated-feedback dynamics, respectively,*

$$p^{\mathrm{est}} \ge p^{A^\pi}, \qquad q_k^{\mathrm{est}} \ge q_k^{A^\pi} \quad (1 \le k \le H). \tag{20}$$

Consequently, the probability of reaching each Hard step is at least as large, and any expected return threshold above c is reached no later than under the updates using A^π alone. In particular, the conclusion holds when $\Delta_\pi \le \widehat{\Delta} \le \Delta_\star$ throughout learning, without requiring the estimated root gap to be positive or close to $\Delta_\star$.

Proof. Let $x = (z, \phi_1, \dots, \phi_H)$ and write F^{A^π} for the A^π-only logit vector field. Its components are

$$F_0^{A^\pi} = p(1-p)(p_\pi - c), \qquad F_k^{A^\pi} = pp_\pi(1-q_k).$$

Each component is nondecreasing in all coordinates other than its own: the corresponding derivatives are $p(1-p)p_\pi(1-q_j)$ at the root, $p(1-p)p_\pi(1-q_k)$ for the root coordinate at Hard step k, and $pp_\pi(1-q_k)(1-q_j)$ for $j \ne k$ at Hard step k. The gap condition and positive prefix weights imply $\dot{x}^{\mathrm{est}} \ge F^{A^\pi}(x^{\mathrm{est}})$ componentwise.

For completeness, fix any finite time interval and consider $y(\tau) = x^{\mathrm{est}}(\tau) + \delta e^{K\tau}\mathbf{1}$, with $\delta > 0$ and K larger than a Lipschitz constant of F^{A^π} in the maximum norm. Then $\dot{y} > F^{A^\pi}(y)$ and $y(0) > x^{A^\pi}(0)$. At a first coordinate contact with the A^π-only solution, equality in that coordinate and the ordering of all other coordinates imply $F_i^{A^\pi}(y) \ge F_i^{A^\pi}(x^{A^\pi})$. The strict derivative inequality rules out that contact. Letting δ decrease to zero proves $x^{\mathrm{est}} \ge x^{A^\pi}$ and hence Equation (20). At any time the return under the A^π-only updates reaches $1 - \epsilon > c$, its probability of completing the Hard sequence satisfies $p_\pi^{A^\pi} > c$. The return $(1-p)c + pp_\pi$ is increasing in both p and p_π in this region, so the return of the policy updated using $\lambda A^\pi + (1-\lambda)\widehat{A}$ is then at least as large. This proves the hitting-time comparison. It does not order current returns at earlier times when Hard still performs worse than Safe. □

The condition applies at every decision throughout learning. Under independent logits and matched coefficient scales, it yields the componentwise comparison above. The next corollary strengthens this comparison to a polynomial upper bound under a uniform quantitative condition.

Corollary 1 (Learning time with estimated action-value differences)**.** *Retain the setup and initialization of Proposition 10, but use $\lambda A^\pi + (1-\lambda)\widehat{A}$ with the exact current-policy advantage A^π and estimated regret-based advantage $\widehat{A}$. Suppose the resulting expected-update dynamics exist for all finite learning times and, at every learning time and every decision prefix,*

$$\widehat{\Delta} \ge (1-\beta)\Delta_\pi + \beta\Delta_\star, \qquad \beta \in (0, 1], \qquad (1-\lambda)\beta \ge c, \tag{21}$$

where β is fixed independently of H. Then the upper bound in Equation (19) holds with its first denominator's factor $1-\lambda$ replaced by $(1-\lambda)\beta$.

Proof. At the root, the action-value difference used by the normalized mixture is at least

$$\big[1 - (1-\lambda)\beta\big]p_\pi + (1-\lambda)\beta - c \ge 0.$$

At Hard step k, this action-value difference is at least

$$\big[1 - (1-\lambda)\beta\big]\prod_{j>k} q_j + (1-\lambda)\beta \ge (1-\lambda)\beta.$$

Thus $p \ge p_0$ and all q_k increase. The continuation proof applies with $1-\lambda$ replaced by $(1-\lambda)\beta$. Once $p_\pi \ge 1 - \epsilon/2$, the root action-value difference is at least $1 - c - \epsilon/2$, so its bound is unchanged. □

Proof of Theorem 2. The comparison of the root and continuation probabilities, and hence of the probabilities of reaching the Hard steps and hitting times above c, follows from Proposition 11. The polynomial bound under the stronger condition follows from Corollary 1. □

At the root, the local ordering $\Delta_\pi \le \widehat{\Delta} \le \Delta_\star$ gives

$$p(1-p)(1-\lambda)(\widehat{\Delta} - \Delta_\pi) \ge 0.$$

This condition is weaker than Equation (21). It can give a strictly better one-step logit update while the estimated gap is still negative. The polynomial bound additionally requires a uniform positive β throughout learning and the weight condition. No upper bound on $\widehat{\Delta}$ is needed for the corollary, and common errors added to both action values cancel from its assumption. For $\lambda > 0$, the condition reads $[w_{\mathrm{reg}}/(1+w_{\mathrm{reg}})]\beta \ge c$ in the main convention. This is an explicit assumption on $\widehat{\Delta}$; estimator evidence is reported in Sections H.6 and H.7.

Effect of the mixture weight. The next lower bound concerns the same continuous expected-update dynamics beyond the initial sign at the root.

Proposition 12 (Learning time with a smaller regret-based component)**.** *Use exact values and the same initialization as in Proposition 10, but suppose $1 - c < \lambda \le 1$. Define the positive constants*

$$d := \frac{\lambda + c - 1}{2\lambda}, \qquad \kappa := \frac{\lambda + c - 1}{2}(1 - p_0)^2.$$

Whenever $q_0^H < d$, for every $0 < \epsilon < 1 - c$,

$$T_\lambda(\epsilon) \ge \frac{(d/q_0^H)^{4\kappa} - 1}{p_0 \kappa}.$$

For fixed c, p_0, q_0, λ with $\lambda > 1 - c$, this lower bound is exponential in H, including mixtures with a nonzero regret-based component.

Proof. Until p_π reaches $d < c$, the root action-value difference is at most $-(\lambda + c - 1)/2$, so $p \le p_0$. Write $s(\tau) := \int_0^\tau p(u)\,du$; finite-time root probabilities are positive, so this change of variable is valid. Equation (5) implies, up to that hitting time,

$$\frac{dp}{ds} = p(1-p)^2\big[\lambda p_\pi + 1 - \lambda - c\big] \le -\kappa p, \qquad p(s) \le p_0 e^{-\kappa s}.$$

On the other hand, since every $q_k \in (0,1)$ and the coefficient in square brackets below is at most one,

$$\begin{aligned}\frac{d\log p_\pi}{ds} &= \sum_{k=1}^{H}\left(\prod_{j<k} q_j\right) q_k(1-q_k)^2\left[\lambda \prod_{j>k} q_j + 1 - \lambda\right] \\ &\le \frac{1}{4}\sum_{k=1}^{H}\left(\prod_{j<k} q_j\right)(1-q_k) = \frac{1}{4}(1-p_\pi) \le \frac{1}{4}.\end{aligned}$$

Thus reaching $p_\pi = d$ requires $s \ge 4\log(d/q_0^H)$. Converting back to learning time yields

$$\tau = \int_0^s \frac{du}{p(u)} \ge \frac{e^{\kappa s} - 1}{p_0\kappa} \ge \frac{(d/q_0^H)^{4\kappa} - 1}{p_0\kappa}.$$

Obtaining return above c requires $p_\pi > c > d$, proving the result. If d is never reached, the hitting time is infinite and the inequality still holds. □

The $A^\star$ component trains each reached continuation decision without multiplication by the current probability of completing the remaining suffix after that decision. Sufficient weight on this component retains root access while those decisions improve. A smaller weight on the regret-based component can improve a local comparison without removing exponential learning time in this model. These bounds characterize continuous expected updates with independent logits, exact expectations, and fixed coefficient scales.

G Complete-Tree Separation with Correct Current-Policy Rankings

Consider a tree-structured MDP with deterministic transitions and rewards whose action tree is complete, depth-H, and K-ary, where $K \geq 2$, with one leaf returning 1 and every other leaf returning $c < 1$. Write $R := 1 - c$. Use an independently parameterized tabular softmax at every prefix, initialized uniformly. Let $a_k^\star$ denote the action on the unique optimal path at step k, and let $q_k := \pi(a_k^\star \mid \xi_k)$. Every action that leaves the optimal path has action value c. Consequently, at a prefix on the optimal path, the current-policy and optimal action-value differences between $a_k^\star$ and any competing action are

$$Q^\pi(\xi_k, a_k^\star) - Q^\pi(\xi_k, a) = R \prod_{j>k} q_j, \qquad Q^\star(\xi_k, a_k^\star) - Q^\star(\xi_k, a) = R.$$

The first difference is positive but depends on successful execution of the remaining continuation.

Under the continuous expected update using the normalized mixture $\lambda A^\pi + (1-\lambda)A^\star$, where $\lambda \in [0, 1]$, define the action-value difference

$$D_k := R \left[\lambda \prod_{j>k} q_j + 1 - \lambda \right] \geq (1-\lambda)R. \tag{22}$$

The $K-1$ competing actions have equal values and, from the uniform initialization, equal probabilities. If $d_k := \prod_{j<k} q_j$ is the probability of reaching ξ_k, the expected softmax update therefore gives

$$\dot{q}_k = \frac{K}{K-1} d_k D_k q_k^2 (1-q_k)^2. \tag{23}$$

Proposition 13 (Complete-tree learning-time separation)**.** *For $0 < \epsilon < 1/2$, let $T_\lambda(\epsilon)$ be the first learning time at which $V^\pi(\xi_0) \geq 1 - \epsilon R$. If $K^{-H} < 1 - \epsilon$, then*

$$T_1(\epsilon) \geq \frac{K-1}{KRH} \left[K^H - \frac{1}{1-\epsilon} \right]. \tag{24}$$

For every $\lambda < 1$,

$$T_\lambda(\epsilon) \leq \frac{K(K-1)H^2}{(1-\epsilon)(1-\lambda)R\epsilon}. \tag{25}$$

Thus, for fixed K, c, ϵ, and $\lambda < 1$, the learning time is exponential in H using A^π alone and polynomial in H using the exact combined advantage.

Proof. First consider A^π alone, corresponding to $\lambda = 1$. Symmetry across the H prefixes on the optimal path is preserved, so write their common optimal-action probability as q. Substituting $d_k = q^{k-1}$ and $D_k = Rq^{H-k}$ into Equation (23) gives

$$\dot{q} = \frac{K}{K-1} R q^{H+1} (1-q)^2 \leq \frac{K}{K-1} R q^{H+1}.$$

The expected root return is $c + Rq^H$. Hence reaching $V^\pi(\xi_0) \geq 1 - \epsilon R$ requires $q^H \geq 1 - \epsilon$. When $K^{-H} < 1 - \epsilon$, integration from $q(0) = 1/K$ to $(1-\epsilon)^{1/H}$ gives Equation (24).

Now fix $\lambda < 1$ and set $\zeta := \epsilon / H$. By Equations (22) and (23), every q_k is nondecreasing. Once the preceding optimal-path actions have probability at least $1 - \zeta$, prefix ξ_k is reached with probability

$$d_k \geq (1-\zeta)^{k-1} \geq 1 - \epsilon.$$

Starting from any $q_k \geq 1/K$, the additional time required to reach $q_k \geq 1 - \zeta$ is at most

$$\frac{K-1}{K(1-\epsilon)(1-\lambda)R} \int_{1/K}^{1-\zeta} \frac{dq}{q^2(1-q)^2} \leq \frac{K(K-1)}{(1-\epsilon)(1-\lambda)R\zeta}.$$

Applying this bound successively to the at most H prefixes gives Equation (25). At that time, the optimal leaf is selected with probability at least $(1-\epsilon/H)^H \geq 1-\epsilon$, which proves the required root-return criterion. Equations (24) and (25) give the stated exponential and polynomial dependence on H. □

The separation concerns learning dynamics given exact action values. With sampled feedback, obtaining the fixed positive contribution in Equation (22) also requires discovering a high-return continuation in the rollout tree.

H ACTION-VALUE FEEDBACK FROM SAMPLED ROLLOUT SUBTREES

This appendix describes how we construct sampled proxies for the optimal action values $Q^\star_{\rho,t}(\xi_t, a)$ and $V^\star_{\rho,t}(\xi_t)$, and hence for the immediate regret $\mathrm{IReg}_{\rho,t}(\xi_t, a) = V^\star_{\rho,t}(\xi_t) - Q^\star_{\rho,t}(\xi_t, a)$. Exact evaluation requires optimizing over future actions while accounting for all possible environment outcomes, which is infeasible in the agentic environments considered here. We therefore construct a sampled rollout tree $\widehat{\mathcal{T}}(\xi_t)$, compute returns-to-go at its sampled leaves, and use them to form $\widehat{Q}_t$, $\widehat{V}_t$, the sampled proxy for immediate regret $\widehat{\mathrm{IReg}}_t = \widehat{V}_t - \widehat{Q}_t$, and its negative, the proxy for the regret-based advantage of action a at prefix ξ_t, $\widehat{A}_t(\xi_t, a) = \widehat{Q}_t(\xi_t, a) - \widehat{V}_t(\xi_t)$.

H.1 SAMPLED ROLLOUT SUBTREES

By construction, $\widehat{\mathcal{T}}(\xi_t)$ is a tree of realized continuations rooted at ξ_t and contains every intermediate prefix between its root and each sampled leaf. Every node is a complete interaction prefix and every nonroot node has one parent.

Local rollout-tree sampling. For each task p, we sample N base trajectories under $\pi_{\theta_{\mathrm{old}}}$ and select a set of root prefixes $\mathcal{X}_p$ at which sampled-search proxies for regret-based advantages will be constructed. Each element of $\mathcal{X}_p$ is a complete prefix ξ_t, with associated agent prompt h_t. We adopt Tree-GRPO's rollout-tree construction (Ji et al., 2026). On MiniWoB++, ALFWorld, and Beat Block Hammer, we select branching prefixes using the old policy. For an agent output $y_t = (\ell_{t,i})_{i=1}^{L_t}$, we compute the mean negative log likelihood

$$\nu(h_t, y_t) := -\frac{1}{|\mathcal{I}_t|} \sum_{i \in \mathcal{I}_t} \log p^{\mathrm{LM}}_{\theta_{\mathrm{old}}} \left(\ell_{t,i} \mid h_t, \ell_{t,1:i-1}\right),$$

where $\mathcal{I}_t$ is the token span used to express the parsed action. When this span is unavailable, $\mathcal{I}_t$ contains the non-boilerplate output tokens. Within each base trajectory, we select the prescribed number of prefixes with the largest values of $\nu(h_t, y_t)$. On WebShop, we instead branch at fixed prefixes, namely the first two steps of each base trajectory.

For each selected root prefix $\xi \in \mathcal{X}_p$, we construct a sampled rollout subtree $\widehat{\mathcal{T}}(\xi)$. Let A be the number of alternatives sampled at each split and J be the maximum number of split points composed along a rollout. In base branching, we freeze each selected prefix on a completed base rollout and sample $A-1$ alternative suffix rollouts; the sampled suffixes are not branched again. Because every selected prefix lies on the base rollout, J equals the number of selected prefixes on it. In a balanced tree, each sampled rollout can itself be branched at a later selected prefix until the remaining split budget is exhausted. In the idealized case with no early termination and no duplicate actions, the balanced construction produces at most A^J completed rollouts per local root. Figure 5 illustrates the two constructions.

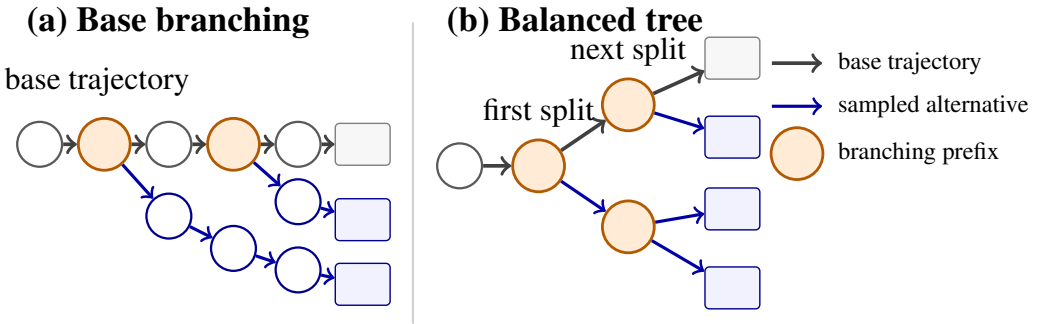

Figure 5: **Agentic rollout-tree constructions.** Base branching samples alternative suffixes from selected prefixes on a base trajectory without recursively branching those suffixes. A balanced tree repeats the split on the resulting rollouts until the specified branching depth. The schematic uses a common horizon for all displayed rollouts.

Each selected prefix on a base trajectory defines a separate local subtree. Within that subtree, rollouts are grouped by their parsed first action; estimates are not pooled across distinct selected prefixes. The regret-based component is set to zero when the sampled subtree does not provide estimates for both the selected action and at least one alternative action.

H.2 Action-Value Estimation

The sampled-maximum estimator $\widehat{Q}_t^{\max}$ uses the maximum return among sampled rollouts under each child action. For our language-agent experiments, we also use the pseudo-posterior predictive estimator $\widehat{Q}_t^{\mathrm{post}}$ described in Section H.4, which incorporates predicted returns for unsampled continuation leaves. Let $\underline{R}_t(\xi_t)$ and $\overline{R}_t(\xi_t)$ be valid lower and upper bounds on realized returns-to-go and set $s_t^R(\xi_t) = \overline{R}_t(\xi_t) - \underline{R}_t(\xi_t) > 0$. For a sampled terminal leaf ℓ, define the normalized return

$$Y_\ell := \frac{R_t(\ell \mid \xi_t) - \underline{R}_t(\xi_t)}{s_t^R(\xi_t)} \in [0,1].$$

For each node v, let $\mathcal{L}(v)$ denote its terminal descendant leaves and let $M(v) = \max_{\ell \in \mathcal{L}(v)} Y_\ell$ be the latent normalized maximum below v. We estimate $M(v)$ from the sampled leaves and map it back to the original return scale:

$$\widehat{Q}_t^{\mathrm{post}}(\xi_t, a) := \underline{R}_t(\xi_t) + s_t^R(\xi_t)\widehat{M}(\xi_t, a).$$

H.3 Prefix-Wise Centering for Policy Optimization

In our agentic experiments, we use the following centering as an optimization adjustment to the estimated signal. The sampled proxy for the regret-based advantage

$$\widehat{A}_t(\xi_t, a) := \widehat{Q}_t(\xi_t, a) - \max_{a'} \widehat{Q}_t(\xi_t, a')$$

is nonpositive and assigns zero to every highest-valued sampled action. For policy optimization, we use the centered advantage estimate

$$\widetilde{A}_t(\xi_t, a) = \widehat{A}_t(\xi_t, a) + b_t(\xi_t), \qquad b_t(\xi_t) := \frac{U_t^{\mathrm{ctr}}(\xi_t) - L_t^{\mathrm{ctr}}(\xi_t)}{2}. \tag{26}$$

The bounds and the resulting offset are held fixed during each policy update. The shift preserves the ranking of actions induced by $\widehat{Q}_t(\xi_t, \cdot)$ at each prefix. For sampled-range centering it places the resulting advantage estimates symmetrically around zero. For available-range centering it instead adds half the width of a componentwise return envelope and need not center the empirical advantage range.

We use two constructions of the range. For sampled-range centering,

$$L_t^{\mathrm{ctr}}(\xi_t) = \min_{a \in \widehat{\mathcal{A}}(\xi_t)} \widehat{Q}_t(\xi_t, a), \qquad U_t^{\mathrm{ctr}}(\xi_t) = \max_{a \in \widehat{\mathcal{A}}(\xi_t)} \widehat{Q}_t(\xi_t, a).$$

For available-range centering, write the return-to-go of a sampled rollout ending at leaf ℓ as

$$R_t(\ell \mid \xi_t) = T_\ell + \sum_{r \geq 0} P_{\ell,r},$$

where $T_\ell \geq 0$ is its discounted terminal contribution and $P_{\ell,r} \leq 0$ are its discounted penalty components, with a missing component represented by zero. The corresponding bounds use the componentwise envelope

$$L_t^{\mathrm{ctr}}(\xi_t) = \min_\ell T_\ell + \sum_{r \geq 0} \min_\ell P_{\ell,r}, \qquad U_t^{\mathrm{ctr}}(\xi_t) = \max_\ell T_\ell + \sum_{r \geq 0} \max_\ell P_{\ell,r},$$

where the extrema are over sampled rollouts rooted at ξ_t. When only an aggregate rollout penalty is available, the sum of componentwise extrema is replaced by the corresponding extrema of that aggregate penalty.

For any fixed prefix-dependent constant $b_t(\xi_t)$ that is independent of the current sampled action, adding the shift leaves the ideal conditional on-policy policy-gradient expectation unchanged when clipping is inactive:

$$\begin{aligned}
&\mathbb{E}_{a\sim\pi_{\theta,t}(\cdot|\xi_t)}\left[\widetilde{A}_t(\xi_t,a)\nabla_\theta \log \pi_{\theta,t}(a \mid \xi_t)\right] \\
&\quad = \mathbb{E}_{a\sim\pi_{\theta,t}(\cdot|\xi_t)}\left[\widehat{A}_t(\xi_t,a)\nabla_\theta \log \pi_{\theta,t}(a \mid \xi_t)\right] + b_t(\xi_t)\nabla_\theta \sum_a \pi_{\theta,t}(a \mid \xi_t) \\
&\quad = \mathbb{E}_{a\sim\pi_{\theta,t}(\cdot|\xi_t)}\left[\widehat{A}_t(\xi_t,a)\nabla_\theta \log \pi_{\theta,t}(a \mid \xi_t)\right].
\end{aligned}$$

Thus a fixed, stop-gradient prefix shift preserves the ideal conditional policy-gradient direction of the sampled regret proxy.

Because the implemented offset is estimated from the sampled prefix group and the update uses finite-batch clipped PPO, positive and negative advantages enter different clipped regions as the likelihood ratio moves away from one, so centering can affect the realized update. Moreover, positive and negative policy-gradient terms can have different empirical learning effects: in long-context reasoning, negative gradients from failed traces have been found to increase exploration and response length, whereas positive gradients reinforce successful traces (Setlur et al., 2026). The positive shift therefore prevents the regret-based component from consisting only of nonpositive advantage values and supplies positive advantages for high-valued sampled actions and negative advantages for sufficiently weaker actions.

H.4 Pseudo-Posterior Prediction for Partially Sampled Rollout Subtrees

For our language-agent experiments, we use the following generalized-Bayes pseudo-posterior construction to extrapolate over unsampled continuation leaves. When a sampled rollout subtree is sparse, we additionally consider a two-level statistical prediction model over the local prefix, its child actions, and their leaves. For a normalized return $y \in [0,1]$, define the fractional Bernoulli log loss

$$\ell_{\mathrm{frac}}(\theta; y) := -y \log \theta - (1-y)\log(1-\theta).$$

Given a Beta prior density $p(\theta)$, we use the loss-based update

$$q(\theta) \propto p(\theta) \exp\left\{-w_{\mathrm{obs}} \sum_i \ell_{\mathrm{frac}}(\theta; Y_i)\right\}, \qquad w_{\mathrm{obs}} > 0,$$

which is a loss-based generalized-Bayes update (Bissiri et al., 2016) with the Gibbs pseudo-posterior form used in PAC-Bayes (Alquier et al., 2016). Each Y_i contributes fractional pseudo-counts $w_{\mathrm{obs}} Y_i$ and $w_{\mathrm{obs}}(1-Y_i)$. We use *pseudo-posterior* to emphasize that this is a loss-based update, not a conjugate posterior derived from a generative likelihood for continuous returns.

Let $\mu_0 \in (0,1)$ and $\eta_0 > 0$ denote the root prior mean and precision, so the root prior is $\mathrm{Beta}(\eta_0\mu_0, \eta_0(1-\mu_0))$. Given all normalized sampled returns below the prefix, with count n_0 and sum S_0, the root pseudo-posterior mean is

$$\bar{\theta}_0 = \frac{\eta_0\mu_0 + w_{\mathrm{obs}} S_0}{\eta_0 + w_{\mathrm{obs}} n_0}.$$

For each sampled child action a, let n_a and S_a be the count and sum of its normalized sampled returns. Using the empirical prior $\mathrm{Beta}(\eta\bar{\theta}_0, \eta(1-\bar{\theta}_0))$, its latent action quality has the pseudo-posterior approximation

$$q_a(\theta_a) = \mathrm{Beta}\big(\eta\bar{\theta}_0 + w_{\mathrm{obs}} S_a, \eta(1-\bar{\theta}_0) + w_{\mathrm{obs}}(n_a - S_a)\big),$$

where $\eta > 0$ controls shrinkage toward the root pseudo-posterior mean. Because $\bar{\theta}_0 \in (0,1)$ and $\eta, w_{\mathrm{obs}} > 0$, both Beta shape parameters are strictly positive, so q_a is a proper probability distribution. Suppose the prediction model assigns K_a possible leaves to action a and therefore $h_a = \max\{0, K_a - n_a\}$ unsampled leaves. For each Monte Carlo draw $\theta_a^{(r)} \sim q_a$, we sample

$$Z_{a,j}^{(r)} \mid \theta_a^{(r)} \sim \mathrm{Beta}\Big(\lambda_{\mathrm{leaf}}\theta_a^{(r)}, \lambda_{\mathrm{leaf}}(1-\theta_a^{(r)})\Big), \qquad j = 1, \ldots, h_a,$$

where $\lambda_{\text{leaf}} > 0$ is the concentration of the predictive leaf-return distribution. We estimate the normalized subtree maximum by averaging the maximum of the sampled and predicted leaf returns:

$$\widehat{M}(\xi_t, a) = \frac{1}{S}\sum_{r=1}^{S} \max\left\{ \max_{\ell \in \widehat{\mathcal{L}}(\xi_t, a)} Y_\ell, \max_{1 \le j \le h_a} Z_{a,j}^{(r)} \right\}.$$

Here the maximum over an empty set is $-\infty$, so when $h_a = 0$ only the sampled leaves enter the maximum.

H.5 Estimator Interpretation and Sampled-Max Coverage

When $h_a = 0$, the pseudo-posterior predictive maximum reduces to the maximum sampled return. More generally, sampled leaves remain candidates in every pseudo-posterior predictive maximum, so $\widehat{Q}_t^{\text{post}} \ge \widehat{Q}_t^{\max}$. Relative to the exact optimal action value $Q^\star_{\rho,t}$, however, the pseudo-posterior estimate need not be a lower bound, an upper bound, or unbiased; its accuracy depends on the hierarchical predictive model. When environment outcomes are deterministic conditional on the task-conditioned complete prefix and action, the target reduces to the full-tree value $Q_t^\star$, and the maximum sampled return is a lower bound that reaches $Q_t^\star$ only after a maximizing leaf has been sampled. In this deterministic setting, the sampled maximum is equivalent to recursive maximum backups through the sampled rollout subtree. It optimizes only over the continuations represented in that subtree. In particular, an optimal trajectory with zero probability under the sampling policy cannot be recovered from sampled rollouts. Pseudo-posterior prediction extrapolates beyond those rollouts, but identifying unsampled regions of the tree depends on the validity of the model.

Under stochastic environment outcomes, $Q^\star_{\rho,t}$ averages over environment outcomes and maximizes over actions at each prefix. A maximum over realized rollout returns does not perform this chance-node averaging: it selects among both action choices and sampled outcomes. Consequently, $\widehat{Q}_t^{\max}$ is not generally a lower bound, upper bound, or unbiased estimator of $Q^\star_{\rho,t}$. Repeated outcome samples with expectation backups, or an environment model, are instead needed to target the stochastic value; the pseudo-posterior construction is a modeling approximation when used for this purpose.

A finite-sample coverage bound for the sampled maximum. The following statement concerns the maximum of continuations discovered in a deterministic environment, not pseudo-posterior extrapolation.

Proposition 14 (Sampled-maximum coverage)**.** *Suppose the induced finite-horizon history-based MDP is deterministic: for every feasible task-conditioned complete prefix–action pair (ξ_k, a_k), $\rho(\cdot \mid \xi_k, a_k)$ is a point mass. Fix a prefix ξ, a nonempty set of sampled actions $\widehat{\mathcal{A}}$, and $\varepsilon \ge 0$. For each $a \in \widehat{\mathcal{A}}$, independently sample $m_a \ge 1$ complete continuations from a fixed policy after forcing a. Let*

$$\kappa_a(\varepsilon) := \mathbb{P}(R_t \ge Q_t^\star(\xi, a) - \varepsilon \mid \xi, a)\,.$$

Then

$$\mathbb{P}\Big(\exists a \in \widehat{\mathcal{A}} :\ Q_t^\star(\xi, a) - \widehat{Q}_t^{\max}(\xi, a) > \varepsilon\Big) \le \sum_{a \in \widehat{\mathcal{A}}} [1 - \kappa_a(\varepsilon)]^{m_a}.$$

On the complementary event, the error in any pairwise action-value difference is at most ε. In particular, a pair whose exact difference exceeds ε retains its strict ordering.

Proof. Every feasible continuation has return at most $Q_t^\star(\xi, a)$. For a given action the maximum is more than ε below this value exactly when all m_a samples miss its ε-optimal continuations, an event of probability $[1 - \kappa_a(\varepsilon)]^{m_a}$. Apply the union bound over actions. On the complementary event the underestimation errors lie in $[0, \varepsilon]$; the difference of any two such errors has absolute value at most ε. □

For $\kappa_a(\varepsilon) \ge \kappa > 0$ and $m_a \ge m$, $m \ge \kappa^{-1} \log(|\widehat{\mathcal{A}}|/\delta)$ suffices for probability at least $1 - \delta$, where $0 < \delta < 1$. Small coverage probability can therefore make useful evidence expensive. The independent-continuation assumption cannot be obtained merely by counting correlated leaves of a recursively sampled tree. The result does not cover unsampled actions or establish the error of the pseudo-posterior predictive estimator.

H.6 Synthetic Bias Evaluation of the Pooled Pseudo-Posterior Estimator

Experimental setting. We evaluate the pooled pseudo-posterior estimator on $1{,}000$ independent synthetic prefixes, each with four actions and 16 terminal continuations per action. To allow shared context while retaining action-specific variation, we draw independent standard-normal variables $U_i, V_{i,a}$ and generate

$$\theta_{i,a} = \mathrm{sigmoid}\Big((U_i + V_{i,a})/\sqrt{2}\Big), \qquad Y_{i,a,j} \mid \theta_{i,a} \sim \mathrm{Beta}(20\theta_{i,a}, 20(1-\theta_{i,a}))\,.$$

The rewards are generated once and then fixed, so each resulting tree-structured MDP has deterministic transitions and rewards, and its true action value is $Q^\star(\xi_i, a) = \max_j Y_{i,a,j}$. We reveal two leaves per action uniformly without replacement and estimate the action value using the pooled pseudo-posterior construction above. This is the primary synthetic setting specified before running the study.

Bias evaluation. We estimate bias by averaging the signed error $\widehat{Q}^{\mathrm{post}}(\xi_i, a) - Q^\star(\xi_i, a)$ over prefixes and actions. A 95% confidence interval is obtained from $2{,}000$ bootstrap replicates, resampling whole prefixes to preserve dependence among sibling actions. The estimated bias is

$$\widehat{\mathrm{Bias}} = -0.00025, \qquad 95\% \text{ confidence interval} = [-0.00274,\ 0.00203].$$

Because this interval contains zero, we do not reject the null hypothesis of zero mean bias at the nominal 5% level using this interval-based assessment.

H.7 Shared-Prefix Information in ALFWorld Continuation Returns

We examine the empirical motivation for partial pooling in the pseudo-posterior predictive estimator. First, we measure how much variation in continuation returns is shared by actions at the same prefix. Second, we test whether observations from sibling actions—distinct actions at the same prefix—improve prediction of held-out continuation returns. Finally, we compare the pseudo-posterior predictive maximum with the stronger-policy proxy from Section 4.2.2 using leave-one-task-out validation.

Data. We use the frozen prefixes from Section 4.2.2: three prefixes from each of 32 held-out ID tasks, with the **RegPO** trajectory action and three distinct admissible alternatives proposed by GPT-5.5 at each prefix. For each of Update 10, Tree-GRPO, and **RegPO**, the data contain eight independent continuations per action, or 3,072 continuations per policy. We use the same discounted returns as in that experiment, including invalid-action penalties. Let $R_{i,a,j}$ denote the discounted return-to-go in the original reward units, where i indexes prefixes, a indexes their four actions, and j indexes continuations. The terminal success reward is 10; discounting and penalties reduce continuation returns. The single GPT-5.5 continuation per action is not used in the variance or held-out-continuation prediction tests; it supplies the reference for the final bias diagnostic. All aggregate results weight the three policies equally. The variance and mean-prediction analyses fit each policy separately. Variance shares are ratios of averaged components, and prediction-error reductions are ratios of averaged errors. Their bootstrap resamples retain all three policies for each sampled task.

Variation shared by a prefix. We use the working variance-components model

$$R_{i,a,j} = \alpha_{p(i)} + f(t_i) + \beta b_{i,a} + u_i + v_{i,a} + \epsilon_{i,a,j},$$

where $p(i)$ is the task, t_i is the prefix depth, f is quadratic, and $b_{i,a}$ indicates the trajectory action rather than a proposed alternative. The centered, mutually uncorrelated terms u_i, $v_{i,a}$, and $\epsilon_{i,a,j}$ represent shared-prefix variation, action-specific variation, and variation among repeated continuations, respectively. We estimate these components by moments, using repeated continuations to correct the dispersion of action and prefix means for sampling variation. The estimates also account for task, depth, action source, and residual degrees of freedom.

Shared prefix [95% CI]	Action-specific	Continuation
17.9 [5.6, 27.4]	60.3	21.8

Table 3: **Aggregate variation in ALFWorld continuation returns.** Estimated shares (%) of the sum of the three variance components, weighting the three policies equally after accounting for task, depth, and action source within each policy. The shared-prefix interval uses 2,000 task-level bootstrap resamples.

Table 3 identifies a positive shared-prefix component in the aggregate, with a 95% interval of $[5.6\%, 27.4\%]$ for its variance share. Action-specific variation remains substantial: shared context does not imply equal action values. For example, in a recorded task requiring a clean soapbar to be placed in a toilet, the agent has already acquired the soapbar at the selected prefix. The mean returns under **RegPO** are 8.145 for moving to either of two sinkbasins, 6.983 for examining the soapbar, and 4.492 for placing it in the toilet immediately. The acquired object and task history are shared, while the next action changes the remaining work.

Predicting held-out continuation returns. At every action, we reveal two of its eight continuations and predict the other six, averaging over all $\binom{8}{2} = 28$ choices of revealed sample indices, using the same index pair at every action within a split. We compare the two-sample action mean with two partial-pooling predictors. All three are mean predictors used to test the value of shared information, rather than estimators of the subtree maximum. The task-pooling predictor also uses the revealed observations at the other two prefixes of the same task. The sibling predictor additionally uses the other three actions at the target prefix. Both adjust for depth and action source. Specifically, the prediction is

$$\widehat{R}_{i,a} = (1 - \lambda - \omega)\bar{R}^{(2)}_{i,a} + \lambda \widehat{m}_{i,a} + \omega \widehat{s}_{i,a}, \qquad \lambda, \omega \geq 0, \quad \lambda + \omega \leq 1,$$

where $\widehat{m}_{i,a}$ is the mean of the revealed returns at the other two task prefixes, adjusted to the target depth and action source, and $\widehat{s}_{i,a}$ is the corresponding adjusted mean over sibling actions. The task-pooling predictor restricts $\omega = 0$; the action mean sets both weights to zero. Each predictor therefore retains the option of no pooling.

We evaluate by leave-one-task-out validation. For each held-out task, depth and action-source effects are fitted on the other 31 tasks. Pooling weights are selected on a 0.05 grid by a second leave-one-task-out validation within those 31 tasks. Only the revealed continuations from the held-out task enter its predictions. We measure mean squared error (MSE) against individual withheld returns and average within tasks. Paired 95% confidence intervals use $2{,}000$ task-level bootstrap resamples of these errors, holding the fitted predictors fixed.

Prediction MSE			Additional reduction (%)
Action mean	Task pooling	With siblings	[95% CI]
1.2604	1.1703	1.1296	3.5 [0.5, 6.8]

Table 4: **Aggregate prediction of held-out ALFWorld continuation returns.** Two continuations per action are observed and six are withheld. MSE is in squared reward units and averaged equally across the three policies; lower is better. The last column gives the relative MSE reduction from adding sibling information to the task-pooling predictor, with a paired task-level bootstrap interval.

Table 4 shows that sibling information improves aggregate prediction beyond task pooling by 3.5% (95% interval: $[0.5\%, 6.8\%]$). Relative to separate action means, the reduction is 10.4% ($[6.5\%, 14.1\%]$).

Together, these results support partial pooling of sparse continuation observations at the selected ALFWorld prefixes. Predicting continuation means, however, does not establish the calibration of the pseudo-posterior predictive maximum or its accuracy relative to $Q^{\star}_{\rho,t}$; see Section H.5.

Comparison with a stronger-policy proxy. We next compare the pseudo-posterior predictive maximum with $\widehat{Q}_{\mathrm{GPT}}$, the stronger-policy proxy from Section 4.2.2. For each learned policy and candidate action, we use all eight observed continuations and set $K_a = |\mathcal{A}_{\mathrm{adm}}(\xi)|$, the number of admissible actions at the original prefix. This gives $K_a \in [11, 71]$ and $h_a = K_a - 8$ predicted continuation leaves. The action count specifies an extrapolation budget; it is not the number of full downstream trajectories. We normalize returns using fixed horizon-based bounds and report estimates in the original reward units.[1] We evaluate the predictive expectation by numerical integration.

We evaluate by leave-one-task-out validation using a common pool of 144 candidate configurations of the pseudo-posterior predictive estimator. For each held-out task, we select one configuration shared across the three policies using only the other 31 tasks. Selection minimizes the magnitude of aggregate signed error against the GPT reference, subject to mean absolute and mean squared errors not exceeding those of the original setting on these 31 tasks. The held-out task's eight continuations per action update its pseudo-posterior; its GPT-reference values are used only for evaluation. Let $\widehat{Q}^{\mathrm{post},\pi}(\xi, a)$ denote the resulting estimate from policy π's continuations. Averaging the signed errors equally over the three policies and their 384 shared prefix–action pairs gives

$$\widehat{B}_{\mathrm{GPT}} := \frac{1}{3 \cdot 384} \sum_{\pi} \sum_{(\xi,a)} \left(\widehat{Q}^{\mathrm{post},\pi}(\xi, a) - \widehat{Q}_{\mathrm{GPT}}(\xi, a) \right) = -0.111.$$

The magnitude of this aggregate bias is **1.11% of the terminal-reward scale of 10**; the mean absolute error (MAE) is 2.277 reward units. The percentage describes aggregate signed error, not absolute error. Opposite-signed errors across policies partly cancel, so a small aggregate bias does not imply uniformly accurate action values or unbiased estimation of $Q^{\star}_{\rho,t}$.

This comparison remains exploratory on the current dataset. GPT supplies one continuation per action rather than an exact optimal value.

[1] With $H = 30 - t$ steps remaining at prefix depth t, we use $\overline{R}_t(\xi_t) = 10(0.95) = 9.5$ and $\underline{R}_t(\xi_t) = -0.1 \sum_{j=1}^{H-1} 0.95^j$. The forced first action is admissible. These bounds replace the sampled return envelope used during training.

I BASELINE ADVANTAGE ESTIMATORS

For a task instance p, let $G(p) = \{\tau_n\}_{n=1}^{N}$ be the trajectories sampled under the old policy and write $R_n = R^{\text{env}}(\tau_n)$.

GRPO. GRPO (Shao et al., 2024) uses the group-normalized trajectory return

$$\widehat{A}_n^{\tau} = \frac{R_n - \overline{R}_G}{s_G},$$

where $\overline{R}_G$ and s_G are the mean and standard deviation of $\{R_n\}_{n=1}^{N}$. In Equation (1), GRPO sets $\widehat{A}_{n,t,i} = \widehat{A}_n^{\tau}$ for every token i generated at every step t of trajectory τ_n.

GiGPO. Group-in-Group Policy Optimization (GiGPO) (Feng et al., 2025) augments the GRPO trajectory-level relative advantage estimate with a step-level relative advantage estimate computed among steps that share an environment-state identifier. Let $s_{n,t}$ denote this identifier at step t of trajectory τ_n, and define the matched-state group

$$\mathcal{I}(n,t) := \{(n',t') : s_{n',t'} = s_{n,t}\}.$$

For each $(n',t') \in \mathcal{I}(n,t)$, the discounted return-to-go is

$$R_{t'}^{(n')} := \sum_{v=t'}^{T_{n'}-1} \gamma^{v-t'} r_v^{\text{step},(n')}.$$

GiGPO normalizes the return-to-go within the matched-state group:

$$\widehat{A}_{n,t}^{S} = \frac{R_t^{(n)} - \text{mean}\Big(\{R_{t'}^{(n')} : (n',t') \in \mathcal{I}(n,t)\}\Big)}{\text{std}\Big(\{R_{t'}^{(n')} : (n',t') \in \mathcal{I}(n,t)\}\Big)}.$$

Its combined advantage estimate at step t is

$$\widehat{A}_{n,t}^{\text{GiGPO}} = \widehat{A}_n^{\tau} + \omega \widehat{A}_{n,t}^{S},$$

where ω controls the relative weight of the step-level component; in Equation (1), GiGPO sets $\widehat{A}_{n,t,i} = \widehat{A}_{n,t}^{\text{GiGPO}}$ for every token i generated at step t of trajectory τ_n.

Tree-GRPO. Tree-GRPO (Ji et al., 2026) samples root-to-leaf trajectories from rollout trees. Let $\mathcal{N}_b$ index the trajectories sampled in tree b, and let $\overline{R}_{\text{pool}}$ and s_{pool} be the mean and standard deviation of returns pooled across the sampled trees. For each tree b, let $\overline{R}_b$ and s_b be the mean and standard deviation of $\{R_n : n \in \mathcal{N}_b\}$. For trajectory $n \in \mathcal{N}_b$, define

$$\widehat{A}_n^{\text{inter}} = \frac{R_n - \overline{R}_{\text{pool}}}{s_{\text{pool}}}, \qquad \widehat{A}_n^{\text{intra}} = \frac{R_n - \overline{R}_b}{s_b}, \qquad \widehat{A}_n^{\text{tree}} = w_{\text{inter}} \widehat{A}_n^{\text{inter}} + w_{\text{intra}} \widehat{A}_n^{\text{intra}}.$$

Tree-GRPO sets $\widehat{A}_{n,t,i} = \widehat{A}_n^{\text{tree}}$ for every token i generated at every step t of τ_n. The baseline weights used in our experiments are reported in Section J.2.2.

J EXPERIMENTAL DETAILS

This section reports the complete experiment configurations.

J.1 FINITE TREE-STRUCTURED MDP EXPERIMENTS

The end-to-end Hard-Safe and random-tree experiments compare GRPO with the normalized mixtures $\lambda\widehat{A}^{\mathrm{intra}} + (1-\lambda)\widehat{A}^{\mathrm{reg}}$ for $\lambda \in \{0, 1/4, 1/2, 3/4, 1\}$. GRPO uses $\widehat{A}^{\tau}$. The endpoint $\lambda = 1$ is the Tree-GRPO recursive-mean condition, $\lambda = 0$ is pure regret-based feedback, and $\lambda = 1/2$ has the direction of the equally weighted additive feedback in $\widehat{A}^{\mathrm{RegPO}}$.

Experiment	Component	Setting
End-to-end Hard-Safe	Core design	$c = 0.6$, $p_0 = 0.5$; tree branching factor $b = 10$
	Sampling	Each mixture condition uses one recursively branched rollout tree per step, sampled separately using the same procedure; at each depth H, GRPO uses $N_H = \lceil \mathbb{E}_{\pi_0}[D_{\text{tree}}] / \mathbb{E}_{\pi_0}[L_{\text{trajectory}}] \rceil$ independent trajectories, fixed across steps and seeds
	Computation	We compute the synthetic estimates from aggregate trajectory and branch counts rather than explicitly enumerating sampled trajectories and rollout trees
	Depth sweeps	Primary $H \in \{4, 6, 8, 10, 12, 14, 16, 20, 30\}$; lower-probability $H \in \{6, 8, 10, 12, 14, 16, 20, 30\}$
	Initialization	Primary $q_0 = 0.72$; lower-probability $q_0 = 0.46$; log-odds heterogeneity 0.35
	Optimization	Root step size 0.2; Hard-step update size 1.5; $1{,}000$ training steps; clipping and KL regularization disabled
	Feedback	$\lambda \in \{0, 1/4, 1/2, 3/4, 1\}$; separately sampled rollout trees for each mixture condition
	Regret-based action values	Node values are computed by leaf-to-root recursive maximization within each training step's rollout tree
	Replication	500 seeds per condition and setting
Random-tree family	Tree and rewards	Depth 4, branching factor 3; root-subtree base returns drawn once per instance from $[0.18, 0.68]$; leaf-noise standard deviation 0.055; returns fixed during training
	High-return leaves	1, 2, 4, or 8 high-return leaves placed uniformly at random without replacement, with returns sampled from $[0.88, 1]$
	Initialization	Logit noise standard deviation 0.55; logits of $Q^\star$-optimal actions at each prefix shifted downward by 0.5, 1, 1.5, or 2
	Sampling	Each mixture condition uses one separately sampled, recursively branched rollout tree with $b \in \{2, 5, 10, 30\}$ actions sampled at every visited nonterminal prefix; GRPO uses $N_b = \lceil (b + b^2 + b^3 + b^4)/4 \rceil$ independent depth-four trajectories to match generated actions
	Optimization	Step size 0.9; $1{,}000$ training steps
	Feedback	$\lambda \in \{0, 1/4, 1/2, 3/4, 1\}$; separately sampled rollout trees for each mixture condition
	Evaluation	First training step at which the expected root return moves 95% of the way from its initial value to the optimal value
	Instances	15 instances for each combination of the number of high-return leaves and initialization bias, totaling 240 instances

Table 5: **Configurations for the finite tree-structured MDP experiments.**

Random-tree action-count matching. In the random-tree family, one recursively branched rollout tree generates $D_b = b + b^2 + b^3 + b^4$ actions, whereas one independent trajectory generates four. We therefore set $N_b = \lceil D_b/4 \rceil$, giving GRPO group sizes 8, 195, 2,778, and 209,483 for $b = 2$, 5, 10, and 30, respectively. The corresponding per-training-step generated-action counts are $(30, 32)$, $(780, 780)$, $(11{,}110, 11{,}112)$, and $(837{,}930, 837{,}932)$ for one recursively branched tree and GRPO. GRPO's independent trajectories are represented exactly by multinomial counts over the $3^4 = 81$ terminal leaves.

Initial-policy action-count matching. For the sampled end-to-end Hard-Safe experiment, let $p_0 := \pi_0(\text{Hard} \mid \xi)$ and $q_{k,0} := \pi_0(a_k^\star \mid \xi_k)$. Let $L_{\text{trajectory}}$ be the number of actions generated by one independent trajectory and D_{tree} the number of actions generated by one recursively branched rollout tree with branching factor b. Early termination gives

$$\mathbb{E}_{\pi_0}[L_{\text{trajectory}}] = 1 + p_0 \sum_{k=1}^{H} \prod_{\ell=1}^{k-1} q_{\ell,0},$$

$$\mathbb{E}_{\pi_0}[D_{\text{tree}}] = b + p_0 \sum_{k=1}^{H} b^{k+1} \prod_{\ell=1}^{k-1} q_{\ell,0},$$

where an empty product equals one. For each depth and initialization, the GRPO group size is fixed to

$$N_H = \left\lceil \frac{\mathbb{E}_{\pi_0}[D_{\text{tree}}]}{\mathbb{E}_{\pi_0}[L_{\text{trajectory}}]} \right\rceil.$$

It remains fixed across training steps and seeds. Thus the matching is defined under the corresponding common initial policy, rather than adjusted dynamically as the methods' policies diverge during training.

J.2 Agentic Benchmarks

J.2.1 Benchmark setup and evaluation

We use RoboTwin 2.0 (Beat Block Hammer) (Chen et al., 2026) for bimanual robotic manipulation. WebShop (Yao et al., 2022) requires an agent to navigate a simulated e-commerce site and purchase an item satisfying a natural-language request. MiniWoB++ (Liu et al., 2018) comprises browser-control tasks such as clicking, typing, and selecting page elements. ALFWorld (Shridhar et al., 2021) evaluates grounded household reasoning through text-based observations and admissible actions.

Table 6 summarizes the model initialization, training horizon, and evaluation protocol for each benchmark.

Benchmark	Initialization	Training horizon	Evaluation
RoboTwin 2.0 (Beat Block Hammer)	Task-specific OpenVLA-OFT 7B checkpoint from SimpleVLA-RL; SFT on 1,000 demonstration trajectories	250 training steps	Fixed ID and OOD seed pools with 128 base trajectories per split; success rate
WebShop	Qwen2.5-1.5B-Instruct	150 training steps	Official test partition; success rate
MiniWoB++	Qwen2.5-1.5B-Instruct	200 training updates	Held-out in-distribution validation with 64 base trajectories; success rate
ALFWorld	Qwen2.5-1.5B-Instruct	150 training updates	Held-out ID evaluation with 140 base trajectories and OOD evaluation with 134 base trajectories; success rate

Table 6: **Agentic-benchmark setup.** Training horizons and evaluation protocols correspond to Table 1.

Initialization and evaluation splits. For Beat Block Hammer, we initialize a vision-language-action model from the task-specific OpenVLA-OFT 7B checkpoint released with SimpleVLA-RL (Kim et al., 2025b;a; Li et al., 2026). The checkpoint was obtained by supervised fine-tuning on 1,000 Beat Block Hammer demonstration trajectories. We operationally define ID as the feasible-seed pool from SimpleVLA-RL's training-seed file and OOD as the disjoint held-out pool from its evaluation-seed file. Both pools use the `demo_randomized` task configuration, and we evaluate the same fixed 128 seeds from each pool at every reported checkpoint. Thus, OOD here denotes generalization to the held-out seed split rather than RoboTwin 2.0's official clean-to-clean (easy) versus clean-to-randomized (hard) protocol. Both splits are reported in Table 1.

For WebShop, MiniWoB++, and ALFWorld, we initialize the agents with Qwen2.5-1.5B-Instruct (Yang et al., 2024). For evaluation, WebShop uses the official test partition of held-out human goals. MiniWoB++ does not define a benchmark-wide ID/OOD instance split; its validation set here samples held-out instances from task families used during training. The ALFWorld validation set is the standard `valid_seen` split. Its `valid_unseen` split instead evaluates out-of-distribution generalization to unseen rooms and scene layouts while retaining the six training task types. Both splits are reported in Table 1.

Checkpoint reporting in Table 1 is benchmark- and method-specific. For **REGPO**, Beat Block Hammer and ALFWorld use the checkpoint with the highest ID success rate and report both ID and OOD performance at that same checkpoint. WebShop and MiniWoB++ instead use the final training checkpoints, at updates 150 and 200, respectively. For the competing methods, we report the checkpoint with the highest score on the corresponding selection split within the training horizon specified in Table 6. ID is the selection split for Beat Block Hammer and ALFWorld; WebShop and MiniWoB++ use their listed test and evaluation splits, respectively. OOD performance does not enter checkpoint selection.

For MiniWoB++, ALFWorld, WebShop, and Beat Block Hammer, per-checkpoint scores are computed from the original evaluation trajectories, excluding auxiliary branch rollouts.

Training seeds and evaluation repetitions. ALFWorld and WebShop use three training seeds and report the mean and one standard deviation across the per-seed scores at the method-specific checkpoints described above, in panel (a) of Table 1. MiniWoB++ uses one training seed and reports the mean and one standard deviation over three evaluation repetitions of a fixed checkpoint. Its deviations therefore measure evaluation variation rather than variation across training seeds. Beat Block Hammer uses one training seed and greedy evaluation on fixed pools of 128 ID and 128 OOD task seeds and therefore reports no standard deviation over evaluation repetitions; this does not imply the absence of variation across task or training seeds. We additionally report ALFWorld learning curves in panel (b) of Table 1 and WebShop learning curves in panel (c) of Figure 6. Both show the mean and one standard deviation across training seeds $\{0, 2, 4\}$ through update 150.

J.2.2 TRAINING IMPLEMENTATION AND BASELINE REPRODUCTION

Optimization and reward settings. MiniWoB++, ALFWorld, and WebShop use AdamW with a constant learning rate of 10^{-6} and no learning-rate warmup. Each policy update uses one PPO epoch, a PPO clipping parameter of 0.2, gradient-norm clipping at 1.0, and a fixed KL coefficient of 0.01. In these benchmarks, the invalid-action penalty of -0.1 is incorporated into token-level rewards before advantage estimation. Beat Block Hammer uses AdamW with an initial learning rate of 2×10^{-6} and cosine decay, without learning-rate warmup. Each update uses one PPO epoch, lower and upper clipping parameters of 0.2 and 0.28, respectively, gradient-norm clipping at 1.0, a KL coefficient of 0, and a binary terminal success reward. Table 7 summarizes the optimization, rollout, and evaluation settings.

Computing Resource Configuration. Training on MiniWoB++, ALFWorld, WebShop, and Beat Block Hammer used two NVIDIA GH200 Grace Hopper Superchips.

Baseline reproduction. We reimplement GRPO (Shao et al., 2024), GiGPO (Feng et al., 2025), and Tree-GRPO (Ji et al., 2026) in the same training stack used by **REGPO** and rerun each baseline rather than transcribing numbers from prior work. We base each baseline configuration on the corresponding prior work where compatible with the common implementation and evaluation protocol.

Setting	MiniWoB++	ALFWorld	WebShop	RoboTwin 2.0 (Beat Block Hammer)
Model	Qwen2.5-1.5B-Instruct	Qwen2.5-1.5B-Instruct	Qwen2.5-1.5B-Instruct	OpenVLA-OFT 7B Fine-Tuned on Expert Trajectory
Training instances	2,048 from 99 task families	3,553	6,410	1,000 task-seed entries (128 unique feasible seeds)
Held-out base trajectories	64	140 (`valid_seen`)	500	128 per split (ID and OOD)
Maximum episode length	10 actions	30 actions	15 actions	400 low-level actions (16 action chunks)
Task batch size	16	16	16	16
AdamW learning rate	10^{-6}, constant	10^{-6}, constant	10^{-6}, constant	2×10^{-6}, cosine
AdamW parameters	$\beta = (0.9, 0.999)$; weight decay 0.01	$\beta = (0.9, 0.999)$; weight decay 0.01	$\beta = (0.9, 0.999)$; weight decay 0.01	$\beta = (0.9, 0.999)$; weight decay 0.01
PPO update	one epoch; minibatch 256; microbatch 4 per GPU	one epoch; minibatch 256; microbatch 16 per GPU	one epoch; minibatch 256; microbatch 4 per GPU	one epoch; minibatch 128; microbatch 1 per GPU
PPO clipping / KL / γ	0.2 / 0.01 / 0.95	0.2 / 0.01 / 0.95	0.2 / 0.01 / 0.95	0.2 lower, 0.28 upper / 0 / 1.0
Prompt / response limit	8,192 / 256 tokens	8,192 / 256 tokens	8,192 / 256 tokens	Not applicable (VLA action generation)
Sampling temperature	1.0 train; 0.4 validation	1.0 train; 0.4 validation	1.0 train; 0.4 validation	1.6 train; greedy evaluation
Terminal success reward	1	10	10	1
Invalid-action penalty	-0.1	-0.1	-0.1	None
Validation interval	every 5 updates	every 5 updates	every 5 updates	every 50 updates
Training seeds	one	three	three	one

Table 7: **Agentic-benchmark optimization and evaluation settings.**

The reproductions exhibit learning at a broadly similar scale to the corresponding source implementations, but do not exactly match their reported numbers. Exact numerical agreement is not expected because the present study uses the current environment wrappers, prompts, action parser, held-out splits, and checkpoint-selection protocol. Because our available compute budget does not support the original ALFWorld rollout settings for all methods, the common protocol shortens the episode horizon from 50 to 30 and lowers the response limit from 512 to 256 tokens relative to the public GiGPO ALFWorld recipe. The common protocol also raises the prompt limit from 2,048 to 8,192 tokens. Tree-GRPO is adapted to these interactive benchmarks from a source evaluation on a different task family. We therefore treat these entries as in-stack reproductions or adaptations under a common protocol, rather than exact replications of previously reported benchmark scores.

All completed 1.5B language-agent runs reported in Table 1, including every baseline and **REGPO**, use 10^{-6}. We fixed this value before the final comparison and did not select a separate learning rate for **REGPO**.

J.2.3 ADVANTAGE ESTIMATION AND ROLLOUT CONSTRUCTION

Advantage estimators. In the agentic experiments, the regret-based component of $\widehat{A}^{\mathrm{RegPO}}$ uses its prefix-wise centered form $\widetilde{A}^{\mathrm{reg}}$ from Equation (26), with $w_{\mathrm{reg}} = 1$. The language-agent experiments use pseudo-posterior predictive action values and available-range centering. The VLA experiments use sampled-maximum action-value estimates and sampled-range centering. Both centering rules are defined in Section H.3; sampled-subtree construction and action-value estimation are described in Section H.

GRPO uses $\widehat{A}^{\tau}$ directly. GiGPO uses $\widehat{A}^{\mathrm{GiGPO}} = \widehat{A}^{\tau} + \omega \widehat{A}^{S}$, where ω is the weight of its step-level component; every evaluated configuration using GiGPO sets $\omega = 1$. Tree-GRPO uses $\widehat{A}^{\mathrm{tree}} = w_{\mathrm{inter}} \widehat{A}^{\mathrm{inter}} + w_{\mathrm{intra}} \widehat{A}^{\mathrm{intra}}$, where $\widehat{A}^{\mathrm{inter}}$ is its GRPO trajectory-level relative advantage estimate over the pooled tree trajectories. The Tree-GRPO baselines set $w_{\mathrm{inter}} = 1$ and $w_{\mathrm{intra}} = 1$, except that Beat Block Hammer uses $w_{\mathrm{intra}} = 1/2$.

For **REGPO**, the base advantage estimate has the Tree-GRPO form:

$$\widehat{A}^{\mathrm{base}} = w_{\mathrm{inter}} \widehat{A}^{\mathrm{inter}} + w_{\mathrm{intra}} \widehat{A}^{\mathrm{intra}}.$$

The regret-based component is then added according to the definition of $\widehat{A}^{\mathrm{RegPO}}$.

The WebShop, MiniWoB++, and ALFWorld configurations add the regret-based component to the same Tree-GRPO advantage estimate used by the corresponding baseline. Beat Block Hammer does not produce trajectory–step pairs sharing an environment-state identifier, so its GiGPO step-level relative advantage estimate and ω are not defined.

Action-value parameters. For the language-agent experiments with **REGPO**, pseudo-posterior predictive action values use two alternatives per branch, depth two, and 128 pseudo-posterior draws. Available-range centering uses terminal-reward bounds $[0, 1]$ on MiniWoB++ and $[0, 10]$ on ALFWorld and WebShop. The GiGPO, Tree-GRPO, and regret-based components each have unit weight. GiGPO uses mean–standard-deviation normalization with unit step weight.

Rollout construction and budget matching. The WebShop, MiniWoB++, and ALFWorld rollout configurations are reported in Table 8. Beat Block Hammer uses four base trajectories and base branching at up to two selected branching prefixes, with two sampled actions at each branching prefix and depth two. Within each benchmark, the GRPO group size and the rollout-tree configurations are chosen to match the rollout budget across GRPO, Tree-GRPO, and **REGPO**. Here, rollout budget is measured by the number of generated agent outputs. Tree-GRPO and **REGPO** use separately sampled rollout trees but the same number of base trajectories, branching-prefix selection rule, number of sampled actions at each branching prefix, and branching depth.

Benchmark	Method	Base trajectories	Tree configuration
MiniWoB++	GRPO / GiGPO	8	None
MiniWoB++	Tree-GRPO / **REGPO**	4	Base branching; highest-NLL selection; up to 2 selected branching prefixes; 2 sampled actions at each branching prefix; depth 2
ALFWorld	GRPO / GiGPO	8	None
ALFWorld	Tree-GRPO / **REGPO**	4	Balanced tree; highest-NLL selection; 2 selected branching prefix; 2 sampled actions at the branching prefix; depth 2
WebShop	GRPO / GiGPO	8	None
WebShop	Tree-GRPO / **REGPO**	4	Base branching; fixed branching prefixes at the first two steps; 2 sampled actions at each branching prefix; depth 2
RoboTwin 2.0 (Beat Block Hammer)	GRPO	8	None
RoboTwin 2.0 (Beat Block Hammer)	Tree-GRPO / **REGPO**	4	Base branching; highest-NLL selection; up to 2 selected branching prefixes; 2 sampled actions at each branching prefix; depth 2

Table 8: **Agentic-benchmark rollout configurations.** The GRPO group size and rollout-tree configurations match the rollout budget. For the language-agent benchmarks, Tree-GRPO and **REGPO** use ten warmup updates before enabling auxiliary branches; GRPO and GiGPO use token-mean aggregation, and the reported Tree-GRPO and **REGPO** runs use sequence-mean/token-mean aggregation. In Equation (1), these conventions correspond to $\alpha_n = Z_n$ and $\alpha_n = 1$, respectively. Beat Block Hammer Tree-GRPO and **REGPO** runs likewise use ten warmup updates before enabling auxiliary branches.

J.2.4 ADDITIONAL RESULTS AND COMPUTATIONAL COST

Learning curves. Figure 6 reports evaluation performance across training updates for Beat Block Hammer, MiniWoB++, and WebShop. The ALFWorld learning curves appear in panel (b) of Table 1 in the main paper.

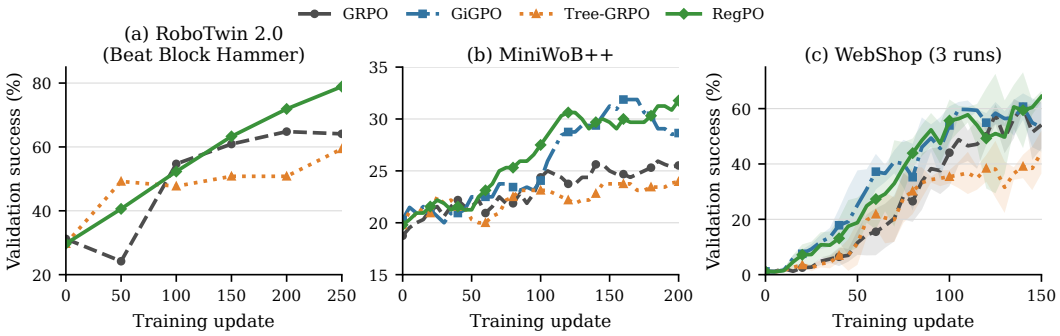

Figure 6: **Agentic benchmark learning curves. (a)** RoboTwin 2.0 (Beat Block Hammer) ID success on 128 fixed base trajectories through update 250. **(b)** MiniWoB++ success on 64 fixed evaluation trajectories through update 200, shown with a centered five-point moving average (a 25-update window). **(c)** WebShop test-partition success rate through update 150, with means and standard-deviation shading across three runs. Beat Block Hammer and MiniWoB++ show one training run per method; GiGPO is omitted from Beat Block Hammer because the required repeated matched environment states are unavailable.

Branching configurations. We evaluate different branching configurations on MiniWoB++ and WebShop to examine sensitivity to rollout construction. Table 9 compares base branching and balanced trees with different numbers of base trajectories and branching depths.

Benchmark	Branching configuration	Tree-GRPO	**REGPO**
MiniWoB++	Base branching (selected)	25.0	35.9
MiniWoB++	Balanced, 3 base trajectories, depth 2	29.7	31.2
WebShop	Base branching (selected)	48.1	65.9
WebShop	Balanced, 2 base trajectories, depth 3	44.0	55.1

Table 9: **Development comparison of branching configurations.** Each result is an evaluation success rate in percent, using the held-out ID split for MiniWoB++ and the test split for WebShop. For **REGPO**, entries use the final training checkpoint; for Tree-GRPO, entries report the highest score observed on the corresponding evaluation split within the training horizon. These development comparisons use training seed 0. Table 1 uses the same method-specific checkpoint reporting convention with its stated aggregation. Rows marked "selected" identify the rollout configurations used for the final comparison and detailed in Table 8. Within each row, Tree-GRPO and **REGPO** use the same rollout configuration.

Rollout-generation cost. We additionally profile ALFWorld rollout generation in a one-GPU test run using `verl-agent`; Table 10 reports the results. Each profile uses eight task instances and allows up to 64 rollouts to run in parallel. GRPO uses 16 trajectories per group. Base branching uses eight base trajectories and samples two actions at each of up to two selected branching prefixes. The balanced tree uses seven base trajectories, two sampled actions at each branching prefix, and depth two. These profiling configurations are separate from those in Table 8.

Rollout configuration	Training step	Rollout generation	Model-generation calls	Average outputs per call
GRPO, 16 trajectories per group	174.9	113.8	60	64.0
Base branching, eight base trajectories	218.9	156.3	74	49.3
Balanced tree, seven base trajectories, depth two	236.8	178.1	88	38.9

Table 10: **ALFWorld rollout-generation.** This is a one-GPU test run. Counts and times are reported per training batch, and times are in seconds. Each profile uses eight task instances, allows up to 64 rollouts to run in parallel, and applies no additional rollout parallelism cap.

The three configurations generate similar numbers of agent outputs. Base branching and the balanced tree use more model-generation calls with fewer outputs per call, and their training-step times are 218.9 and 236.8 seconds, respectively, compared with 174.9 seconds for GRPO.

J.2.5 PROMPTS

Prompt templates. We follow the ReAct prompt design (Yao et al., 2023), interleaving reasoning and environment actions with observation feedback. The prompts in Figures 7 and 8 adapt this design to ALFWorld and WebShop, with environment-specific observations, action spaces, and output formats. Fields in braces are filled at runtime. The displayed ALFWorld and WebShop templates include recent interaction history. For MiniWoB++, we adapt the system-prompt design of WebVoyager (He et al., 2024) and the message structure and action interface of the BrowserGym demo agent (de Chezelles et al., 2025). The system and user-message templates appear in Figures 9 and 10.

Observations and interaction history. For MiniWoB++, each observation contains the current accessibility-derived Document Object Model (DOM), a Hypertext Markup Language (HTML) rendering capped at 4,400 tokens, and a compact interaction history. We retain at most 150 characters for each previous action or execution error and at most 1,000 error characters in total. Actions use

the BrowserGym bid-based interface. For ALFWorld and WebShop, the prompt retains the two most recent interaction turns.

ALFWorld prompt

```
You are an expert agent operating in the ALFRED Embodied Environment.
Your task is to: {task_description}
Prior to this step, you have already taken {step_count} step(s).
Below are the most recent {history_length} observations and the
corresponding actions you took: {action_history}
You are now at step {current_step} and your current observation is:
{current_observation}
Your admissible actions of the current situation are:
[{admissible_actions}].
Now it's your turn to take an action.
You should first reason step-by-step about the current situation.
This reasoning process MUST be enclosed within <think> </think> tags.
Once you've finished your reasoning, you should choose an admissible
action for current step and present it within <action> </action> tags.
```

Figure 7: The prompt template of ALFWorld agents.

WebShop prompt

```
You are an expert autonomous agent operating in the WebShop e-commerce
environment.
Your task is to: {task_description}.
Prior to this step, you have already taken {step_count} step(s).
Below are the most recent {history_length} observations and the
corresponding actions you took: {action_history}
You are now at step {current_step} and your current observation is:
{current_observation}.
Your admissible actions of the current situation are:
[
{available_actions}
].
Now it's your turn to take one action for the current step.
You should first reason step-by-step about the current situation, then
think carefully which admissible action best advances the shopping
goal. This reasoning process MUST be enclosed within <think> </think>
tags.
Once you've finished your reasoning, you should choose an admissible
action for current step and present it within <action> </action> tags.
```

Figure 8: The prompt template of WebShop agents.

MiniWoB++ system prompt

```
Imagine you are a robot browsing the web, just like humans.
You need to complete a task by iteratively choosing the best next
action.
Your answer will be executed by a program.
All valid actions are listed in the Action Space section.
End every response with exactly one plain, unfenced line such as
Action:  click("15").

* Action Guidelines *
1) Execute only one action per iteration.
2) When an action targets a page element, copy the exact bid attribute
shown in the current DOM. For <button bid="13" id="subbtn">, use
click("13"), not click("subbtn").
3) Do NOT use HTML ids, CSS selectors, visible text labels, or
coordinate strings as element identifiers.
4) STRICTLY avoid repeating the same action if the webpage remains
unchanged.
5) Continuous use of the Wait action is NOT allowed.

* MiniWoB Guidelines *
1) Read the MiniWoB task and the current DOM before choosing an
action.
2) Use only the current DOM for element bids; do not reuse bids from
previous pages unless they still appear.
3) If the previous action produced an error, choose a different valid
action.

This reasoning process MUST be enclosed within <think> </think> tags.
After closing </think>, end the response with exactly one plain action
line, for example Action:  click("15").  Do not wrap the action in
backticks or a Markdown code fence.

Your task is to {task}
```

Figure 9: The system prompt template of MiniWoB++ agents.

MiniWoB++ user message

```
[Task]
{task_description}

[Actions before the windows]
Step {i}:
Action: {action_i}
Error: {error_i}
[End of Actions before the windows]

[Recent Observations and Actions window]
Below are the most recent {n} actions and action errors:
Step {i}:
Action: {action_i}
Error: {error_i}
[End of Recent Observations and Actions window]

[Turn Instruction]
# Previous Attempts (optional)

You've already tried a few actions, but the web page did not change.

Below is the action set you've chosen:

Attempt {j}:
Action: {action_j}
Error Message: {error_j}

Based on the above history, please think carefully and choose another action.

You are now at step {current_step} and your current observation:

{current_observation}

{action_space_description}

Now it's your turn to take an action.
You should first reason step-by-step about the current situation. This
reasoning process MUST be enclosed within <think> </think> tags. After closing
</think>, end with exactly one plain action line such as Action: click("15").
Do not use Markdown fences.
```

Figure 10: The user-message prompt template of MiniWoB++ agents.

K Additional Results on Finite Tree-Structured MDPs

K.1 A Random-Tree Mechanism Example

Consider a representative instance with one high-return leaf on the action sequence $(1, 1, 0, 2)$, where actions are indexed from zero. At initialization, $Q^{\pi_{\theta_0}}(\xi_0, 1) - Q^{\pi_{\theta_0}}(\xi_0, 2) = -0.282$, whereas $Q^\star(\xi_0, 1) - Q^\star(\xi_0, 2) = 0.228$. Thus the current-policy and optimal action values rank these actions differently. Table 11 reports the results at training step 100.

Measurement at training step 100	$\lambda = 1$	$\lambda = 1/2$	$\lambda = 0$
Probability of selecting root action 1 ↑	1.86	88.42	**96.03**
Probability of reaching prefix $(1, 1, 0)$ ↑	0.43	83.94	**92.56**
$Q^\pi(\xi_0, 1)$ ↑	0.443	0.861	**0.871**
Expected root return ↑	0.624	0.828	**0.858**

Table 11: **A random-tree instance connecting an initial action-ranking disagreement to action selection, downstream reach, and action value.** Probabilities are in percent; action values and expected return use the fixed leaf returns. Up arrows indicate that higher values are better; red denotes the best value in each row.

K.2 Lower Hard-Step Optimal-Action Probability

Under the primary initialization in Figure 2, GRPO samples a reward-one rollout in every run but does not satisfy $Q^\pi(\xi, \mathrm{Hard}) \geq c$ in any run for $H \geq 14$ within the 1,000-step horizon.

Figure 11 reports an initialization with lower Hard-action probabilities. Before a reward-one Hard rollout appears in the current rollout tree, the estimated Hard root value is zero, so the regret-based advantage estimate can disfavor Hard. We use $q_0 = 0.46$, $H \in \{6, 8, 10, 12, 14, 16, 20, 30\}$, exactly 1,000 training steps, and 500 seeds per setting. We measure the first training step at which $Q^\pi(\xi, \mathrm{Hard}) \geq c$, treating runs that do not reach this threshold within 1,000 steps as right-censored.

The per-node branching factor b differs from the independent-rollout count m in Proposition 3. Let $w_{H+1} = 1$ and let w_k be the probability that one visit to Hard prefix k samples at least one reward-one leaf. Under this b-way recursion,

$$w_k = 1 - (1 - q_k w_{k+1})^b, \qquad \mathbb{P}(\text{one tree samples a reward-one leaf}) = 1 - (1 - p_0 w_1)^b.$$

For $b = 10$, this probability at $H = 30$ is 0.9990 under the primary initialization and 0.9989 under the lower-probability initialization. Thus the tree sampler samples a reward-one leaf with near-unit probability even though the number 10 is far smaller than the independent-rollout count required by Proposition 3.

Across the lower-probability sweep, every condition samples a reward-one Hard rollout in every run. The current-policy and pure regret-based components satisfy $Q^\pi(\xi, \mathrm{Hard}) \geq c$ in every run, while GRPO does so only at $H = 6$. The current-policy component requires approximately $2.2\times$ as many training steps as pure regret-based feedback at $H = 10$ and $3.5\times$ as many at $H = 30$.

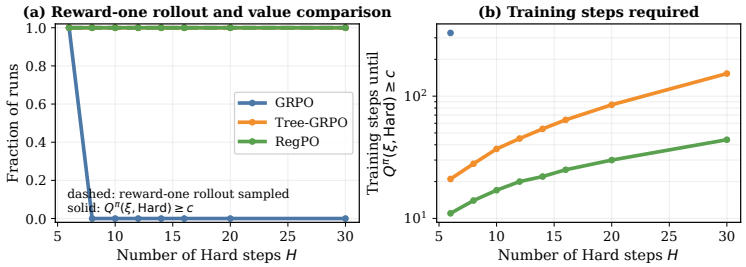

Figure 11: **End-to-end Hard-Safe training from a lower-probability initialization.** **(a)** Dashed curves show whether a reward-one Hard rollout has been sampled, and solid curves show the fraction of runs satisfying $Q^\pi(\xi, \text{Hard}) \geq c$. **(b)** Median first training step at which this inequality holds; missing points indicate that fewer than half of the runs satisfy it within $1{,}000$ training steps. Curves summarize 500 seeds.

L ALFWORLD POLICY BEHAVIOR EVALUATION

This section reports the evaluation protocol and additional results for Figure 4. It uses the 96 frozen selected prefixes from Section 4.2.2, with three prefixes from each of 32 held-out ID tasks.

Forced-action continuations. The action-ranking evaluation contains four candidate actions at every selected prefix and eight independent continuations per evaluated policy and action. The next action is fixed to the candidate action, so success measures execution after that action. The 384 prefix–action pairs also identify 279 actions for which the GPT-5.5 continuation succeeds while none of the eight Update-10 continuations succeeds. The analysis of strict disagreement pairs selects 153 action pairs from 65 prefixes and 31 tasks for which the Update-10 mean action-value estimate and the stronger-policy proxy give opposite rankings. In each pair, a^+ denotes the action preferred by the stronger-policy proxy. This selection uses only the forced-action data.

Policy-selected continuations. Starting from each frozen prefix, the evaluated policy selects the next action. We sample eight first actions. After each nonterminal first-action occurrence, we sample eight second actions, and after each nonterminal two-action occurrence, we sample four complete continuations. Repeated actions are retained, and each sampled continuation is weighted by its probability under this $8 \times 8 \times 4$ sampling procedure: continuations reaching the final sampling stage have weight $1/256$, while continuations terminating after the first or second action retain weights $1/8$ or $1/64$. All three policies use sampling temperature 0.4 and the same remaining allowance under a total limit of 30 environment actions.

For action selection, we estimate the probability assigned to a^+ in each of the 153 strict disagreement pairs by its frequency among the eight first actions.

Intermediate-state definitions. Fixed task-family rules instantiate candidate intermediate-state conditions from each task's symbolic goal parameters. Table 12 lists the exact conditions.

Task family	Candidate intermediate-state conditions
Pick and place	Reach a target object's location; hold a target object; reach a destination of the required type while holding a target object.
Heat, clean, or cool, then place	Hold a target object; reach the corresponding processing station while holding it; hold the transformed object; reach the destination while holding the transformed object.
Examine using a light	Reach a target object's location; hold a target object; reach the required lamp while holding a target object.
Place two objects	Hold a target object; place one target object in the destination; hold a distinct second target object while the first remains placed; reach the same destination while holding the second object.

Table 12: **Fixed intermediate-state rules for ALFWorld task families.** "Reach" denotes symbolic co-location of the agent and the specified object, station, or destination. The task's goal parameters instantiate the object, transformation, and destination types.

Within each condition, existential bindings require the symbolic facts to refer to the same target object. The two-object conditions additionally require distinct objects and the same destination. At each selected prefix, we retain a candidate condition only if it is false at that prefix and becomes true at a nonterminal state in a fixed successful GPT-5.5 continuation. A sampled continuation counts a condition once when its symbolic environment state first satisfies it. This gives 269 eligible prefix–condition pairs from 90 prefixes. Success is evaluated from all 96 prefixes. We first average within each task and then across tasks. Confidence intervals resample whole tasks, using 2,000 resamples for the analysis of strict disagreement pairs and 4,000 for the other rows.

Figure 12 measures intermediate-state reach after 25%, 50%, 75%, and 100% of the remaining action budget. The **REGPO**–Tree-GRPO difference is 29.00 percentage points after the first quarter and 30.22 points over the full budget.

Measurement	Update 10	Tree-GRPO	**REGPO**
Selection of proxy-preferred action	18.62 [11.33, 26.69]	24.15 [14.52, 35.15]	**38.19 [28.04, 49.14]**
Task-relevant intermediate-state reach	19.19 [12.15, 26.90]	53.71 [40.45, 67.34]	**83.93 [72.13, 94.37]**
Forced-action success	7.68 [3.03, 13.67]	53.68 [40.33, 67.06]	**76.14 [64.84, 87.08]**
Policy-selected success	6.41 [2.44, 11.51]	49.99 [35.92, 64.33]	**81.49 [69.39, 92.81]**

Table 13: **Policy behavior from frozen ALFWorld prefixes (%).** Brackets give task-bootstrap 95% confidence intervals. The four rows use the respective cohorts defined above.

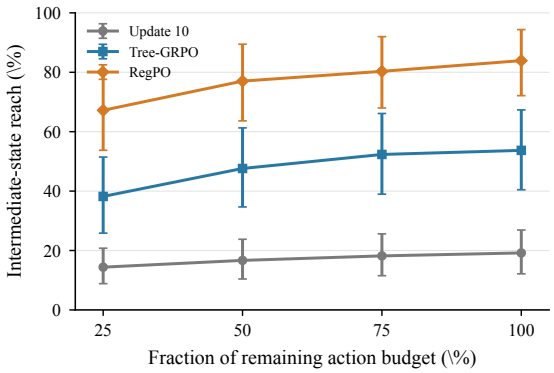

Figure 12: **ALFWorld intermediate-state reach by action budget.** Curves show task-level means and error bars show task-bootstrap 95% confidence intervals over 32 tasks.

Illustrative decisions. For "put some cellphone on bed," the Update-10 mean action-value estimate ranks `go to bed 1` below `inventory`, whereas the stronger-policy proxy gives the opposite ranking. Update 10 selects `go to bed 1` in $2/8$ trials; Tree-GRPO and **REGPO** select it in $8/8$. The probability of reaching a bed while holding the cellphone is 74.61%, 100%, and 100%, respectively, and forced-action success after `go to bed 1` is $5/8$, $8/8$, and $8/8$.

For "put a hot mug in cabinet," at one selected prefix Tree-GRPO selects `go to fridge 1` in $8/8$ trials, while **REGPO** selects `go to microwave 1` in $8/8$. Their probabilities of reaching the microwave while holding the mug are 0% and 100%, respectively. Both policies succeed in $8/8$ forced-action continuations after `go to microwave 1`, separating selection of the action from execution after that action.

M MiniWoB++ Policy Behavior Evaluation

We evaluate action selection, intermediate-state reach, forced-action success, and policy-selected success in MiniWoB++ using the definitions in Section 4.2.2. This diagnostic compares Update 10, trained Tree-GRPO, and trained **RegPO** on 32 held-out instances from eight selected task families, starting from 86 selected prefixes held fixed during evaluation.

Prefixes come from successful scripted references. Each prefix has four shared candidate actions: the scripted reference action, together with deterministic alternatives derived from the webpage's document object model (DOM) or control actions, giving 344 prefix–action pairs. In ALFWorld, prefixes come from trained **RegPO** trajectories and alternative candidates come from GPT-5.5. For each evaluated policy and candidate action, we fix the next action to the candidate and sample eight independent continuations with subsequent actions selected by the policy. We denote their mean discounted return by $\widehat{Q}^{\pi}_{\mathrm{mean}}(\xi, a)$. One GPT-5.5 continuation from the same prefix and forced action provides $\widehat{Q}_{\mathrm{GPT}}(\xi, a)$.

Action selection, downstream reach, and continuation execution. As in Section 4.2.2, a strict disagreement pair (a^+, a^-) satisfies

$$\widehat{Q}^{10}_{\mathrm{mean}}(\xi, a^+) < \widehat{Q}^{10}_{\mathrm{mean}}(\xi, a^-), \qquad \widehat{Q}_{\mathrm{GPT}}(\xi, a^+) > \widehat{Q}_{\mathrm{GPT}}(\xi, a^-).$$

This gives 88 pairs from 41 prefixes, 19 instances, and seven families. They contain 59 distinct preferred prefix–action pairs, counted with their multiplicity across strict disagreement pairs. Using the fixed pseudo-posterior configuration without MiniWoB++-specific tuning, the analogous gap inequality holds for 84.1% of the disagreement pairs:

$$\widehat{Q}^{\mathrm{post},10}(\xi, a^+) - \widehat{Q}^{\mathrm{post},10}(\xi, a^-) \geq \widehat{Q}^{10}_{\mathrm{mean}}(\xi, a^+) - \widehat{Q}^{10}_{\mathrm{mean}}(\xi, a^-).$$

[2]

Intermediate-state conditions are frozen task-family DOM predicates. At a selected prefix ξ, a condition M is eligible if it is false at that prefix and becomes true at a nonterminal state in a successful scripted reference. ALFWorld uses symbolic goal conditions witnessed in GPT-5.5 continuations. Each condition is counted once per sampled continuation, at its first occurrence. This gives 119 eligible prefix–condition pairs from 31 instances and all eight families.

Let $S(\tau)$ indicate *positive-reward success*: the continuation receives positive environment reward, which can include partial credit. In the notation of Section 4.2.2, action selection is $\pi(a^+ \mid \xi)$, intermediate-state reach is $\Pr_{\tau \sim P^{\pi}_{\rho}(\cdot|\xi)}[\exists\, \xi' \in \tau : M(\xi') = 1]$, forced-action success is $\Pr_{\tau \sim P^{\pi}_{\rho}(\cdot|\xi,a)}[S(\tau) = 1]$, and policy-selected success is $\Pr_{\tau \sim P^{\pi}_{\rho}(\cdot|\xi)}[S(\tau) = 1]$. Full-bank forced-action success uses all 344 prefix–action pairs; policy-selected success uses all 86 prefixes, both covering all 32 instances and eight families.

For the forced-action success measurement in Figure 13, we measure *perfect success*, defined as full environment credit, on a cohort of 136 prefix–action pairs from 29 instances and eight families. For each retained pair, the GPT-5.5 continuation achieves full credit and none of the eight Update-10 continuations receives positive reward. Let $S_{\mathrm{perfect}}(\tau)$ indicate full credit. For this measurement, we replace S by S_{perfect} and average over the retained pairs.

In Figure 13, the **RegPO** point estimates exceed those of Tree-GRPO by 4.48 percentage points for action selection, 17.61 points for intermediate-state reach, 12.78 points for forced-action perfect success on the 136 prefix–action pairs, and 6.61 points for policy-selected success.

Sampling and aggregation. For policy-selected continuations, we sample eight first actions, eight second actions after each nonterminal first-action occurrence, and four complete continuations after each nonterminal two-action occurrence. Repeated actions retain their probability mass under this $8 \times 8 \times 4$ sampling procedure: continuations reaching the final stage have weight $1/256$, while those terminating after the first or second action retain weights $1/8$ or $1/64$. We estimate action selection,

[2] For MiniWoB++, $K(\xi)$ is the number of concrete actions enumerated by the frozen DOM-based candidate generator before four-action truncation.

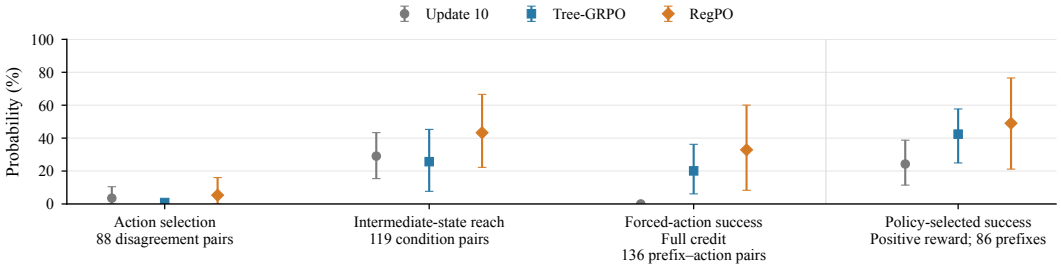

Figure 13: **Policy behavior from frozen selected prefixes in MiniWoB++.** Points show equal-family means and error bars show family-bootstrap 95% confidence intervals. Action selection uses 88 strict disagreement pairs; intermediate-state reach uses 119 eligible prefix–condition pairs. Forced-action success measures perfect success on 136 prefix–action pairs with GPT full credit and $0/8$ Update-10 positive-reward successes; policy-selected success reports positive-reward success over all 86 prefixes.

$\pi(a^+ \mid \xi)$, by the frequency of a^+ among the eight first actions. All continuations respect the remaining allowance under a total limit of ten environment actions, compared with 30 in ALFWorld.

We average eligible units within each instance, instances within each family, and then weight families equally. We estimate uncertainty using 10,000 paired whole-family bootstrap resamples; ALFWorld uses task-bootstrap resampling. The figure shows 95% confidence intervals, and intervals for policy differences use paired resampling.