MILEGPO: MILESTONE INFERENCE WITH LOCAL EVIDENCE FOR GRAPH-BASED POLICY OPTIMIZATION OF LONG-HORIZON LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Credit assignment is challenging in long-horizon agentic reinforcement learning, where supervision typically comes from discrete final rewards. Existing methods refine trajectory-level signals into step-level credits through anchor-based step grouping or graph-based discriminative advantage estimation, but still overlook nontrivial milestones in the intermediate process. To address this, we propose **MileGPO** (Milestone Inference with Local Evidence for Graph-Based Policy Optimization), developed through three nested credit designs over grouped on-policy rollouts. First, **Milestone Discovery** identifies candidate milestones from successful rollouts and recurring traps from failed ones. Second, **Reliability-Calibrated Shaping (RCS)** assigns each candidate a confidence-based weight: high-confidence milestones receive stronger positive credit, high-confidence traps incur stronger penalties, and uncertain candidates are down-weighted. Finally, **Progress-Contrastive Calibration (PCC)** determines which candidate milestones deserve stronger credit by checking whether they mark local progress and whether the transition reaching them outperforms other observed choices from the same state. Using only existing grouped on-policy rollouts and rewards, MileGPO provides fine-grained process-level credit without external supervision, auxiliary models, or additional environment interaction. Extensive experiments demonstrate that MileGPO achieves superior performance on both the ALFWorld and WebShop benchmarks, while the marginal performance gap between the in-distribution and out-of-distribution ALFWorld test sets further supports its generalization to the ALFWorld out-of-distribution split. Cumulative ablations support the incremental benefits of RCS and PCC in the evaluated configurations, while rollout diagnostics characterize how outcome-derived signals modify intermediate credit allocation.

1 INTRODUCTION

In recent years, large language models (LLMs) have evolved from single-turn text generators into capable agents that can reason, invoke external tools, and execute tasks over extended horizons (Yao et al., 2023; Schick et al., 2023; Wang et al., 2023). Such agents are increasingly applied to long-horizon tasks such as web navigation and embodied instruction following (Deng et al., 2023; Zhou et al., 2024a; Shridhar et al., 2021), where successful completion depends on a coherent sequence of interdependent decisions. Reinforcement learning (RL) provides a natural framework for learning to improve such agents from final rewards at the task-level (Ouyang et al., 2022; Guo et al., 2025; Wang et al., 2025). However, when supervision is reduced to a final reward, assigning credit to intermediate decisions becomes challenging. As illustrated in Figure 1, trajectory-level methods assign the same final-reward-derived advantage to all actions in a rollout, while step-level grouping methods compare actions taken under the same state and provide finer-grained credit (Williams, 1992; Schulman et al., 2017; Feng et al., 2025). GraphGPO (Cheng et al., 2026) further organizes grouped on-policy rollouts into a task-local transition graph, where each transition represents a state–action–next-state interaction step, and propagates credit according to each state's shortest-path distance to a successful final state. Although this graph structure provides dense intermediate supervision, final goal distance primarily captures *reachability*, rather than *reliable progress*: states with the same

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

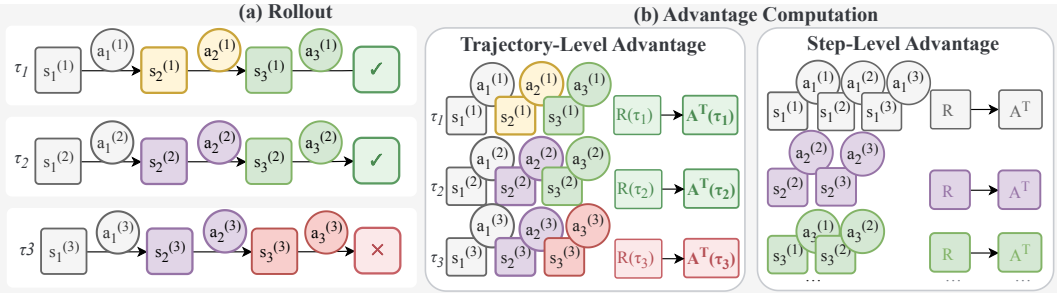


Figure 1: Trajectory-level and same-state step-level credit for two successful rollouts and one failed rollout. Here, τ_i is rollout i; $s_t^{(i)}$, $a_t^{(i)}$, $R(\tau_i)$, and $A^{\mathrm{T}}(\tau_i)$ denote its state, action, final reward, and trajectory-level advantage. Matching colors mark the same state across rollouts.

distance can have substantially different probabilities of leading to success, while important intermediate stages may receive weak credit simply because they remain far from the final goal.

As shown in Figure 2, in early GraphGPO rollouts on ALFWorld and WebShop, about 74% of transitions originate from states shared across rollouts, and most shared states contain multiple actions, providing recurring states and sibling branches from which to infer intermediate credit. However, the final-goal distance assigns the same credit to 54.4% and 72.7% of same-state action pairs, respectively, while identical transitions can appear in both successful and failed trajectories. Thus, the rollout graph offers evidence for credit assignment, but exploiting it raises three questions: how to discover, weight, and validate intermediate credit anchors.

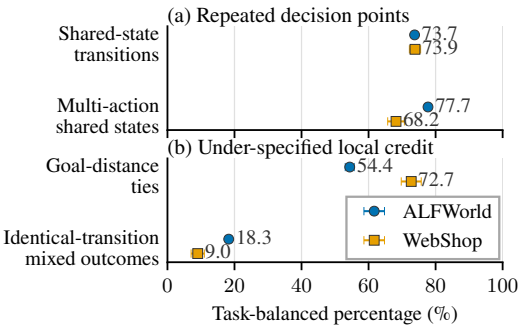


Figure 2: Ambiguity in early GraphGPO rollouts.

To address these questions, we propose the **MileGPO** (Milestone Inference with Local Evidence for Graph-Based Policy Optimization), illustrated in Figure 3, with a sequence of nested credit designs: (1) **Milestone Discovery (MD)**: Not every state on a successful trajectory marks meaningful progress. MD uses final rewards and recurring rollout structures to identify candidate milestones from successful trajectories and recurring traps from failed ones. (2) **Reliability-Calibrated Shaping (RCS)**: Candidate anchors have unequal final-reward evidence. RCS weights their positive or negative influence by empirical reliability, strengthening well-supported candidates while suppressing uncertain ones. (3) **Progress-Contrastive Calibration (PCC)**: Final-reward association alone cannot distinguish genuine progress from incidental correlation. PCC calibrates candidate scores using local advancement and branch-contrastive credit (BCC), which compares sibling transitions from the same state. MileGPO modifies only step-level advantage estimation and requires no external process annotations or auxiliary inference overheads.

We evaluate MileGPO on ALFWorld and WebShop, where it consistently outperforms the locally reproduced GiGPO and GraphGPO baselines across different model scales(1.5B and 7B). At 1.5B, it improves ALFWorld OOD success and WebShop success over GraphGPO by 3.13 and 3.78 %, respectively, with gains also observed at 7B. Cumulative ablations show that RCS improves performance over uniform MD and that adding PCC yields further gains, with a larger increase on ALFWorld OOD than ID. On fixed WebShop rollouts, RCS assigns higher credit to transitions from successful trajectories more often than uniform MD among pairs tied under final-goal-distance credit. The share of candidates satisfying PCC's local-evidence criteria also increases during training. These findings support the task-level utility of the nested design and characterize its intermediate credit allocation.

Our main contributions are as follows:

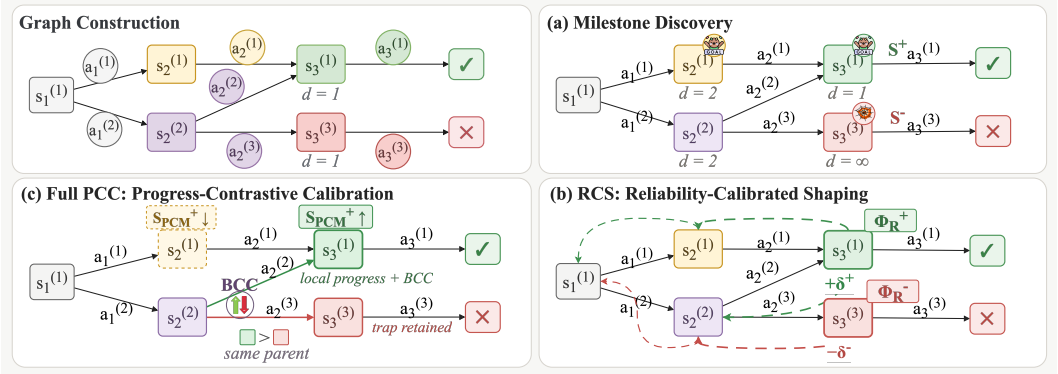


Figure 3: Overview of MileGPO. Squares denote canonicalized states, circles denote sampled actions, and check and cross marks indicate success and failure. Here, d is directed shortest-path distance (∞ if unreachable); $\mathcal{S}^+$ and $\mathcal{S}^-$ are the milestone and trap sets; Φ_{R}^+ and Φ_{R}^- are their score-weighted potentials; and δ^+ and δ^- are the corresponding increases along a transition. MD discovers candidates, RCS weights and propagates them, and PCC adjusts positive scores using local progress and BCC comparisons between branches sharing an observation anchor.

- **Revealing unreliable intermediate credit.** We reveal that final-goal-distance credit leaves many same-state branches indistinguishable and that success-visited states do not necessarily represent reliable progress.
- **Proposing a rollout-native policy optimization algorithm.** We introduce MileGPO, which learns from rollout graphs and on-policy rewards without external supervision, auxiliary models, or additional environment interaction.
- **Achieving strong empirical performance.** MileGPO achieves superior results on ALFWorld and WebShop, with ablations and diagnostics validating credit calibration.

2 Related Work

Reinforcement Learning for LLM Agents. LLM post-training builds on policy-gradient estimators such as REINFORCE, GAE, and PPO (Williams, 1992; Schulman et al., 2018; 2017). RLHF learns human-preference rewards (Stiennon et al., 2020; Ouyang et al., 2022; Bai et al., 2022), whereas DPO bypasses an explicit reward-model RL loop (Rafailov et al., 2024). For verifiable reasoning, RLOO, GRPO, and DAPO compare sampled responses without a value model (Kool et al., 2019; Ahmadian et al., 2024; Shao et al., 2024; Yu et al., 2025), as exemplified by DeepSeek-R1, while process supervision adds intermediate feedback (Guo et al., 2025; Lightman et al., 2024). These objectives extend to agents that interact with search engines, websites, embodied environments, and other agents over multiple turns (Zhou et al., 2025). Reflection, search-guided collection, and hierarchical actor–critic methods use verbal memory, tree search, or learned values (Shinn et al., 2023; Putta et al., 2024; Zhou et al., 2024b), while Agent Lightning decouples agent execution from training (Luo et al., 2025). Recent methods refine credit granularity: GiGPO compares actions at recurring states, HGPO conditions comparisons on interaction history (He et al., 2026), StepPO, BiPACE, and Progress Advantage use step-aligned, action-conditioned, or policy-derived signals (Wang et al., 2026a;b; Oh et al., 2026), and graph-based methods merge grouped trajectories into task-local structures (Wang et al., 2026c).

Process-Level Credit Assignment Process reward models provide step-level supervision but require process labels and may degrade under distribution shift (Lightman et al., 2024; Wang et al., 2024; Zhang et al., 2025). Annotation-light alternatives estimate intermediate values through search or derive process signals from outcome rewards (Chen et al., 2024; Cui et al., 2026). TreeRPO and TreeRL share prefixes and compare branches through tree-structured rollouts, while GraphPO merges equivalent reasoning states during structured sampling (Yang et al., 2025; Hou et al., 2025; Zhan et al., 2026). These approaches introduce auxiliary process estimation or alter rollout collection through search and branching.

3 METHOD

3.1 PRELIMINARIES: GROUPED ROLLOUT GRAPH

For each task q, the current policy samples a group of K trajectories $\mathcal{T}_q = \{\tau_i\}_{i=1}^K$. Each trajectory $\tau_i = (s_{i,0}, a_{i,0}, \ldots, s_{i,T_i})$ has length T_i and a final environment reward R_i^{env}, and $y_i = \mathbf{1}[R_i^{\text{env}} > \theta_R]$ marks success under the task-specific threshold θ_R. Following GraphGPO (Cheng et al., 2026), we merge all trajectories for task q into a directed transition graph $\mathcal{G}_q = (\mathcal{V}_q, \mathcal{E}_q)$ whose nodes group identical observation anchors regardless of prompt history and whose edges $e = (u, v)$ are observed transitions. We write g for the successful final state and $d(u, v)$ for the directed shortest-path distance, with $d(u, v) = \infty$ when v is unreachable. For goal-distance computations, the implementation uses a finite fallback for unreachable states. GraphGPO then assigns each observed transition a dense return according to the distance from its destination to g:

$$r^{\text{G}}(u, v) = c\,\gamma_G^{d(v,g)}, \tag{1}$$

where c sets the return scale and $\gamma_G \in (0, 1]$ controls decay with distance. This return captures reachability but cannot distinguish destinations at the same graph distance; ***MileGPO preserves it and supplements it with intermediate credit as shown in Figure 3.***

3.2 MD: MILESTONE DISCOVERY

MD uses the outcome rewards and transition structure of the rollout group to discover two kinds of intermediate targets. A *milestone candidate* is a nonfinal state visited by successful trajectories. A *trap candidate* is a state observed only in failed trajectories that either appears across failures or is revisited. MD first scores these states and then propagates every selected state with equal weight.

3.2.1 FINDING CANDIDATES.

Let $\mathcal{T}_q^+ = \{\tau_i : y_i = 1\}$ and $\mathcal{T}_q^- = \{\tau_i : y_i = 0\}$ be the successful and failed trajectories for task q. For node v, let $\mathcal{T}(v)$ contain trajectories visiting v (each counted once), with $\mathcal{T}^+(v)$ and $\mathcal{T}^-(v)$ denoting successful and failed subsets. We compute the group- and node-conditional success rates as

$$p_q^+ = \frac{|\mathcal{T}_q^+|}{|\mathcal{T}_q|}, \qquad p^+(v) = \frac{|\mathcal{T}^+(v)|}{|\mathcal{T}(v)|}. \tag{2}$$

To score a milestone candidate v, MD first compares its conditional success rate $p^+(v)$ with the rollout-group success rate p_q^+. We retain only positive differences:

$$\ell^+(v) = \max\left(p^+(v) - p_q^+, 0\right). \tag{3}$$

Thus, $\ell^+(v) > 0$ only when trajectories visiting v have a higher success rate than the average across the rollout group. We separately measure how broadly v appears across successful trajectories:

$$m(v) = \frac{|\mathcal{T}^+(v)|}{\max(|\mathcal{T}_q^+|, 1)}. \tag{4}$$

Here, $m(v)$ is the fraction of sampled successful trajectories that visit v. We also define $C(v)$ as the sum of the in-degree and out-degree of v in $\mathcal{G}_q$, divided by the largest such sum over success-visited states, so that $C(v) \in [0, 1]$ measures how central v is among the states that successful trajectories traverse. MD combines these three quantities into the initial milestone score:

$$S_0^+(v) = w_s \ell^+(v) + w_m m(v) + w_c C(v). \tag{5}$$

Let $p_q^- = 1 - p_q^+$ be the group failure rate and let $p^-(v) = |\mathcal{T}^-(v)|/|\mathcal{T}(v)|$ be the failure rate among trajectories that visit v. For a state no successful trajectory visits, we keep only the amount by which this rate exceeds the group average: $\ell^-(v) = \max(p^-(v) - p_q^-, 0)$. We also compute the fraction of failed trajectories that visit v as $f^-(v) = |\mathcal{T}^-(v)|/\max(|\mathcal{T}_q^-|, 1)$. $L(v) = n_{\text{re}}(v)/\max(|\mathcal{T}(v)|, 1)$ is the average number of revisits per visiting trajectory, where $n_{\text{re}}(v)$ counts visits to v after its first occurrence in each trajectory. MD combines these quantities into the trap score:

$$S^-(v) = w_f \ell^-(v) + w_l f^-(v) L(v), \tag{6}$$

where w_f and w_l weight the excess failure rate and the recurrence of v in failed behavior. We denote the selected milestones and traps by $\mathcal{S}^+$ and $\mathcal{S}^-$, respectively.

3.2.2 UNIFORM TARGET PROPAGATION.

MD assigns every selected state unit weight and propagates it through the graph distance. With decay $\omega \in (0, 1]$, the two potential maps are

$$\Phi_{\mathrm{S}}^{+}(s) = \max_{v \in \mathcal{S}^+} \omega^{d(s,v)}, \qquad \Phi_{\mathrm{S}}^{-}(s) = \max_{v \in \mathcal{S}^-} \omega^{d(s,v)}. \tag{7}$$

Here, $\Phi_{\mathrm{S}}^{+}(s)$ and $\Phi_{\mathrm{S}}^{-}(s)$ measure how close s is to the nearest milestone and the nearest trap, respectively, with the maximum retaining the largest distance-decayed target value. RCS next replaces the unit weights with candidate scores.

3.3 RCS: RELIABILITY-CALIBRATED SHAPING

RCS uses the same MD candidates, weighted by their task-wise max-normalized scores $\overline{S}_0^+$ and $\overline{S}^-$:

$$\Phi_{\mathrm{R}}^{+}(s) = \max_{v \in \mathcal{S}^+} \overline{S}_0^{+}(v)\omega^{d(s,v)}, \qquad \Phi_{\mathrm{R}}^{-}(s) = \max_{v \in \mathcal{S}^-} \overline{S}^{-}(v)\omega^{d(s,v)}. \tag{8}$$

Thus $\Phi_{\mathrm{R}}^{+}(s)$ and $\Phi_{\mathrm{R}}^{-}(s)$ are the score-weighted counterparts of $\Phi_{\mathrm{S}}^{+}(s)$ and $\Phi_{\mathrm{S}}^{-}(s)$, so that higher-scoring targets produce larger potentials. We keep milestones and traps separate because their contributions have opposite signs. RCS next checks whether a transition increases either potential. Writing Φ^+ and Φ^- for the positive and negative potentials in use, which are Φ_{R}^{+} and Φ_{R}^{-} here, a transition from s_t to s_{t+1} discards negative changes:

$$\delta_t^+ = \max\left(\gamma_\Phi w_+ \Phi^+(s_{t+1}) - w_+ \Phi^+(s_t), 0\right), \tag{9}$$

$$\delta_t^- = \max\left(\gamma_\Phi w_- \Phi^-(s_{t+1}) - w_- \Phi^-(s_t), 0\right). \tag{10}$$

Here, δ_t^+ and δ_t^- are the discounted increases in milestone and trap potential along the transition, w_+ and w_- scale these changes, and γ_Φ discounts the next-state potential. The return uses $\delta_t^+ - \delta_t^-$, rewarding movement toward milestones and penalizing movement toward traps. Transitions with the same final-goal distance can have different milestone or trap potential changes, allowing RCS to distinguish transitions that receive identical GraphGPO returns.

3.4 PCC: PROGRESS-CONTRASTIVE CALIBRATION

RCS uses group-level outcome rewards, so a state may score highly even when entering it makes little progress. PCC refines positive candidates with two transition-level scores. BCC compares branches sharing the same observation anchor, while local progress measures movement toward the goal and association with success. These scores update candidates rather than add rewards.

3.4.1 BRANCH-CONTRASTIVE CREDIT (BCC).

At a source observation anchor u, grouped rollouts may contain several distinct outgoing edges. We treat each observed edge $e = (u, v)$ as one sampled branch from u (Appendix A.5). BCC compares the success rate of this branch with those of the other branches sharing the same source anchor. Let $\mathcal{T}(e)$ contain the trajectories that use e, with $\mathcal{T}^+(e)$ and $\mathcal{T}^-(e)$ denoting its successful and failed subsets. Their rates are

$$p^+(e) = \frac{|\mathcal{T}^+(e)|}{|\mathcal{T}(e)|}, \qquad p^-(e) = \frac{|\mathcal{T}^-(e)|}{|\mathcal{T}(e)|}. \tag{11}$$

Let $\mathcal{B}(u)$ contain these sampled branches. For $e_i \in \mathcal{B}(u)$, the BCC margin is

$$\widetilde{c}_{\mathrm{bcc}}(e_i) = p^+(e_i) - \mu^+(e_i). \tag{12}$$

Here, $\mu^+(e_i)$ is the mean success rate of the branches in $\mathcal{B}(u) \setminus \{e_i\}$, that is, of the alternative branches sharing the same source anchor; we set $\widetilde{c}_{\mathrm{bcc}}(e_i) = 0$ when e_i is the only sampled branch. We divide each margin by the largest absolute margin in task q:

$$c_{\mathrm{bcc}}(e_i) = \frac{\widetilde{c}_{\mathrm{bcc}}(e_i)}{\max_{e \in \mathcal{E}_q} |\widetilde{c}_{\mathrm{bcc}}(e)|}, \qquad b(v) = \max_{e=(u,v)} \left\{ c_{\mathrm{bcc}}(e) : c_{\mathrm{bcc}}(e) > 0, \mathcal{T}^+(e) \neq \emptyset \right\}. \tag{13}$$

Thus, $b(v)$ summarizes the strongest observed success-rate preference among branches leading to v, with $b(v) = 0$ when no incoming transition satisfies both conditions.

3.4.2 LOCAL PROGRESS.

BCC compares branches but does not capture goal progress. For $e = (u, v)$, we combine distance reduction $\Delta d(e) = d(u, g) - d(v, g)$, success-rate gain $p^+(v) - p_q^+$, and excess failure rate $\ell^-(e) = \max(p^-(e) - \mu^-(e), 0)$, where $\mu^-(e)$ is the mean failure rate of sibling branches in $\mathcal{B}(u)$. Thus, $\ell^-(e)$ is the edge-level counterpart of $\ell^-(v)$ and falls back to $\max(p^-(v) - p_q^-, 0)$ when no sibling exists. The progress score is:

$$\psi(e) = \alpha_d \Delta d(e) + \alpha_s \left[p^+(v) - p_q^+\right] - \alpha_f \ell^-(e), \tag{14}$$

where α_d, α_s, and α_f weight the three terms. Let $\mathcal{E}^+_{\mathrm{pg}}(v)$ contain successful incoming transitions to v with $\psi(e) > 0$. PCC takes the largest score and normalizes it over the graph:

$$\widetilde{\psi}(v) = \max_{e \in \mathcal{E}^+_{\mathrm{pg}}(v)} \psi(e), \qquad \overline{\psi}(v) = \frac{\widetilde{\psi}(v)}{\max_{v' \in \mathcal{V}_q} \widetilde{\psi}(v')}. \tag{15}$$

$\widetilde{\psi}(v)$ is the best progress score among successful incoming transitions and $\overline{\psi}(v) \in [0, 1]$ its max-normalized value. Empty maxima and zero normalizers yield zero.

3.4.3 UPDATING CANDIDATE SCORES.

PCC combines the branch score and the progress score:

$$E(v) = w_{\mathrm{bc}} b(v) + w_{\mathrm{pg}} \overline{\psi}(v), \tag{16}$$

where w_{bc} and w_{pg} set their contributions. A retention indicator $z(v)$ marks candidates with local support: $z(v) = 1$ when $\overline{\psi}(v) > 0$, when coverage $m(v)$ reaches a threshold θ_m, or when $\kappa_{\mathrm{bc}} = 1$ and $b(v) > 0$; otherwise, $z(v) = 0$. The binary switch κ_{bc} controls whether branch evidence alone may retain a candidate, and BCC contributes to $E(v)$ regardless of which condition retains it. PCC updates the score as:

$$S^+_{\mathrm{pcc}}(v) = \begin{cases} S^+_0(v)\left[1 + w_{\mathrm{pcc}} E(v)\right], & z(v) = 1, \\ \rho S^+_0(v), & z(v) = 0. \end{cases} \tag{17}$$

Here, w_{pcc} amplifies retained candidates according to $E(v)$, while $\rho \in [0, 1]$ shrinks the others. Max-normalization within task q then produces $\overline{S}^+_{\mathrm{pcc}}$.

3.5 MILEGPO RETURN AND POLICY UPDATE

MileGPO instantiates the Φ^+ of Equation 9 by inserting the PCC scores into the RCS form:

$$\Phi^+(s) = \max_{v \in \mathcal{S}^+} \overline{S}^+_{\mathrm{pcc}}(v)\, \omega^{d(s,v)}. \tag{18}$$

The trap potential remains $\Phi^- = \Phi^-_{\mathrm{R}}$ from Equation 8. Using increments $\delta^\pm_t$, and suppressing the trajectory index on step-level quantities, the MileGPO step return, denoted by r^{M}_t, is

$$r^{\mathrm{M}}_t = c\, \gamma_G^{d(s_{t+1}, g)} + c\lambda \left(\delta^+_t - \delta^-_t\right). \tag{19}$$

The first term is the GraphGPO return, and λ scales the milestone and trap correction. Directly normalizing their sum can let a small correction dominate when GraphGPO returns are tied. We instead normalize the graph and MileGPO returns separately. Let Norm_{q,s_t} normalize transitions with the same task and source state, and let $r^{\mathrm{G}}_t = r^{\mathrm{G}}(s_t, s_{t+1})$:

$$\widehat{A}^{\mathrm{G}}_t = \mathrm{Norm}_{q,s_t}(r^{\mathrm{G}}_t), \quad \widehat{A}^{\mathrm{mix}}_t = \mathrm{Norm}_{q,s_t}(r^{\mathrm{M}}_t). \tag{20}$$

Here, $\widehat{A}^{\mathrm{G}}_t$ is the step advantage from the GraphGPO return alone, while $\widehat{A}^{\mathrm{mix}}_t$ is the step advantage from the shaped MileGPO return. Their difference, $\widehat{A}^{\mathrm{res}}_t = \widehat{A}^{\mathrm{mix}}_t - \widehat{A}^{\mathrm{G}}_t$, isolates the milestone correction. For the episode-level term, let $Z_{i,t} = \sum_\ell r^{\mathrm{tok}}_{i,t,\ell}$ be the episode-derived score assigned to the response at step t of trajectory τ_i, where ℓ indexes response tokens. Norm_q normalizes these scores over step samples within task q. Each token in action a_t then receives:

$$A_t = w_{\mathrm{step}} \left[\widehat{A}^{\mathrm{G}}_t + \eta \widehat{A}^{\mathrm{res}}_t\right] + w_{\mathrm{episode}}\, \mathrm{Norm}_q(Z_{i,t}). \tag{21}$$

Here, w_{step} and w_{episode} weight different level advantages, and η scales the milestone correction.

The actor uses the same clipped token-level objective and KL regularization as the group-based baselines. **MD, RCS, and PCC use only the rollout graph and on-policy outcome rewards**, requiring no external supervision, auxiliary models, or additional environment interaction.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Benchmarks and Evaluation. We evaluate MileGPO on two long-horizon benchmarks, ALFWorld (Shridhar et al., 2021) and WebShop (Yao et al., 2022), following prior agentic RL settings. ALFWorld contains 3,827 tasks across six household activity categories, with 3,553 training, 140 in-distribution (ID), and 134 out-of-distribution (OOD) tasks; we report success on both evaluation splits. WebShop contains over 1.1 million products and 12k shopping instructions, requiring agents to search, navigate, and purchase products satisfying user constraints. We report Task Score and task success rate. GiGPO and GraphGPO serve as the primary step-group and graph-based credit-assignment baselines. Results are averaged over 3 random seeds (123, 456, and 789) during testing.

Implementation Details. We conduct local experiments with Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct (Qwen et al., 2025), using identical rollout, optimizer, environment, and evaluation settings for MileGPO and the reproduced GiGPO and GraphGPO baselines within each model scale. Because the original experimental environments and implementation details may not be perfectly replicated, our reproduced results may differ from those reported in the original studies; nevertheless, our unified setup provides a fair and consistent basis for comparing methods. Following the shared agent protocol (Feng et al., 2025), the policy observes the two most recent interaction steps and generates reasoning within `<think>` tags followed by an action within `<action>` tags; each iteration samples 16 task groups with eight rollouts per group. The original GraphGPO protocol uses the final checkpoint at 160 optimization steps; since GraphGPO and GiGPO had not consistently converged by then, we train all local methods for 300 steps and select the best validation checkpoint. We use maximum horizons of 50 steps on ALFWorld and 30 on WebShop, an actor learning rate of 10^{-6}, a KL coefficient of 0.01, and an invalid-action penalty of 0.1, with training on four NVIDIA H20 GPUs. Additional details are provided in Appendix A.

4.2 MAIN RESULTS

Table 1 reports higher mean evaluation performance for MileGPO than for the locally reproduced GraphGPO and GiGPO baselines across both environments and model scales. With Qwen2.5-1.5B-Instruct, MileGPO achieves 94.60% OOD success on ALFWorld, outperforming GraphGPO and GiGPO by 3.13 and 4.43%, respectively. On WebShop, it improves success by 3.78 and 2.41% and task score by 2.05 and 0.48% over the two baselines. Notably, the 1.5B MileGPO also surpasses both 7B baselines in ALFWorld ID and OOD success and WebShop task score, demonstrating cross-scale effectiveness. With Qwen2.5-7B-Instruct, MileGPO achieves mean ID and OOD success rates of 98.05% and 95.64% on ALFWorld, respectively, and a mean task score of 90.68% on WebShop. Its mean WebShop success rate is 80.66%, compared with 80.01% for reproduced GraphGPO, a modest numerical difference of 0.65%. At the 1.5B scale, MileGPO further achieves the smallest ALFWorld ID–OOD gap of 1.69%, compared with 3.78 for GraphGPO and 1.89 for GiGPO. Together with Figure 2, these results support our motivation that finer-grained intermediate credit helps resolve ambiguities left by final-goal-distance credit.

4.3 ABLATION AND MECHANISM ANALYSIS

Component Ablations. We use cumulative ablation because the three stages are nested: RCS weights the candidates discovered by MD, and PCC further calibrates the candidate scores used in RCS shaping. Table 2 evaluates PCC and RCS through cumulative removal. Each removal degrades performance across both benchmarks, while the relatively small changes on ALFWorld ID contrast with the larger losses on ALFWorld OOD and WebShop, suggesting that the two components contribute more strongly to robustness under distribution shift and downstream task completion than to in-distribution fitting. Removing PCC decreases ALFWorld OOD success by 3.71% while reducing ID success by only 0.39%, widening the ID–OOD gap from 1.69 to 5.01%. It also reduces WebShop task score and success by 2.38 and 1.43%, respectively. Further removing RCS causes an additional 1.24-point decrease in ALFWorld OOD success and the largest additional drop in WebShop success (3.32%). Without PCC and RCS, uniform milestone propagation falls below reproduced GraphGPO on ALFWorld OOD and WebShop success, suggesting that the gains depend on how intermediate

Table 1: Performance (%) on ALFWorld and WebShop. ALFWorld In-Success and Out-Success denote in-distribution (ID) and out-of-distribution (OOD) success rates, respectively. † denotes the results reported in GraphGPO (Cheng et al., 2026); $*$ indicates our reproduced baselines.

Model	Type	Method	ALFWorld		WebShop	
			In-Success	Out-Success	Task Scores	Task Success Rates
Closed	Prompting	GPT-4o†	48.0	-	31.8	23.7
	Prompting	Gemini-2.5-Pro†	60.3	-	42.5	35.9
Qwen2.5-1.5B-Instruct	Prompting	Qwen2.5†	4.1	-	23.1	5.2
	Prompting	ReAct†	12.8	-	40.1	11.3
	Prompting	Reflexion†	21.8	-	55.8	21.9
	RL Training	PPO (with critic)†	54.4±3.1	-	73.8±3.0	51.5±2.9
	RL Training	RLOO†	69.7±2.5	-	73.9±5.6	52.1±6.7
	RL Training	GRPO†	77.86±1.33	-	84.73±0.49	71.35±2.05
	RL Training	GiGPO†	90.88±0.97	-	87.94±0.43	73.83±2.30
	RL Training	GraphGPO†	92.71±1.32	-	89.29±1.48	78.65±3.86
	RL Training	GiGPO*	92.06±0.72	90.17±0.09	89.81±0.93	76.17±1.57
	RL Training	GraphGPO*	95.25±0.60	91.47±0.51	88.24±1.40	74.80±2.36
	RL Training	**MileGPO**	**96.29±0.55**	**94.60±0.33**	**90.29±0.80**	**78.58±1.20**
Qwen2.5-7B-Instruct	Prompting	Qwen2.5†	14.8	-	26.4	7.8
	Prompting	ReAct†	31.2	-	46.2	19.5
	Prompting	Reflexion†	42.7	-	58.1	28.8
	RL Training	PPO (with critic)†	80.4±2.7	-	81.4±3.1	68.7±5.1
	RL Training	RLOO†	75.5±4.6	-	80.3±3.2	65.7±4.0
	RL Training	GRPO†	83.33±2.05	-	84.31±1.27	75.00±2.78
	RL Training	GiGPO†	94.27±1.33	-	86.72±1.44	78.38±1.94
	RL Training	GraphGPO†	95.31±1.10	-	86.94±0.68	80.31±1.33
	RL Training	GiGPO*	94.34±0.64	91.99±1.10	88.04±2.15	78.39±1.99
	RL Training	GraphGPO*	96.09±0.16	93.29±0.49	87.41±0.82	80.01±0.51
	RL Training	**MileGPO**	**98.05±0.16**	**95.64±0.24**	**90.68±0.80**	**80.66±1.20**

Table 2: Cumulative ablation results (%) with Qwen2.5-1.5B-Instruct. Parentheses show standard deviations; Gap is ID minus OOD(%).

Method	ALFWorld			WebShop	
	ID ↑	OOD ↑	Gap ↓	Score ↑	Succ. ↑
MileGPO	**96.29 (0.55)**	**94.60 (0.33)**	**1.69**	**90.29 (0.80)**	**78.58 (1.20)**
w/o PCC	95.90 (0.48)	90.89 (0.46)	5.01	87.91 (1.85)	77.15 (2.22)
w/o PCC, RCS	95.31 (0.00)	89.65 (0.28)	5.66	87.37 (1.46)	73.83 (2.16)

targets are weighted and calibrated. These cumulative ablations support the contribution of PCC to OOD performance and the benefit of RCS weighting based on observed outcome support.

Credit-Calibration Diagnostics. To examine how MileGPO affects the training process, we use WebShop as a representative case because it exhibits a higher rate of final-distance ties than ALFWorld. Figure 4 tracks the evolution of candidate evidence and validation performance during training, as well as how reliability weighting differentiates transitions tied under GraphGPO's final-distance credit. Panel (a) reports validation success throughout optimization, showing the training dynamics of the three methods. Panel (b) compares uniform MD and RCS on fixed GraphGPO rollouts using a shared candidate set, with PCC disabled in both variants. RCS ranks transitions from successful trajectories above their failed-trajectory counterparts in 26.5% of 569 eligible ties, compared with 17.0% for uniform MD. Because the trajectories and candidate set are held fixed, this difference reflects the effect of reliability weighting on credit allocation within the replay. This diagnostic illustrates how reliability weighting converts observed outcome support into intermediate credit distinctions. Panel (c) partitions success-visited candidates before PCC reweighting. The

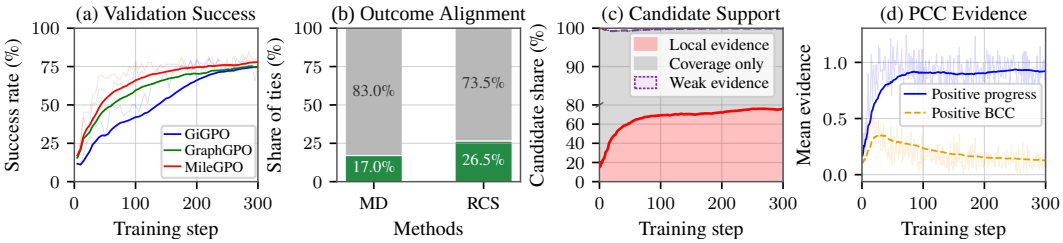


Figure 4: WebShop credit-calibration diagnostics. (a) Validation success compares three locally trained methods. (b) Outcome alignment in offline replay with PCC disabled. Among opposite-outcome transition pairs tied under GraphGPO, colored segments show the fraction assigned higher credit on the successful-trajectory side; gray segments show the remaining pairs. (c) Candidate support partitions success-visited candidates into those with local evidence (positive progress or BCC), successful-path coverage only, and weak candidates. (d) PCC evidence separates positive progress from positive sibling-branch contrast.

increasing local-evidence share suggests that more candidate milestones are supported by transition-level evidence. Panel (d) further separates positive progress from sibling-branch contrast, showing how the two sources of local evidence evolve during training. Paired examples illustrating the credit differences introduced by MD and RCS are provided in Appendix C.2.

4.4 Computational Efficiency

Figure 5 shows MileGPO's computational overhead. For a graph with $|\mathcal{V}|$ nodes, $|\mathcal{E}|$ edges, N transitions, and maximum out-degree Δ, final-distance search costs $O((|\mathcal{V}|+|\mathcal{E}|)\log|\mathcal{V}|)$, candidate statistics cost $O(N)$, and sibling comparisons cost $O(|\mathcal{E}|\Delta)$. The two target potentials add $O(|\mathcal{V}|+|\mathcal{E}|)$ storage, leaving model-scale and rollout complexity unchanged.

Figure 5(a) reports published GraphGPO iteration times, with Agg., Adv., Roll., and Upd. denoting graph aggregation, graph advantage computation, rollout, and policy update, respectively. Graph processing accounts for a small fraction of iteration time compared with rollout and policy updates. Panel (b) reports median single-thread CPU replay times and interquartile ranges over five stored batches on WebShop and ALFWorld, with graph construction shared across methods. Across analyses, MileGPO's post-rollout computation remains lightweight relative to training costs. These results show that MileGPO improves the credit signal without additional environment interactions or model inference, while confining its overhead to graph processing.

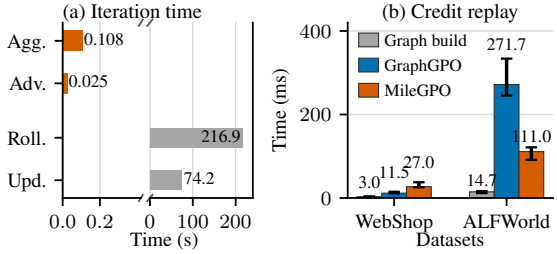


Figure 5: Computational efficiency: (a) iteration time; (b) credit replay time.

5 Conclusion

We study a central limitation of graph-based credit assignment for long-horizon LLM agents: final-goal distance captures reachability but can leave locally competing actions indistinguishable and cannot determine whether a success-visited state represents reliable progress. We introduce **MileGPO**, which extracts intermediate credit from grouped on-policy rollouts using reliability-calibrated milestones, local progress, and shared-anchor branch evidence, without external supervision, auxiliary models, or additional environment interaction. Across ALFWorld and WebShop, MileGPO achieves strong performance at both the 1.5B and 7B scales, with ablations supporting the contributions of RCS and PCC and diagnostics showing how reliability weighting distinguishes transitions tied by final-goal-distance credit. MileGPO adds lightweight post-rollout graph processing, showing that rollout groups provide sufficient local evidence for effective process-level credit.

AI USE STATEMENT

In this work, we used Large Language Models(LLMs) to edit the manuscript for grammar, wording, and readability at the phrase and paragraph levels. Additionally, we used generative AI assistance for LaTeX formatting and compilation checks, including table layout and figure placement. We have reviewed the AI-assisted revisions. We take responsibility for the final content of this work, including its text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We describe the MileGPO method in Section 3 and the experimental setup in Section 4. Appendix A provides the training procedure, prompt templates, hyperparameters, baseline alignment, checkpoint selection, and evaluation settings. Appendix B states the assumptions and provides proofs of the theoretical propositions. We provide the MileGPO implementation, configurations, and training scripts as anonymous supplementary code. The implementation uses the same verl-agent training framework as the locally reproduced GraphGPO and GiGPO baselines. The accompanying README describes the package structure and training commands and links to the upstream documentation for environment setup and dataset downloads.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce-style optimization for learning from human feedback in llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, 2024.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022. URL https://arxiv.org/abs/2204.05862.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. Alphamath almost zero: Process supervision without process. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/30dfe47a3ccbee68cffa0c19ccb1bc00-Abstract-Conference.html.

Xin Cheng, Shuo He, Lang Feng, HaiYang Xu, Ming Yan, Lei Feng, and Bo An. Beyond trajectory-level attribution: Graph-based credit assignment for agentic reinforcement learning. In *Forty-third International Conference on Machine Learning*, 2026.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, Jiarui Yuan, Huayu Chen, Kaiyan Zhang, Xingtai Lv, Shuo Wang, Yuan Yao, Xu Han, Hao Peng, Yu Cheng, Zhiyuan Liu, Maosong Sun, Bowen Zhou, and Ning Ding. Process reinforcement through implicit rewards. *Trans. Mach. Learn. Res.*, 2026, 2026. URL https://openreview.net/forum?id=9SkkifLopZ.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samual Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/5950bf290a1570ea401bf98882128160-Abstract-Datasets_and_Benchmarks.html.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training, 2025. URL https://arxiv.org/abs/2505.10978.

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

Shuo He, Lang Feng, Qi Wei, Xin Cheng, Lei Feng, and Bo An. Hierarchy-of-groups policy optimization for long-horizon agentic tasks, 2026. URL https://arxiv.org/abs/2602.22817.

Zhenyu Hou, Ziniu Hu, Yujiang Li, Rui Lu, Jie Tang, and Yuxiao Dong. Treerl: LLM reinforcement learning with on-policy tree search. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pp. 12355–12369. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.ACL-LONG.604. URL https://doi.org/10.18653/v1/2025.acl-long.604.

Wouter Kool, Herke van Hoof, and Max Welling. Buy 4 REINFORCE samples, get a baseline for free! In *Deep Reinforcement Learning Meets Structured Prediction, ICLR 2019 Workshop, New Orleans, Louisiana, United States, May 6, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=r1lgTGL5DE.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Xufang Luo, Yuge Zhang, Zhiyuan He, Zilong Wang, Siyun Zhao, Dongsheng Li, Luna K. Qiu, and Yuqing Yang. Agent lightning: Train any ai agents with reinforcement learning, 2025. URL https://arxiv.org/abs/2508.03680.

Changdae Oh, Wendi Li, Seongheon Park, Samuel Yeh, Tanwi Mallick, and Sharon Li. Neglected free lunch from post-training: Progress advantage for llm agents, 2026. URL https://arxiv.org/abs/2606.26080.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback, 2022.

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.

Qwen, :, 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, 2025. URL https://arxiv.org/abs/2412.15115.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2024. URL https://arxiv.org/abs/2305.18290.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation, 2018. URL https://arxiv.org/abs/1506.02438.

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.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew J. Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=0IOX0YcCdTn.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. Learning to summarize with human feedback. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1f89885d556929e98d3ef9b86448f951-Abstract.html.

Daoyu Wang, Qingchuan Li, Mingyue Cheng, Jie Ouyang, Shuo Yu, Qi Liu, and Enhong Chen. Steppo: Step-aligned policy optimization for agentic reinforcement learning, 2026a. URL https://arxiv.org/abs/2604.18401.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023. URL https://arxiv.org/abs/2305.16291.

Hanyang Wang, Weijieying Ren, Yuxiang Zhang, Ding Cao, Zhizhao Zeng, Ke Zeng, and Tianxiang Zhao. Bipace: Bisimulation-guided policy optimization with action counterfactual estimation for llm agents, 2026b. URL https://arxiv.org/abs/2606.25556.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In 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), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 9426–9439. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.510. URL https://doi.org/10.18653/v1/2024.acl-long.510.

Yunan Wang, Minghui Song, Zihan Zhang, Shaohan Huang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, and Qi Zhang. Group-graph policy optimization for long-horizon agentic reinforcement learning, 2026c. URL https://arxiv.org/abs/2606.22995.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, Yiping Lu, Kyunghyun Cho, Jiajun Wu, Li Fei-Fei, Lijuan Wang, Yejin Choi, and Manling Li. Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning, 2025. URL https://arxiv.org/abs/2504.20073.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256, 1992. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

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

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/82ad13ec01f9fe44c01cb91814fd7b8c-Abstract-Conference.html.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: an open-source LLM reinforcement learning system at scale. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/a4277440d50f1f15d2cb4c14f7e0c0d2-Abstract-Conference.html.

Yuliang Zhan, Xinyu Tang, Jian Li, Dandan Zheng, Weilong Chai, Jingdong Chen, Jun Zhou, Ge Wu, Wenyue Tang, and Hao Sun. Graphpo: Graph-based policy optimization for reasoning models, 2026. URL https://arxiv.org/abs/2606.18954.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, volume ACL 2025 of *Findings of ACL*, pp. 10495–10516. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.FINDINGS-ACL.547. URL https://doi.org/10.18653/v1/2025.findings-acl.547.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, volume 2024, pp. 15585–15606, 2024a.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. Archer: Training language model agents via hierarchical multi-turn rl, 2024b. URL https://arxiv.org/abs/2402.19446.

Yifei Zhou, Song Jiang, Yuandong Tian, Jason Weston, Sergey Levine, Sainbayar Sukhbaatar, and Xian Li. Sweet-rl: Training multi-turn llm agents on collaborative reasoning tasks, 2025. URL https://arxiv.org/abs/2503.15478.

Table 3: Training and evaluation hyperparameters shared across the local Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct runs on ALFWorld and WebShop. Environment-specific settings are described in the text.

Group	Hyperparameter	ALFWorld / WebShop
Model and data	Policy model	Qwen2.5-1.5B-Instruct / Qwen2.5-7B-Instruct
	Training task groups/iteration	16
	Rollouts per task group (K)	8
	Maximum prompt / response tokens	4096 / 512
	History length	2
Optimization	Actor learning rate	1×10^{-6}
	KL loss coefficient / type	0.01 / low-variance KL
	Invalid-action penalty coefficient	0.1
	Optimization steps / validation interval	300 / 5
Rollout and evaluation	Tensor model parallel size	2
	vLLM GPU memory utilization	0.40
	Validation decoding	temperature 0.4, sampling enabled
	Checkpoint selection	best validation success rate

A Implementation and Reproducibility Details

A.1 Training Procedure

Algorithm 1 summarizes how MileGPO is inserted into the grouped-rollout training pipeline. The rollout and graph-construction stages are shared with GraphGPO; MileGPO changes the computation between graph construction and the policy update. In particular, MD, RCS, and PCC are all computed from the current task-local rollout graph and its observed outcomes. They do not invoke the environment or an auxiliary model.

The separate normalization in line 10 is important when several outgoing transitions are tied under final-goal distance: it prevents a numerically small potential correction from replacing the scale of the graph return. The policy update instead adds the normalized residual between the shaped and graph-only advantages, as defined in the main method section.

A.2 Shared Training and Evaluation Configuration

Local experiments use Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct as the policies. Each training iteration samples 16 task groups with eight rollouts per group. The agent retains the two most recent interaction steps and is limited to 50 environment steps on ALFWorld and 30 on WebShop. We train for 300 optimization steps with an actor learning rate of 10^{-6}, a KL coefficient of 0.01, and an invalid-action penalty of 0.1. Validation uses stochastic decoding at temperature 0.4.

Within each model scale, rollout, optimizer, and environment settings are held fixed across the local comparisons. Method-specific return discounts follow the corresponding baseline and MileGPO recipes. Advantage normalization is a recipe-level setting: ALFWorld uses mean–standard-deviation normalization, whereas WebShop uses mean-only normalization. For each reported method, we run three test evaluations with seeds 123, 456, and 789. WebShop evaluation uses 512 examples per seed from the official-small split; ALFWorld evaluation uses the full out-of-distribution split of 134 tasks.

Table 3 lists settings shared by both environments. The environment-specific settings are 128/256 training-time validation examples, 50/30 maximum environment steps, mean–standard-deviation versus mean-only advantage normalization, 256/64 PPO mini-batch sizes, and 64/4 actor and rollout/reference micro-batch sizes per GPU for ALFWorld/WebShop, respectively.

Algorithm 1 MileGPO training procedure

Require: Policy π_θ, task distribution $p(q)$, group size K, maximum horizon T, graph discount γ_G, shaping coefficient λ, residual correction coefficient η

for each training iteration **do**
  Set the rollout policy to the current policy
  Sample task groups $q \sim p(q)$ and collect K trajectories per task
  Canonicalize observations and construct one directed rollout graph G_q per task
  Compute final-goal distances and GraphGPO returns r^{G}
  Mine candidate milestones and traps from visitation and outcome statistics (MD)
  Weight positive and negative candidates and construct the two target potentials (RCS)
  Compute local-progress and shared-anchor branch evidence; retain, amplify, or shrink milestone candidates (PCC)
  Form the shaped step returns r^{M} from r^{G} and the potential increments
  Normalize r^{G} and r^{M} within each task–source-state group
  Combine the graph, residual milestone, and episode-level advantages
  Update θ with the clipped policy objective and KL regularization
end for

ALFWorld — AGENT INTERACTION 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 finished, choose an admissible action and present it within <action> </action> tags.

Figure 6: ALFWorld prompt template. Runtime placeholders are blue and output tags are green.

A.3 Baseline Alignment and Checkpoint Selection

The locally reproduced GiGPO and GraphGPO baselines share the policy model, task data, prompt construction, rollout group size, interaction horizon, optimizer settings, validation interval, and hardware allocation with MileGPO. Each method retains its return definition, return discounts, and advantage normalization convention. The main table distinguishes results from these local runs, marked as reproduced, from results taken from the corresponding source papers, marked as reported.

All aligned local methods are trained for 300 optimization steps because their validation curves had not converged under the shorter reported schedule. We evaluate the checkpoint with the highest training-time validation success rate. ALFWorld ID and OOD use separate evaluation splits, while WebShop uses the official-small protocol.

A.4 Agent Prompt Templates

The prompts are shared by MileGPO and the locally reproduced baselines. At each interaction step, placeholders are filled with the task, the two most recent observation–action pairs, the current observation, and the admissible actions. The reasoning and executable action are enclosed by `<think>` and `<action>` tags, respectively.

WebShop AGENT INTERACTION 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. You should first reason step-by-step, then consider which admissible action best advances the shopping goal. This reasoning process MUST be enclosed within <think> </think> tags. Once finished, choose an admissible action and present it within <action> </action> tags.

Figure 7: WebShop prompt template. Runtime placeholders are blue and output tags are green.

A.5 Observation Anchors and Branch Aggregation

We build a separate graph for each task using exact matches between observation anchors. An ALFWorld anchor combines the current observation, agent location, inventory and object states, recorded object movements, and sorted admissible commands. A WebShop anchor consists of the formatted current page; the separately supplied list of available actions is not appended. Both anchors exclude the recent observation–action history in the policy prompt, so decisions with different histories can share a node.

Each BCC branch is identified by a source–destination pair (u, v). Full decoded responses, including any reasoning, are stored as transition attributes. All transitions with the same endpoints are pooled, even when their executable actions differ. Branch success and failure rates are computed over distinct trajectories, counting each trajectory once per branch regardless of repeated traversals. BCC compares a branch with the unweighted mean success rate of its sibling branches and assigns zero credit when no sibling exists. The resulting credit reflects empirical outcome differences between endpoint-defined branches under the chosen observation abstraction. When omitted history affects subsequent behavior or task completion, pooling can obscure differences in node-level success rates; similarly, endpoint-based branches can obscure differences between executable actions. These effects can propagate to milestone scores and credit assignment. The usefulness of the pooled statistics therefore depends on whether the anchors preserve distinctions relevant to continuation outcomes. Our experiments evaluate the specified abstraction; sensitivity to history-aware nodes and action-aware branches remains untested.

A.6 MileGPO Hyperparameters

For each environment, the same MileGPO hyperparameters are used for the 1.5B and 7B policies, without scale-specific tuning. For MileGPO, we use $c = 10$, $\omega = 0.20$, $\gamma_\Phi = 1$, and $\lambda = 0.25$, with $\gamma_G = 0.20$ in both environments. The milestone and trap weights are $w_+ = 1$ and $w_- = 0.25$. Candidate mining uses $w_s = w_m = w_f = 1$, $w_c = w_l = 0.25$, a minimum node support of one trajectory, and a trap threshold of 0.1 with at least two failed-trajectory visits unless a revisit is observed. The PCC stage sets $\alpha_d = \alpha_s = \alpha_f = w_{\text{bc}} = w_{\text{pg}} = w_{\text{pcc}} = 1$ in both environments. The policy update uses $w_{\text{step}} = w_{\text{episode}} = 1$. ALFWorld uses $\theta_m = 1.1$, $\rho = 0$, $\eta = 0.20$, and $\kappa_{\text{bc}} = 0$. Since $m(v) \leq 1$, coverage alone cannot retain a candidate under this configuration. Positive progress evidence is therefore required for retention, while BCC still contributes to the score enhancement of retained candidates. The WebShop run reported in the main table uses $\theta_m = 0.5$, $\rho = 0.5$, and $\eta = 1$, together with the full progress-or-branch eligibility rule ($\kappa_{\text{bc}} = 1$) for its more ambiguous shopping-state graph. Section A.7 records the additional configurations explored during development and the rationale for these controls

Table 4: MileGPO hyperparameters. MD denotes milestone discovery, RCS denotes reliability-calibrated shaping, and PCC denotes progress-contrastive calibration. The environment-specific PCC controls are fixed during training and evaluation, as are the policy-update weights. All settings are shared by the 1.5B and 7B policies.

Stage	Parameter	Role	ALFWorld	WebShop
Graph return	c	Return scale	10	10
	γ_G	Graph-distance discount	0.20	0.20
	γ_Φ	Potential discount	1.00	1.00
	λ	Shaping coefficient	0.25	0.25
MD and RCS	w_+, w_-	Positive / negative potential weights	1.00 / 0.25	1.00 / 0.25
	w_s, w_m, w_f	Success, coverage, failure weights	1 / 1 / 1	1 / 1 / 1
	w_c, w_l	Centrality and revisit weights	0.25 / 0.25	0.25 / 0.25
	Minimum node support	Distinct trajectory count	1	1
	Minimum trap visits	Failed-trajectory support	2	2
	Minimum trap score	Trap admission threshold	0.10	0.10
	ω	Milestone/trap propagation discount	0.20	0.20
	`return_mode`	Graph and potential composition	`graph_plus_potential`	
PCC	$\alpha_d, \alpha_s, \alpha_f$	Distance, success, failure terms	1 / 1 / 1	1 / 1 / 1
	$w_{\mathrm{bc}}, w_{\mathrm{pg}}$	BCC and progress evidence	1 / 1	1 / 1
	w_{pcc}	PCC reweighting strength	1	1
	θ_m	Coverage retention threshold	1.1	0.5
	κ_{bc}	Branch-only retention switch	0	1
	ρ	Shrink factor for weak candidates	0	0.5
Policy update	$w_{\mathrm{step}}, w_{\mathrm{episode}}$	Step / episode advantage weights	1 / 1	1 / 1
	η	Residual milestone correction weight	0.20	1.00

Table 5: Configurations explored during development and the settings used in the main experiments.

Environment	Configuration	η	θ_m	ρ	κ_{bc}
WebShop	Hard filtering	1	0.5	0	1
	Hard filtering with reduced residual	0.1	0.5	0	1
	Main-experiment configuration	**1**	**0.5**	**0.5**	**1**
ALFWorld	Hard filtering	1	0.5	0	1
	Hard filtering with reduced residual	0.1	0.5	0	1
	Reduced residual with coverage-only retention disabled	0.1	1.1	0	1
	Main-experiment configuration	**0.2**	**1.1**	**0**	**0**

Table 4 gives the MileGPO configuration used in the local comparisons. Symbols follow the definitions in the method section.

A.7 Configurations Explored During Development

We first developed the configuration on WebShop and used its selected settings, $(\eta, \theta_m, \rho, \kappa_{\mathrm{bc}}) = (1, 0.5, 0.5, 1)$, as the starting point for subsequent adjustments on ALFWorld. Table 5 summarizes two additional configurations explored on WebShop and three on ALFWorld, together with the configurations used in the main experiments. The selected ALFWorld configuration uses $(\eta, \theta_m, \rho, \kappa_{\mathrm{bc}}) = (0.2, 1.1, 0, 0)$.

On WebShop, we initialized the configuration as $(\eta, \theta_m, \rho, \kappa_{\mathrm{bc}}) = (1, 0.5, 0, 1)$ and explored variations in the residual weight η and shrink factor ρ. Reducing η to 0.1 did not yield the best result, while restoring $\eta = 1$ and increasing ρ to 0.5 led to the strongest performance among the explored settings. We therefore use $(1, 0.5, 0.5, 1)$ for the main WebShop experiments.

For ALFWorld, we initialized tuning from the selected WebShop configuration and explored adjustments to ρ, η, θ_m, and branch-only retention. We first set $\rho = 0$, then varied η and increased θ_m to 1.1. The best-performing configuration among those explored was $(\eta, \theta_m, \rho, \kappa_{\mathrm{bc}}) = (0.2, 1.1, 0, 0)$, which disables branch-only retention and is used in the main ALFWorld experiments.

A.8 Configuration Consistency in Component Ablations

The configurations marked as reported in Table 5 are the main-experiment settings and the reference configurations for component ablations: $(\eta, \theta_m, \rho, \kappa_{\mathrm{bc}}) = (1, 0.5, 0.5, 1)$ on WebShop and $(0.2, 1.1, 0, 0)$ on ALFWorld. Each component ablation inherits the corresponding environment's configuration and changes only the removed mechanism. Hyperparameters outside that mechanism are held fixed, including the graph and potential discounts, shaping coefficient, residual coefficient, step and episode weights, and advantage normalization. The policy model, rollout group size, data splits, training budget, and checkpoint-selection and evaluation protocols are also shared with the corresponding full-method run.

Removing PCC bypasses its candidate eligibility and score reweighting, corresponding to $w_{\mathrm{pcc}} = 0$ in the implementation, and uses the initial MD candidate scores for RCS. The PCC-specific controls θ_m, ρ, and κ_{bc} are then inactive. The residual coefficient remains $\eta = 0.2$ on ALFWorld and $\eta = 1$ on WebShop, and the MD and RCS settings are retained. The cumulative removal of PCC and RCS follows the same principle: settings outside the removed components retain their main-experiment values.

B Theoretical Properties of MileGPO

We analyze how MileGPO incorporates intermediate credit into GraphGPO on a fixed sampled rollout graph. The following propositions characterize the interpolation between graph-only and shaped advantages, the ability to distinguish transitions tied under graph-distance credit, and the magnitude of the raw shaping correction. These properties explain how the additional signal enters the policy-update advantage. Its practical contribution to task performance is evaluated through the comparisons and ablations in Section 4. Let the step-level component of the token advantage in Equation 21 be

$$\begin{aligned} B_t &= A_t - w_{\mathrm{episode}} \operatorname{Norm}_q(Z_i) \\ &= w_{\mathrm{step}} \left[\widehat{A}_t^{\mathrm{G}} + \eta \widehat{A}_t^{\mathrm{res}} \right]. \end{aligned} \tag{22}$$

We assume $w_{\mathrm{step}} \geq 0$ and $\eta \in [0, 1]$, as in the reported configuration. All normalized advantages below are computed within the same task–source-state group.

Proposition 1 (Controlled residual interpolation)**.** *Define the GraphGPO and fully shaped step components as $B_t^{\mathrm{G}} = w_{\mathrm{step}} \widehat{A}_t^{\mathrm{G}}$ and $B_t^{\mathrm{M}} = w_{\mathrm{step}} \widehat{A}_t^{\mathrm{mix}}$, respectively. Then MileGPO satisfies*

$$B_t = (1 - \eta) B_t^{\mathrm{G}} + \eta B_t^{\mathrm{M}}, \tag{23}$$

and its deviation from the GraphGPO step component is

$$|B_t - B_t^{\mathrm{G}}| = w_{\mathrm{step}} \eta |\widehat{A}_t^{\mathrm{mix}} - \widehat{A}_t^{\mathrm{G}}|. \tag{24}$$

Consequently, $\eta = 0$ recovers the GraphGPO step component, $\eta = 1$ recovers the normalized shaped step component, and intermediate values provide a convex interpolation between them.

Proof. By definition, $\widehat{A}_t^{\mathrm{res}} = \widehat{A}_t^{\mathrm{mix}} - \widehat{A}_t^{\mathrm{G}}$. Substituting this identity into Equation 22 gives

$$\begin{aligned} B_t &= w_{\mathrm{step}} \big[\widehat{A}_t^{\mathrm{G}} + \eta (\widehat{A}_t^{\mathrm{mix}} - \widehat{A}_t^{\mathrm{G}}) \big] \\ &= (1 - \eta) B_t^{\mathrm{G}} + \eta B_t^{\mathrm{M}}. \end{aligned} \tag{25}$$

Subtracting B_t^{G} and taking absolute values yields Equation 24. Since $\eta \in [0, 1]$, the two coefficients in Equation 23 are nonnegative and sum to one. □

Proposition 2 (Resolution of graph-distance ties)**.** *Consider two observed transitions* $e_1 = (s, a_1, s'_1)$ *and* $e_2 = (s, a_2, s'_2)$ *in the same task–source-state normalization group. If GraphGPO assigns them equal step advantages, then*

$$B(e_1) - B(e_2) = w_{\text{step}}\eta \left[\widehat{A}^{\text{mix}}(e_1) - \widehat{A}^{\text{mix}}(e_2)\right]. \tag{26}$$

Thus, for $w_{\text{step}}\eta > 0$*, any strict ordering produced by the normalized shaped advantage becomes the strict ordering of the MileGPO step-level credit. In particular, equal destination-to-goal distances imply equal GraphGPO returns and hence satisfy the premise.*

Proof. Applying Equation 23 to the two transitions and taking their difference gives the expression below, where $\Delta_{\text{G}} = \widehat{A}^{\text{G}}(e_1) - \widehat{A}^{\text{G}}(e_2)$ and $\Delta_{\text{mix}} = \widehat{A}^{\text{mix}}(e_1) - \widehat{A}^{\text{mix}}(e_2)$:

$$B(e_1) - B(e_2) = (1-\eta)w_{\text{step}}\Delta_{\text{G}} + \eta w_{\text{step}}\Delta_{\text{mix}}. \tag{27}$$

The first difference is zero by the premise, which proves Equation 26. Moreover, the GraphGPO return is $c\gamma_G^{d(s',g)}$; therefore $d(s'_1, g) = d(s'_2, g)$ gives equal raw graph returns. Applying the same normalization to equal values in the same group preserves equality. □

Proposition 3 (Bounded raw shaping correction)**.** *Assume the task-wise normalized candidate scores lie in* $[0, 1]$*,* $\omega, \gamma_\Phi \in (0, 1]$*, and* $c, \lambda, w_+, w_- \geq 0$*. Then the one-sided potential increments obey*

$$0 \leq \delta_t^+ \leq w_+, \qquad 0 \leq \delta_t^- \leq w_-, \tag{28}$$

and the raw MileGPO return differs from the GraphGPO return by at most

$$-c\lambda w_- \leq r_t^{\text{M}} - r_t^{\text{G}} \leq c\lambda w_+. \tag{29}$$

For the reported values $c = 10$*,* $\lambda = 0.25$*,* $w_+ = 1$*, and* $w_- = 0.25$*, the correction lies in* $[-0.625, 2.5]$.

Proof. Each positive or negative potential is the maximum of terms of the form $\overline{S}(v)\omega^{d(s,v)}$. The normalized score, distance decay, and their product all lie in $[0, 1]$, so $0 \leq \Phi^+(s), \Phi^-(s) \leq 1$. From Equations 9 and 10,

$$\begin{aligned} 0 \leq \delta_t^+ &\leq \gamma_\Phi w_+ \Phi^+(s_{t+1}) \leq w_+, \\ 0 \leq \delta_t^- &\leq \gamma_\Phi w_- \Phi^-(s_{t+1}) \leq w_-. \end{aligned} \tag{30}$$

Finally, Equation 19 gives $r_t^{\text{M}} - r_t^{\text{G}} = c\lambda(\delta_t^+ - \delta_t^-)$. Substituting the increment bounds proves Equation 29; substituting the reported hyperparameters gives the stated numerical interval. □

Proposition 1 identifies η as the control over interpolation between graph-only and shaped step advantages. Proposition 2 shows that shaping can introduce distinctions where final-goal distance provides none, while Proposition 3 bounds the added return before normalization. These results characterize how intermediate credit modifies the graph-distance signal on a fixed rollout graph.

The experiments evaluate the practical utility of this design. Table 1 reports higher task success rates than the reproduced GraphGPO and GiGPO baselines across both benchmarks and model scales. The cumulative ablations in Table 2 show lower performance after removing PCC and then RCS, supporting their contributions within the evaluated configurations. Figure 4(b) examines how reliability weighting changes credit allocation: among 569 opposite-outcome transition pairs tied under GraphGPO, RCS assigns higher credit to the transition from the successful trajectory in 26.5% of pairs, compared with 17.0% for uniform MD. This replay uses the same rollout outcomes to construct the weights and evaluate the ordering, with PCC disabled, and therefore describes within-batch outcome alignment. Together, the performance comparisons and ablations support the task-level utility of the intermediate credit design, while the replay illustrates its allocation on observed trajectories. They do not establish that individual candidates are valid subgoals or that the branch score estimates true action values.

B.1 Full WebShop Training Dynamics

Figure 8 shows the training reward and validation success rate throughout optimization for GiGPO, GraphGPO, and MileGPO on WebShop. MileGPO improves more rapidly during the early and middle stages on both metrics, while the curves become closer toward the end of training. Final test results for the validation-selected checkpoints are reported in Section 4.

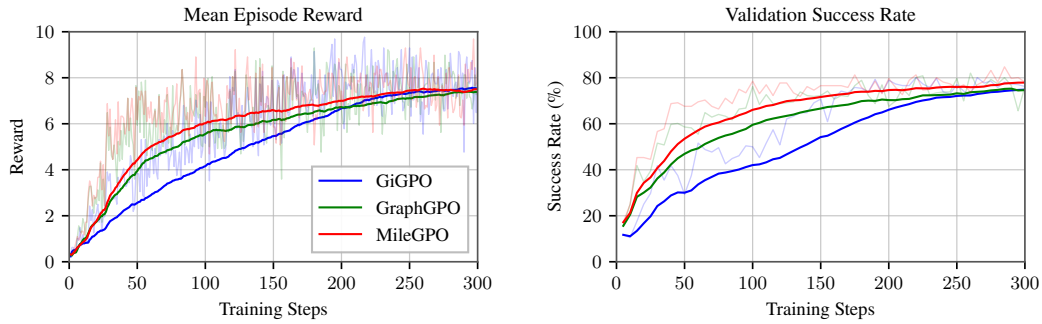


Figure 8: WebShop training dynamics of the aligned GiGPO, GraphGPO, and MileGPO runs. From left to right, the panels show mean episode reward and validation success rate. Faint lines show per-checkpoint measurements, and solid lines show the smoothed training trends.

C Additional Experimental Analyses

C.1 Checkpoint-Wise Replay of Distance Ties

Figure 9 expands the pooled replay result in Figure 4 by resolving it across checkpoints. We identify sibling transitions that share a parent state, have opposite episode outcomes, and are tied under GraphGPO's final-state-distance advantage. On the same GraphGPO trajectories, RCS ranks the successful transition above the failed one for 26.5% of the 569 eligible pairs, compared with 17.0% for uniform MD. Figure 9 shows where these corrections occur across checkpoints and also reveals that the number of eligible ties decreases later in training.

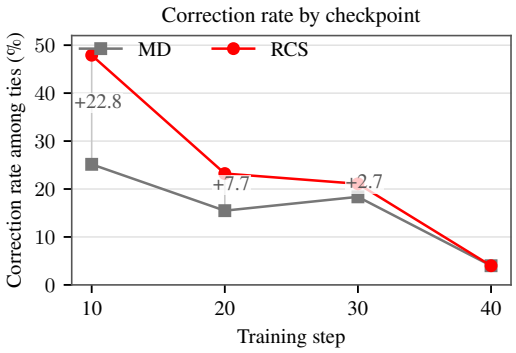


Figure 9: Checkpoint-wise correction rate by checkpoint from offline replay on fixed GraphGPO WebShop rollouts. Step 50 has no eligible ties and is omitted.

We group transitions by task and canonicalized parent state, label a transition as successful when its trajectory-level episode return is positive, and declare a GraphGPO advantage tie at an absolute difference of at most 10^{-6}. Both replay variants use the same RCS-mined candidate set. Uniform MD assigns unit weight to every mined milestone and trap, whereas RCS retains the mined reliability weights; PCC is disabled in both variants to isolate the effect of reliability weighting on the shared candidate set. The replay uses $\gamma_G = 0.20$, $\omega = 0.20$, and $\lambda = 0.25$, matching the reported WebShop recipe. At each checkpoint, the outcome-aligned tie-breaking rate is $N_{\text{aligned}}/N_{\text{tie}}$, where N_{aligned} counts initially tied pairs for which replayed shaping assigns a higher advantage to the transition from the successful trajectory. Because the same rollout outcomes determine the reliability weights and the pair labels, this replay provides a descriptive analysis of outcome-aligned credit allocation.

C.2 Representative Credit-Assignment Corrections

Table 6 illustrates how RCS differentiates two initial transitions tied under GraphGPO credit. The rollouts share the task and initial observation but issue different search queries; both subsequently select the same product priced at $16.99. GraphGPO assigns their initial transitions equal advantages ($\Delta_{\text{G}} = 0.000$), whereas RCS assigns a higher advantage to the transition from the successful trajectory ($\Delta_{\text{RCS}} = +1.533$). The successful rollout completes the task in 28 actions, including

repeated or ineffective interactions, while the other terminates after five actions with a task score of 0.9 and no binary success. MD provides intermediate credit anchors, and RCS uses their outcome support after rollout collection to differentiate earlier transitions under the binary completion objective.

Two additional same-state corrections exhibit the same pattern. On a product page, choosing the listed option `click[xnj-tshirt342-black]` instead of emitting text outside the action grammar changes the RCS margin by $+0.803$. On a compatible loafer page, selecting the requested size with `click[12]` instead of retreating with `click[< prev]` changes it by $+0.562$.

These paired traces illustrate the credit corrections introduced by MD and RCS. The contribution of PCC is examined in Table 2 and Figure 4.

Table 6: Paired WebShop rollouts illustrating the price-constraint correction at training step 30. All 28 transitions of the successful rollout and five transitions of the failed rollout are shown.

Branch	t	**Agent action**	**Environment feedback / observation**	r_t
Successful	0	`search[machine wash men's dress shirts with polyester heathers, heathers cotton, cotton heather, needle sleeve, classic fit with color: kelly green, and fit type: youth, and size: x-small, and price lower than 40.00 dollars]`	Results page 1 (50 products); `B09P39QN2W`, priced at \$16.99, is listed.	0
	1	`click[b09p39qn2w]`	Product page for `B09P39QN2W`; fit-type, color, and size options are shown.	0
	2	`click[kelly green]`	Confirms `kelly green`; product page remains open.	0
	3	`click[x-small]`	Confirms `x-small`; selected options are kelly green and x-small.	0
	4	`click[classic fit]`	Product page unchanged; the requested attribute is not an offered button.	0
	5	`click[needle sleeve]`	Product page unchanged; the requested attribute is not an offered button.	0
	6	`click[classic fit]`	Product page unchanged.	0
	7	`click[youth]`	Confirms `youth`; selected options are youth, kelly green, and x-small.	0
	8	`click[x-small]`	Product page unchanged; x-small remains selected.	0
	9	`click[polyester]; click[heathers]; click[cotton]; click[needle sleeve]; click[classic fit]; click[men]; click[price lower than 40.00 dollars]`	Product page unchanged; the three valid selections remain active.	0
	10	`click[x-small]`	Product page unchanged.	0
	11	`click[needle sleeve]`	Product page unchanged.	0
	12	`click[needle sleeve]`	Product page unchanged.	0

Continued on next page

Table 6: Complete paired WebShop rollout (continued).

Branch	t	Agent action	Environment feedback / observation	r_t
	13	`click[classic fit]; click[needle sleeve]`	Product page unchanged.	0
	14	`click[kelly green]; click[40.00]`	Product page unchanged.	0
	15	`click[polyester heathers]; click[heathers cotton]; click[cotton heathers]; click[lower than 40.00]`	Product page unchanged.	0
	16	`click[polyester heathers]`	Product page unchanged.	0
	17	`click[x-small]`	Product page unchanged.	0
	18	`click[classic fit]; click[price lower than 40.00]`	Product page unchanged.	0
	19	`click[x-small]`	Product page unchanged.	0
	20	`click[classic fit]`	Product page unchanged.	0
	21	`click[x-small]`	Product page unchanged.	0
	22	`click[kelly green]`	Product page unchanged.	0
	23	`click[x-small]`	Product page unchanged.	0
	24	`click[classic fit]`	Product page unchanged.	0
	25	`click[kelly green]`	Product page unchanged.	0
	26	`click[youth]`	Product page unchanged.	0
	27	`click[buy now]`	`success`; purchase accepted.	10
Failed	0	`search[kelly green, machine wash, polyester heathers, heathers cotton, cotton heathers, needle sleeve, classic fit, youth, x-small, price: $40.00--$50.00]`	Results page 1 (50 products); `B09P39QN2W`, priced at $16.99, is listed.	0
	1	`click[b09p39qn2w]`	Product page for `B09P39QN2W`; fit-type, color, and size options are shown.	0
	2	`click[kelly green]`	Confirms `kelly green`; product page remains open.	0
	3	`click[x-small]`	Confirms `x-small`; selected options are kelly green and x-small.	0
	4	`click[buy now]`	Purchase response reports task score 0.9 but no binary success; episode terminates as failure.	0