GRAPHHCA: CLOSED-FORM HINDSIGHT CREDIT ASSIGNMENT FOR LONG-HORIZON LLM AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Group-based reinforcement learning (RL) has advanced large language models (LLMs) and is increasingly extending to agentic tasks, where sparse terminal rewards make step-level credit assignment essential. Existing methods assign credit from what follows an action in sampled rollouts, but do not explicitly capture its retrospective relation to the realized outcome. Hindsight credit assignment (HCA) instead attributes credit through the ratio of hindsight to behavior-policy probabilities, but estimating the hindsight distribution requires an auxiliary model or an extra pass. To address this estimation bottleneck, we propose **GraphHCA**, a model-free realization of HCA that eliminates explicit hindsight-distribution estimation. For terminal-goal tasks with deterministic transitions, Bayes' rule reduces the hindsight ratio to a ratio of behavior-policy success probabilities at consecutive states. Taking logs yields a state-wise *success potential*, whose increment across a transition provides step-level credit. GraphHCA estimates this potential from pooled rollouts through a discounted recursion on the induced transition graph, which admits a unique fixed point on any directed graph. The resulting step-level signal is combined with the trajectory-level advantage, requiring neither a learned hindsight model nor an extra forward pass and recovering GRPO when the step-level weight is zero. Among all compared baselines, GraphHCA achieves state-of-the-art results on ALFWorld and WebShop at both LLM scales, and on Sokoban with a vision-language agent. For example, on ALFWorld it improves overall success rate by up to 24.6 points over GRPO and by up to 4.7 points over the strongest step-level baseline.

1 INTRODUCTION

Large language models (LLMs) are increasingly deployed as controllers of interactive agents (Achiam et al., 2023; Team, 2023; Guo et al., 2025), performing embodied household tasks (Shridhar et al., 2021), navigating web and mobile environments (Yao et al., 2022; Rawles et al., 2023), and invoking external tools for complex reasoning (Yao et al., 2023; Shinn et al., 2023; Schick et al., 2023). These tasks typically require long-horizon interaction and sequential decision making over large natural-language action spaces.

A fundamental challenge in applying reinforcement learning (RL) (Sutton et al., 1998; 1999) to such agents is the sparsity of outcome-based rewards. Many agentic tasks provide only a scalar reward at the end of an episode, leaving intermediate actions without direct supervision. This gives rise to the *credit assignment* problem: determining how a terminal outcome should be attributed to the individual decisions that produced it. The problem becomes increasingly difficult as interaction horizons and action spaces grow.

Conventional actor–critic methods address credit assignment through learned value estimates (Schulman et al., 2017), but training a critic at LLM scale is computationally expensive and particularly challenging under sparse rewards. Group-based methods instead sample a group of rollouts for each task, as illustrated in Fig. 1(a), and estimate advantages directly from their outcomes without an explicit critic. GRPO (Shao et al., 2024), for example, assigns the same trajectory-level advantage to all actions within a rollout, so intermediate decisions are not distinguished by their individual contributions to the final outcome. This coarse attribution has motivated step-level variants: GiGPO (Feng et al., 2025) compares transitions from shared states using the outcomes of

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

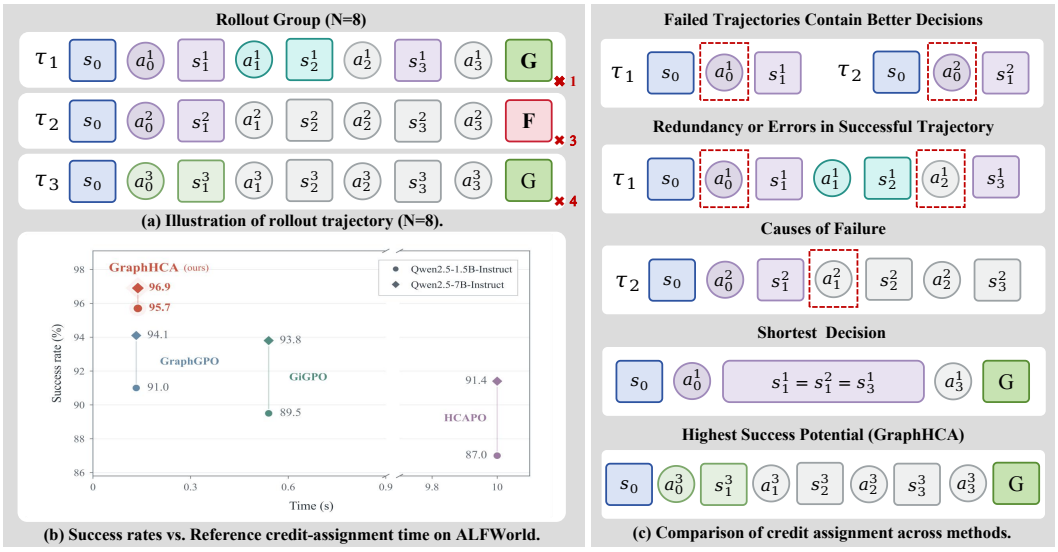


Figure 1: **Comparison of credit-assignment principles.** (a) Illustration of rollout trajectory (N=8). Squares and circles denote states and actions; matching non-gray state colors indicate identical states. (b) ALFWorld success rates versus reference credit-assignment time. (c) Comparison of credit assignment across methods, highlighting the limitations of relying solely on trajectory outcomes or shortest paths.

their respective rollouts, whereas GraphGPO (Cheng et al., 2026) scores transitions by shortest-path progress in the pooled rollout graph. Although these methods provide finer-grained signals, their credit is still derived primarily from forward evidence contained in sampled continuations, without explicitly modeling the retrospective relation between an intermediate action and the realized outcome. As illustrated in Fig. 1(c), a failed rollout may still contain useful intermediate decisions, while a successful rollout may include redundant or erroneous ones; likewise, shortest-path progress need not identify the continuation with the highest empirical expected success. HCA (Harutyunyan et al., 2019) attributes credit retrospectively through the hindsight-to-policy ratio, but estimating the hindsight distribution is expensive: it requires either an auxiliary model or, as in HCAPO (Tan et al., 2026), an extra outcome-conditioned policy pass. Fig. 1(b) shows that the additional model pass of HCAPO raises its credit-assignment time to about 10 seconds, the highest among all methods. We therefore face a dilemma: step-level credit is either cheap but only forward-looking (Fig. 1(c)), or retrospective but too expensive to obtain (Fig. 1(b)).

Our key observation is that, in terminal-goal tasks with deterministic transitions, the hindsight ratio admits a closed-form expression, so it need not be estimated by an auxiliary model or an extra forward pass. By Bayes' rule, it reduces to the ratio of behavior-policy success probabilities at the successor and current states, eliminating the need to model the hindsight distribution explicitly and thus avoiding the outcome-conditioned pass that makes HCAPO's credit-assignment time the highest in Fig. 1(b). We define the logarithm of this state-wise success probability as the *success potential*, whose change across a transition corresponds to the log hindsight ratio and serves as a dense step-level credit signal. Hindsight credit assignment is thus reduced from estimating a distribution over the combinatorial action space to estimating a single scalar per state.

Having reduced hindsight credit to estimating a single state-wise scalar, the remaining challenge is to estimate that scalar from a finite group of sampled rollouts. We therefore propose **GraphHCA**, a model-free realization of HCA that estimates the success potential directly from sampled rollouts, as illustrated in Fig. 2. Because each rollout observes only one sampled continuation at a visited state, GraphHCA pools a group of rollouts into a shared transition graph, merges identical states, and propagates terminal outcomes according to empirical action frequencies. To recover the path-length information lost through state merging, we introduce a propagation discount, which also makes the resulting expectation backup contractive and guarantees a unique fixed point even on cyclic graphs. The resulting potential differences provide transition-level credit, which is standardized among actions leaving the same state and combined with the trajectory-level advantage for policy optimization. GraphHCA thus extracts dense hindsight credit from rollouts already collected for training,

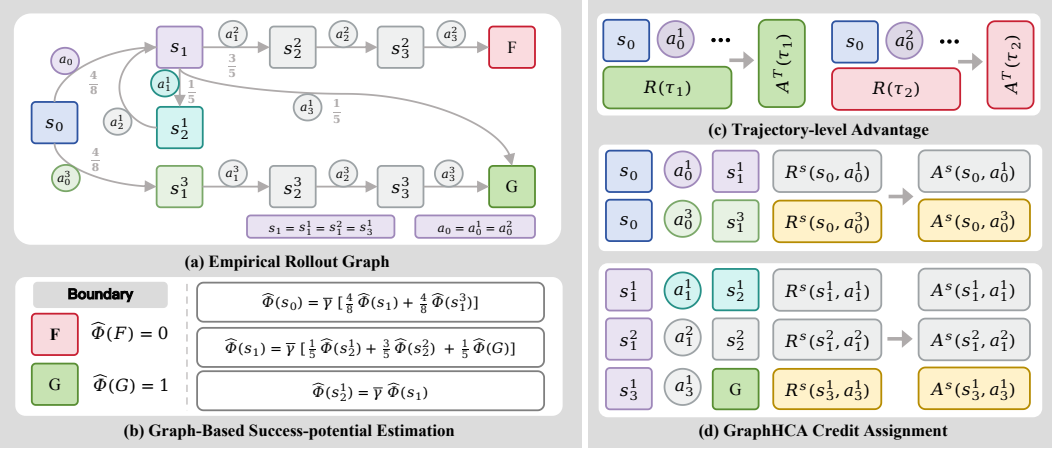


Figure 2: **Overview of GraphHCA. (a)** Rollouts are merged into a graph where identical states share a node and edges carry action frequencies. **(b)** The success probability $\hat{\Phi}$ is the fixed point of a discounted recursion with $\hat{\Phi}(F) = 0$ and $\hat{\Phi}(G) = 1$. **(c)** The trajectory-level advantage $A^T(\tau)$ is shared by all steps of τ. **(d)** GraphHCA takes the increment of $\log \hat{\Phi}$ as the step reward R^s and standardizes it among transitions sharing a source state to obtain A^s.

without learning an auxiliary hindsight model or requiring an additional outcome-conditioned forward pass, and reduces to GRPO when the step-level term is disabled. Our contributions are summarized as follows:

- **Closed-form hindsight credit.** We show that, for terminal-goal tasks with deterministic transitions, HCA reduces to state-wise success probabilities, replacing hindsight-distribution estimation with a single scalar per state.
- **Model-free graph estimation.** We estimate this scalar directly from pooled rollouts through a discounted expectation recursion on the transition graph, with a unique fixed point on arbitrary directed graphs, and requires no learned critic, auxiliary hindsight model, or extra forward pass.
- **Consistent empirical gains.** Across ALFWorld (Shridhar et al., 2021), WebShop (Yao et al., 2022), and 6×6 Sokoban (Schrader, 2018), GraphHCA achieves the best performance among all compared baselines across both LLM scales and the vision-language setting. For example, GraphHCA improves success by up to 24.6 points over GRPO and by up to 4.7 points over the strongest step-level baseline on ALFWorld.

2 Related Work

RL for LLM agents. Reinforcement learning has become a central paradigm for LLM post-training, progressing from RL from human feedback (Ziegler et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022; Bai et al., 2022; Rafailov et al., 2023) to RL with verifiable rewards (Guo et al., 2025; Kimi Team, 2025; Shao et al., 2024), with automatically checkable feedback supporting reasoning, code, search, and tool use (Le et al., 2022; Jin et al., 2025; Qian et al., 2025). For long-horizon LLM agents in interactive environments (Yao et al., 2023; Shinn et al., 2023; Schick et al., 2023; Shridhar et al., 2021; Yao et al., 2022; Rawles et al., 2023; Yang et al., 2024b; Trivedi et al., 2024), early approaches often relied on learned critics, including PPO-style and agent-specific hierarchical or search-augmented methods (Schulman et al., 2017; Peng et al., 2019; Zhou et al., 2024; Putta et al., 2024; Bai et al., 2024). At LLM scale, however, critic training is costly and unstable under sparse terminal rewards, motivating critic-free advantage estimation from sampled rollouts.

Step-level credit assignment. Group-based methods replace learned critics with rollout statistics, but GRPO (Shao et al., 2024) broadcasts a single trajectory-level advantage across steps. Efforts to refine credit follow two main directions. Dense shaping adds stepwise signals: process reward models (Lightman et al., 2024) require extra supervision, intrinsic rewards (Wang et al., 2026) rely

on surrogate objectives, and potential-based shaping (Ng et al., 1999) specifies a policy-invariant reward form but leaves the potential unspecified. Rollout-based methods use forward evidence from sampled continuations: GiGPO (Feng et al., 2025) compares transitions from shared states, while GraphGPO (Cheng et al., 2026) scores shortest-path progress on pooled rollout graphs. Hindsight credit assignment (HCA) (Harutyunyan et al., 2019) instead formalizes outcome-conditioned credit through the hindsight-to-policy ratio, but estimating the hindsight distribution typically requires an auxiliary model or an extra outcome-conditioned policy pass (Tan et al., 2026). GraphHCA removes this bottleneck by deriving the ratio in closed form for deterministic terminal-goal tasks and estimating the resulting state-wise success potential directly from pooled rollouts.

3 PRELIMINARIES

RL for long-horizon LLM agents. We model an interactive agent task as a finite-horizon decision process. Let x denote the task instruction and s_t the environment state at step t. At each step, the LLM policy π_θ samples a natural-language action $a_t \sim \pi_\theta(\cdot \mid s_t, x)$, after which the environment transitions to s_{t+1}. A rollout yields $\tau = (s_0, a_0, \ldots, a_{T-1}, s_T)$ under the behavior policy $q(a \mid s, x) = \pi_{\theta_{\text{old}}}(a \mid s, x)$ and receives the terminal reward $R(\tau) = \mathbf{1}[s_T \in \mathcal{G}]$, where $\mathcal{G}$ and $\mathcal{F}$ denote the goal and failure sets. Since all rollouts within a group share the same task x, we suppress this common conditioning below and write $q(a \mid s)$ and $\pi_\theta(a \mid s)$ for brevity. Maximizing $J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)]$ gives the policy gradient $\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T-1} A_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\right]$, which requires a per-step advantage A_t, whereas the terminal reward reveals only whether the task succeeded, not which intermediate decisions should receive credit for the outcome.

Group-based advantage estimation. Classically, A_t is estimated by a learned critic (Schulman et al., 2017), which is costly and can be unstable at LLM scale. Group-based methods instead derive advantages directly from rollout statistics. GRPO (Shao et al., 2024) samples N trajectories per task under q and assigns the standardized terminal return

$$A^{\mathrm{T}}(\tau_i) = \frac{R(\tau_i) - \mu_R}{\sigma_R} \tag{1}$$

to every step of τ_i, leaving intermediate credit dependent on subsequent decisions. For finer credit, GiGPO (Feng et al., 2025) compares transitions from shared states using their trajectory outcomes, while GraphGPO (Cheng et al., 2026) scores transitions by shortest-path progress on the pooled rollout graph. Both still derive credit from forward evidence in sampled futures, without explicitly conditioning the action distribution on the realized outcome.

Hindsight credit assignment. HCA (Harutyunyan et al., 2019) takes a retrospective view, defining step-level credit through the relation between an action and a realized future outcome. Its *hindsight distribution* $h(a_t \mid s_t, s_k)$ gives the probability of taking a_t at s_t conditioned on later reaching s_k. The ratio to the behavior policy,

$$\rho(s_t, a_t, s_k) = \frac{h(a_t \mid s_t, s_k)}{q(a_t \mid s_t)}, \tag{2}$$

upweights actions whose probability increases under outcome conditioning and downweights those whose probability decreases. Its practical difficulty lies in estimating h, which classical HCA learns with an auxiliary model over the action space, while HCAPO (Tan et al., 2026) approximates it through an additional outcome-conditioned pass of the policy LLM.

4 METHOD

We propose **GraphHCA**, as illustrated in Fig. 2, a model-free realization of hindsight credit assignment (HCA) for long-horizon LLM agents. Our key observation is that, for terminal-goal task with deterministic transitions, the hindsight ratio admits a closed form in terms of state-wise success probabilities. We define the logarithm of this success probability as the *success potential*, estimate it model-free on a pooled rollout graph, and use its transition differences as step-level credit for policy optimization. The following subsections derive the potential (Sec. 4.1), estimate it on the rollout graph (Sec. 4.2), integrate it into policy optimization (Sec. 4.3), and analyze its properties (Sec. 4.4). The detailed algorithm is presented in Appendix A.

4.1 The Success Potential from Bayesian Hindsight

Hindsight as success-posterior inference. We specialize HCA to terminal-goal tasks with binary outcomes. Let $\mathcal{G}$ and $\mathcal{F}$ denote the terminal goal and failure sets, and let $G = \{s_T \in \mathcal{G}\}$ denote the success event. Rather than conditioning on an arbitrary future state s_k, we condition on G, so that $h(a_t \mid s_t, s_k)$ becomes the success-conditioned posterior $h(a \mid s, G)$. By Bayes' rule,

$$h(a \mid s, G) = \frac{q(a \mid s)\, P(G \mid s, a)}{P(G \mid s)}, \qquad P(G \mid s) = \sum_a q(a \mid s)\, P(G \mid s, a). \tag{3}$$

Here, $P(G \mid s)$ is the marginal probability of eventual success from s. For the exact analysis, we assume that the conditional probability of eventual success under q depends only on the current environment state, independently of the preceding history and time step. Under deterministic transitions, each action a uniquely determines a successor $s'_a = f(s, a)$; together with the preceding assumption, this gives $P(G \mid s, a) = P(G \mid s'_a)$. The action-dependent likelihood in the hindsight posterior is therefore characterized by the success probability of its successor state.

The success potential. Define the state-wise success probability

$$\Phi(s) = P(G \mid s), \qquad \Phi(s) = 1 \text{ for } s \in \mathcal{G}, \quad \Phi(s) = 0 \text{ for } s \in \mathcal{F}, \tag{4}$$

and its log-scale *success potential*

$$\Psi(s) = \log \max\{\Phi(s), \epsilon\}, \qquad \epsilon \in (0, 1). \tag{5}$$

The floor only keeps the logarithmic representation finite when $\Phi(s) = 0$ and does not alter the underlying success probability. Under the terminal reward $R(\tau) = \mathbf{1}[s_T \in \mathcal{G}]$, Φ is also the value function of the behavior policy q.

Proposition 4.1 (Closed-form hindsight ratio)**.** *Under the preceding assumption and deterministic transitions, the success probability at any non-terminal state satisfies*

$$\Phi(s) = \sum_a q(a \mid s)\Phi(s'_a) = \mathbb{E}_{a\sim q}[\Phi(s'_a)]\,. \tag{6}$$

For any state with $\Phi(s) > 0$ and action with $q(a \mid s) > 0$, the hindsight ratio specialized to the success event admits the closed form

$$\rho(s, a) = \frac{h(a \mid s, G)}{q(a \mid s)} = \frac{P(G \mid s, a)}{P(G \mid s)} = \frac{\Phi(s'_a)}{\Phi(s)}. \tag{7}$$

The closed form in Eq. (7) avoids explicit estimation of the hindsight distribution. Using the floored potential Ψ in Eq. (5), we define the step credit $\ell(s, a)$ as

$$\ell(s, a) = \Psi(s'_a) - \Psi(s). \tag{8}$$

This credit remains finite when the underlying potential vanishes and equals $\log \rho(s, a)$ whenever the floor is inactive at both s and s'_a. Thus, hindsight credit assignment reduces to estimating the scalar state potential Φ. When the floor is inactive, the resulting additive credit admits the log-likelihood-ratio interpretation analyzed in Sec. 4.4.

4.2 Model-Free Estimation on the Rollout Graph

Pooled rollout graph. The marginalization Eq. (6) characterizes the exact success probability Φ, but sampled rollouts reveal only a subset of the available transitions. Following Cheng et al. (2026), we pool the rollout group into a shared graph, merging identical environment states across rollouts and time steps. Each realized transition $s \xrightarrow{a} s'_a$ is assigned the empirical weight $\hat{q}(a \mid s) = n(s, a)/n(s)$, where $n(s)$ counts visits to s and $n(s, a)$ those on which action a is taken. Pooling enables evidence sharing across trajectories, but merging repeated occurrences of the same state removes their temporal positions and hence their path-length information. Consequently, an undiscounted backup may assign no cost to additional transitions or detours. We therefore define the graph value through the discounted expectation recursion

$$\hat{\Phi}(s) = \bar{\gamma}\, \mathbb{E}_{a\sim\hat{q}}\big[\hat{\Phi}(s'_a)\big], \qquad \bar{\gamma} \in (0, 1), \tag{9}$$

with boundary values $\hat{\Phi} = 1$ on $\mathcal{G}$ and $\hat{\Phi} = 0$ on $\mathcal{F}$. Unlike the exact recursion Eq. (6), $\bar{\gamma}$ is not inherited from the task objective, but acts as a propagation discount that restores sensitivity to path length after state merging. The resulting $\hat{\Phi}$ is a discounted empirical surrogate for Φ, aggregating visits with potentially different interaction histories and remaining horizons.

Proposition 4.2 (Contraction and convergence)**.** *For any directed graph, including graphs with cycles, the operator in Eq. (9) is a $\bar{\gamma}$-contraction in the sup norm. Hence, iteration from any initialization converges geometrically to a unique fixed point $\hat{\Phi}$, corresponding to the $\bar{\gamma}$-discounted expected terminal success under the empirical policy $\hat{q}$.*

Using the fixed point $\hat{\Phi}$, we define the graph success potential $\hat{\Psi}(s) = \log \max\{\hat{\Phi}(s), \epsilon\}$ and the corresponding raw transition credit

$$\hat{\ell}(s, a) = \hat{\Psi}(s'_a) - \hat{\Psi}(s) \in [-\log(1/\epsilon),\, \log(1/\epsilon)]. \tag{10}$$

4.3 Policy Optimization

Step-level advantage. We take the graph credit of Sec. 4.2 as the reward of a realized transition, $R^{\mathrm{s}}(s_t, a_t) = \hat{\ell}(s_t, a_t)$. Following the state-level grouping of (Feng et al., 2025), we standardize it within the group $\mathcal{B}_s$ of all transitions leaving the same state s across the N rollouts,

$$A^{\mathrm{s}}(s_t, a_t) = \frac{R^{\mathrm{s}}(s_t, a_t) - \mu_{\mathcal{B}_{s_t}}}{\sigma_{\mathcal{B}_{s_t}}}, \qquad \mu_{\mathcal{B}_s} = \mathrm{mean}_{\mathcal{B}_s}(R^{\mathrm{s}}), \quad \sigma_{\mathcal{B}_s} = \mathrm{std}_{\mathcal{B}_s}(R^{\mathrm{s}}). \tag{11}$$

We set $A^{\mathrm{s}} = 0$ at singleton states ($|\mathcal{B}_s| = 1$). For both trajectory- and state-level normalization, we also assign zero advantages to groups with zero variance. For groups with positive variance, standardization preserves the ordering induced by $\hat{\ell}$ and therefore ranks the sampled actions from a common state by their successor potential $\hat{\Psi}(s'_a)$.

Combined objective. We add the step term to the trajectory-level advantage $A^{\mathrm{T}}(\tau)$ of Eq. (1),

$$\hat{A}_t = A^{\mathrm{T}}(\tau) + w_{\mathrm{s}}\, A^{\mathrm{s}}(s_t, a_t), \qquad w_{\mathrm{s}} \geq 0, \tag{12}$$

the two being complementary, since A^{T} anchors the update to task success and stays defined at singleton states while A^{s} densifies credit within a trajectory. Using $\hat{A}_t$ for the policy update at step t, we optimize the clipped surrogate with a reference-KL penalty,

$$\mathcal{J}(\theta) = \mathbb{E}\left[\frac{1}{\sum_i T_i}\sum_{i=1}^{N}\sum_{t}\min\Big(\varrho_{i,t}(\theta)\,\hat{A}_{i,t},\ \mathrm{clip}\left(\varrho_{i,t}(\theta), 1-\varepsilon, 1+\varepsilon\right)\hat{A}_{i,t}\Big)\right] - \beta_{\mathrm{KL}}\, D_{\mathrm{KL}}\big(\pi_\theta \,\|\, \pi_{\mathrm{ref}}\big), \tag{13}$$

with $\varrho_{i,t}(\theta) = \pi_\theta(a_{i,t} \mid s_{i,t}) / \pi_{\theta_{\mathrm{old}}}(a_{i,t} \mid s_{i,t})$. Setting $w_{\mathrm{s}} = 0$ recovers GRPO exactly, so GraphHCA is a strict, model-free refinement of it, adding no model of its own since $\hat{\Psi}$ is read directly off the pooled graph, whose $\hat{\Phi}$ draws on the whole group including sub-paths of failed trajectories that never reach the goal on their own.

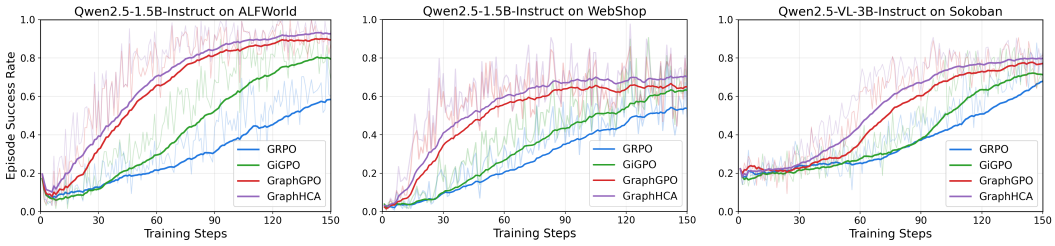
4.4 Properties of the GraphHCA Credit

We further establish that the exact credit is the log-likelihood ratio to the hindsight posterior when the floor is inactive (Proposition 4.3), and that the graph credit separates from GraphGPO's distance surrogate by aggregation rather than by reachability (Proposition 4.4). Proofs are in Appendix B.

Proposition 4.3 (Hindsight KL identity and bounded step credit)**.** *Under the assumptions of Proposition 4.1, fix a non-terminal state s such that $\Phi(s) \geq \epsilon$ and $\Phi(s'_a) \geq \epsilon$ for every action with $q(a \mid s) > 0$. Then, on the support of $q(\cdot \mid s)$, Proposition 4.1 gives $\ell(s, a) = \log[h(a \mid s, G)/q(a \mid s)]$. Therefore,*

$$\mathbb{E}_{a\sim q(\cdot|s)}\big[\ell(s, a)\big] = -\mathrm{KL}\big(q(\cdot \mid s) \,\big\|\, h(\cdot \mid s, G)\big) \leq 0, \tag{14}$$

with equality if and only if $q(\cdot \mid s) = h(\cdot \mid s, G)$. Independently, for the empirical graph, $\hat{\Phi}(s) \in [0, \bar{\gamma}]$ at every interior node. Whether or not the floor is active, it gives $\hat{\Psi}(s) \in [\log \epsilon, 0]$ and $|\hat{\ell}(s, a)| \leq \log(1/\epsilon)$.



(a) ALFWorld (b) WebShop (c) Sokoban

Figure 3: Training success rate over update steps on ALFWorld and WebShop with Qwen2.5-1.5B-Instruct, and on Sokoban with Qwen2.5-VL-3B-Instruct. GraphHCA converges faster and reaches higher final success than the fine-grained credit-assignment baselines across all three environments.

Proposition 4.4 (Expectation versus best-case aggregation)**.** *Let $R^{\mathrm{G}}(s, a, s'_a)$ denote GraphGPO's step reward, and $R^{\mathrm{G}}(s, a, s'_a) \propto \bar{\gamma}^{\,d(s'_a)+c(s,a)}$. $d(\cdot)$ the shortest-path distance to $\mathcal{G}$ on the empirical graph, and adopt the unit-cost setting of (Cheng et al., 2026). Replacing the expectation in Eq. (9) by a maximum under the same boundary defines the fixed point $\Phi^{\max}$, and*

$$\hat{\Phi}(s) \;\le\; \Phi^{\max}(s) \;=\; \bar{\gamma}^{\,d(s)} \qquad (s \text{ goal-reachable}), \tag{15}$$

with equality if and only if $\hat{q}$ is supported on the argmax successors at s and downstream. The two credits are therefore one backup under two aggregations, $\hat{\ell}$ ranking the actions at s by $\hat{\Psi}(s'_a)$ and R^{G} by $d(s'_a)$.

Under the floor-inactive conditions of Proposition 4.3, the exact step credit has a log-likelihood-ratio interpretation relative to the hindsight posterior. Proposition 4.4 further distinguishes GraphHCA from GraphGPO by their aggregation rules: GraphHCA evaluates successors through empirical expected continuation values, whereas GraphGPO relies on shortest-path distance and thus reflects a best-case continuation. Consequently, GraphHCA remains sensitive to transition frequencies and can distinguish equidistant successors whose distance-based scores coincide. HCAPO (Tan et al., 2026) targets the same hindsight ratio through an additional outcome-conditioned policy evaluation, while GraphHCA obtains its credit directly from the pooled rollout graph without an auxiliary hindsight model.

5 Experiments

5.1 Experimental Setup

Benchmarks. We evaluate GraphHCA on three challenging multi-turn agentic benchmarks: ALFWorld (Shridhar et al., 2021) and WebShop (Yao et al., 2022) for LLM agents, and the 6×6 Sokoban game (Schrader, 2018) for a vision-language model (VLM) agent. Following prior work (Feng et al., 2025; Cheng et al., 2026), we report the success rate on ALFWorld, the task score and success rate on WebShop, and the success rate on Sokoban, averaged over three random seeds for all RL methods. Detailed descriptions of the three benchmarks are provided in Appendix C.

Implementation Details. Following Cheng et al. (2026), we use Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct (Yang et al., 2024a) as the base LLMs and Qwen2.5-VL-3B-Instruct (Bai et al., 2025) as the base VLM. The agent keeps only the two most recent interaction steps as memory and emits its reasoning in `<think>` tags before the action in `<action>` tags (Wei et al., 2022). All methods share the same training hyperparameters: group size $N = 8$, learning rate 1×10^{-6}, and 150 update steps. GraphHCA adds three quantities of its own: the propagation discount $\bar{\gamma} = 0.95$ of Eq. (9), failure floor $\epsilon = 1 \times 10^{-2}$ of Eq. (10), and dense weight $w_{\mathrm{s}} = 1$ of Eq. (12). Appendix D reports the more configurations.

5.2 Main Results

LLM Agents on ALFWorld and WebShop. As shown in Table 1, GraphHCA achieves the best overall ALFWorld success rate and the best result on both WebShop metrics at both model scales.

Table 1: Test performance on ALFWorld and WebShop. For ALFWorld, we report the average success rate (%) for each subtask and the overall result. For WebShop, we report the average task score and the average success rate (%). RL results are averaged over three random seeds. The best performance in each column is highlighted in **bold**.

Type	Method	ALFWorld							WebShop	
		Pick	Clean	Cool	Look	Heat	Pick2	All	Score	Succ.
Closed-Source Models										
Prompting	GPT-4o	75.3	60.8	31.2	56.7	21.6	49.8	48.0	31.8	23.7
Prompting	Gemini-2.5-Pro	92.8	63.3	62.1	69.0	26.6	58.7	60.3	42.5	35.9
Qwen2.5-1.5B-Instruct										
Prompting	Qwen2.5	5.9	5.5	3.3	9.7	4.2	0.0	4.1	23.1	5.2
Prompting	ReAct	17.4	20.5	15.7	6.2	7.7	2.0	12.8	40.1	11.3
Prompting	Reflexion	35.3	22.2	21.7	13.6	19.4	3.7	21.8	55.8	21.9
RL Training	PPO	$64.8_{\pm 3.5}$	$40.5_{\pm 6.9}$	$57.1_{\pm 4.9}$	$60.6_{\pm 6.6}$	$46.4_{\pm 4.0}$	$47.4_{\pm 1.9}$	$54.4_{\pm 3.1}$	$73.8_{\pm 3.0}$	$51.5_{\pm 2.9}$
RL Training	RLOO	$88.3_{\pm 3.0}$	$52.8_{\pm 8.6}$	$71.0_{\pm 5.9}$	$62.8_{\pm 8.7}$	$66.4_{\pm 5.5}$	$56.9_{\pm 4.7}$	$69.7_{\pm 2.5}$	$73.9_{\pm 5.6}$	$52.1_{\pm 6.7}$
RL Training	GRPO	$88.2_{\pm 2.9}$	$72.6_{\pm 15.9}$	$67.5_{\pm 7.5}$	$43.3_{\pm 18.3}$	$76.6_{\pm 0.4}$	$51.6_{\pm 9.5}$	$71.1_{\pm 1.6}$	$81.9_{\pm 3.5}$	$64.1_{\pm 2.3}$
RL Training	EMPG	85.5	33.5	78.9	76.2	74.7	69.1	73.7	80.4	60.8
RL Training	GiGPO	$97.6_{\pm 2.4}$	$93.7_{\pm 1.3}$	$\mathbf{89.4}_{\pm 0.6}$	$65.6_{\pm 3.1}$	$95.0_{\pm 5.0}$	$80.6_{\pm 6.9}$	$89.5_{\pm 1.2}$	$85.2_{\pm 0.8}$	$74.2_{\pm 0.8}$
RL Training	HCAPO	$88.6_{\pm 7.0}$	$97.6_{\pm 1.8}$	$84.2_{\pm 0.0}$	$75.0_{\pm 0.0}$	$90.7_{\pm 6.9}$	$74.2_{\pm 6.9}$	$87.0_{\pm 4.1}$	$83.8_{\pm 0.7}$	$68.5_{\pm 1.0}$
RL Training	GraphGPO	$95.6_{\pm 4.4}$	$97.6_{\pm 2.4}$	$83.5_{\pm 3.5}$	$71.3_{\pm 8.8}$	$\mathbf{97.6}_{\pm 2.4}$	$86.6_{\pm 2.3}$	$91.0_{\pm 2.0}$	$88.1_{\pm 0.2}$	$\mathbf{81.3}_{\pm 0.8}$
RL Training	**GraphHCA**	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{100.0}_{\pm 0.0}$	$85.7_{\pm 0.7}$	$\mathbf{89.9}_{\pm 2.4}$	$96.7_{\pm 3.3}$	$\mathbf{95.8}_{\pm 4.2}$	$\mathbf{95.7}_{\pm 1.2}$	$\mathbf{89.3}_{\pm 0.1}$	$\mathbf{81.3}_{\pm 0.8}$
Qwen2.5-7B-Instruct										
Prompting	Qwen2.5	33.4	21.6	19.3	6.9	2.8	3.2	14.8	26.4	7.8
Prompting	ReAct	48.5	35.4	34.3	13.2	18.2	17.6	31.2	46.2	19.5
Prompting	Reflexion	62.0	41.6	44.9	30.9	36.3	23.8	42.7	58.1	28.8
RL Training	PPO	$92.3_{\pm 4.0}$	$64.0_{\pm 8.4}$	$\mathbf{92.5}_{\pm 2.4}$	$89.5_{\pm 7.0}$	$80.3_{\pm 2.0}$	$68.8_{\pm 8.3}$	$80.4_{\pm 2.7}$	$81.4_{\pm 3.1}$	$68.7_{\pm 5.1}$
RL Training	RLOO	$87.6_{\pm 4.3}$	$78.2_{\pm 8.3}$	$87.3_{\pm 5.8}$	$81.3_{\pm 7.6}$	$71.9_{\pm 5.2}$	$48.9_{\pm 8.4}$	$75.5_{\pm 4.6}$	$80.3_{\pm 3.2}$	$65.7_{\pm 4.0}$
RL Training	GRPO	$88.2_{\pm 5.9}$	$71.2_{\pm 9.6}$	$70.0_{\pm 5.0}$	$50.0_{\pm 12.5}$	$64.3_{\pm 2.4}$	$73.7_{\pm 10.5}$	$73.4_{\pm 0.8}$	$82.3_{\pm 2.7}$	$75.4_{\pm 2.7}$
RL Training	EMPG	92.9	75.2	74.8	86.3	73.7	65.3	78.5	81.0	69.3
RL Training	GiGPO	$\mathbf{100.0}_{\pm 0.0}$	$92.3_{\pm 7.7}$	$85.0_{\pm 0.0}$	$81.3_{\pm 6.3}$	$\mathbf{97.6}_{\pm 2.4}$	$94.7_{\pm 0.1}$	$93.8_{\pm 0.8}$	$86.9_{\pm 3.9}$	$82.0_{\pm 3.1}$
RL Training	HCAPO	$99.1_{\pm 1.3}$	$97.3_{\pm 1.9}$	$90.8_{\pm 6.6}$	$90.3_{\pm 2.0}$	$81.8_{\pm 8.8}$	$81.9_{\pm 10.0}$	$91.4_{\pm 2.3}$	$85.1_{\pm 1.3}$	$73.8_{\pm 2.8}$
RL Training	GraphGPO	$98.5_{\pm 1.5}$	$98.1_{\pm 2.7}$	$87.5_{\pm 3.5}$	$68.8_{\pm 8.8}$	$95.2_{\pm 0.0}$	$97.4_{\pm 3.7}$	$94.1_{\pm 0.6}$	$88.6_{\pm 3.9}$	$82.0_{\pm 3.1}$
RL Training	**GraphHCA**	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{100.0}_{\pm 0.0}$	$86.7_{\pm 2.9}$	$\mathbf{95.8}_{\pm 7.2}$	$96.8_{\pm 2.8}$	$\mathbf{98.3}_{\pm 3.0}$	$\mathbf{96.9}_{\pm 0.8}$	$\mathbf{90.5}_{\pm 0.8}$	$\mathbf{82.4}_{\pm 1.2}$

Table 2: Test performance of VLM agents using Qwen2.5-VL-3B-Instruct on the interactive game environment Sokoban. We report the average success rate (%) over three random seeds.

	Qwen2.5-VL	**GRPO**	**GiGPO**	**GraphGPO**	**GraphHCA**
Type	Prompting	RL Training	RL Training	RL Training	**RL Training**
Sokoban [6×6]	11.7	$71.1_{\pm 3.9}$	$79.0_{\pm 3.1}$	$81.6_{\pm 4.03}$	$\mathbf{84.0}_{\pm 1.2}$

It improves over the trajectory-level baseline GRPO by 24.6 and 23.5 points of overall ALFWorld success and by 17.2 and 7.0 points of WebShop success rate for the 1.5B and 7B models respectively. Against the closest baselines GiGPO, HCAPO, and GraphGPO, it still improves overall ALFWorld success by 4.7 to 8.7 points at 1.5B and by 2.8 to 5.5 points at 7B, so the gain comes from the credit rule itself. The largest gains appear on Look and Pick2, the two longest-horizon subtasks, where it leads the best 1.5B baseline by 13.7 and 9.2 points, while the near-saturated Cool and Heat remain comparable. Figures 3a and 3b show the success rate over training steps. GraphHCA achieves the highest success rate over most of training on both ALFWorld and WebShop. It also learns fastest, reaching the final success of GRPO in about a third of the training steps. The success potential provides informative credit when successful rollouts are still rare, which accelerates early learning.

VLM Agent on Sokoban. On Sokoban, where observations are rendered images rather than text, GraphHCA attains a success rate of 84.0%, outperforming GRPO by 12.9 points, GiGPO by 5.0, and GraphGPO by 2.4 (Table 2). It is also the most stable across seeds, with a standard deviation of 1.2 against 3.1 or more for every RL baseline. Figure 3c shows where this gain arises: all four methods stay near a 0.2 success rate for the first twenty updates, GraphHCA is the first to leave that plateau, and the resulting margin holds to the end of training.

Table 3: Ablation over the power-mean order ω of Eq. (16) with Qwen2.5-1.5B-Instruct on ALFWorld: success rate (%) averaged over three random seeds. The per-column best is in **bold**.

ω	Pick	Clean	Cool	Look	Heat	Pick2	All
1 (GraphHCA)	**100.0**$_{\pm 0.0}$	**100.0**$_{\pm 0.0}$	85.7$_{\pm 0.7}$	**89.9**$_{\pm 2.4}$	96.7$_{\pm 3.3}$	**95.8**$_{\pm 4.2}$	**95.7**$_{\pm 1.2}$
5	98.5$_{\pm 2.1}$	**100.0**$_{\pm 0.0}$	**90.0**$_{\pm 7.1}$	87.5$_{\pm 17.7}$	**100.0**$_{\pm 0.0}$	86.8$_{\pm 11.2}$	95.3$_{\pm 4.5}$
10	92.9$_{\pm 10.1}$	95.5$_{\pm 6.4}$	85.9$_{\pm 1.2}$	70.9$_{\pm 5.9}$	85.1$_{\pm 14.3}$	91.2$_{\pm 2.4}$	91.8$_{\pm 5.5}$
max ($\omega \to \infty$)	89.7$_{\pm 2.1}$	98.1$_{\pm 2.7}$	77.5$_{\pm 3.5}$	75.0$_{\pm 0.0}$	95.2$_{\pm 0.0}$	86.9$_{\pm 3.7}$	90.1$_{\pm 1.1}$
GraphGPO	95.6$_{\pm 4.4}$	97.6$_{\pm 2.4}$	83.5$_{\pm 3.5}$	71.3$_{\pm 8.8}$	97.6$_{\pm 2.4}$	86.6$_{\pm 2.3}$	91.0$_{\pm 2.0}$

5.3 Ablation Study

Computational overhead. GraphHCA introduces no additional learned model and operates solely on the pooled rollout graph constructed from the sampled trajectories. Graph construction requires a single pass over the observed transitions, and each sweep of the graph backup in Eq. (9) likewise scales linearly with the graph size, without additional model inference or gradient computation. As shown in Figure 4, rollout and policy update dominate the iteration time at 184.0 s and 25.1 s, respectively, whereas graph construction and step-level advantage computation require only 0.028 s and 0.110 s. Together, the GraphHCA-specific computation accounts for only 0.062% of the measured per-iteration runtime, providing dense step-level credit with negligible overhead.

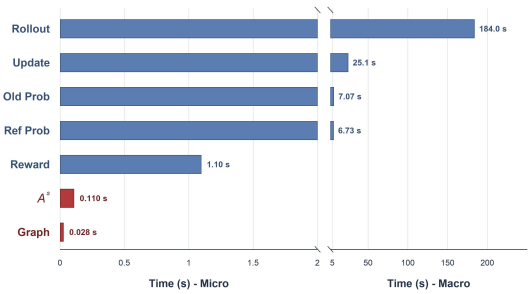


Figure 4: Per-iteration runtime breakdown of the training stages. Blue bars denote stages shared by all group-based methods, while red bars denote the overhead of GraphHCA.

Expectation versus best-case aggregation. Proposition 4.4 relates the difference between GraphHCA and GraphGPO to their aggregation of successor values. To isolate this factor within the same training pipeline, we replace the expectation in Eq. (9) with a $\hat{q}$-weighted power mean,

$$\hat{\Phi}_\omega(s) = \bar{\gamma}\left(\mathbb{E}_{a\sim\hat{q}}\left[\hat{\Phi}_\omega(s'_a)^\omega\right]\right)^{1/\omega}, \qquad \omega \in [1,\infty), \tag{16}$$

while keeping the boundary conditions, numerical floor, and credit computation unchanged. At $\omega = 1$, Eq. (16) recovers the original expectation backup of GraphHCA; increasing ω progressively emphasizes high-valued successors, and $\omega \to \infty$ yields the max backup whose fixed point is $\bar{\gamma}^{d(s)}$ on goal-reachable states, which is ranking-equivalent to GraphGPO's shortest-path criterion. As shown in Table 3, overall success decreases from 95.7% at $\omega = 1$ to 95.3% at $\omega = 5$, 91.8% at $\omega = 10$, and 90.1% under max aggregation. The max variant closely matches GraphGPO at 91.0%. These results are consistent with Proposition 4.4 and indicate that retaining empirical transition frequencies through expectation provides more informative credit than reducing each state to its best observed continuation. Appendix D.2 reports the more ablations.

6 Conclusion

We presented **GraphHCA**, a model-free framework that derives dense step-level credit from the HCA hindsight ratio and estimates it directly from pooled rollout graph. By reducing hindsight credit to state-wise success potentials, GraphHCA avoids auxiliary critics, hindsight models, and additional outcome-conditioned policy evaluations while remaining lightweight in practice. Extensive experiments on ALFWorld, WebShop, and Sokoban show consistent improvements over GRPO and existing step-level baselines across language and vision-language settings. Overall, GraphHCA provides a simple and effective way to extract finer-grained supervision from rollouts already collected for policy optimization.

AI USE STATEMENT

We used large language models to help improve the grammar and wording of the manuscript.

ETHICS STATEMENT

This study evaluates GraphHCA on three publicly available benchmarks: ALFWorld, WebShop, and Sokoban. All agent interactions occur within benchmark environments. The study involves no recruitment of human participants or collection of new personal data.

REPRODUCIBILITY STATEMENT

To facilitate independent replication, we specify the GraphHCA training procedure and report the experimental configurations, including benchmark and model settings, training hyperparameters, evaluation metrics, computational resources, and interaction prompts. The procedure is summarized in Algorithm 1, with experimental settings detailed in Section 5.1 and Appendix D.1, and interaction prompts provided in Appendix E. Results for reinforcement learning methods are averaged over three random seeds to reduce dependence on any single run. The theoretical results are stated under explicit assumptions and supported by proofs in Appendix B.

REFERENCES

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

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 *ACL 2024*, 2024.

Hao Bai, Yifei Zhou, Jiayi Pan, Mert Cemri, Alane Suhr, Sergey Levine, and Aviral Kumar. DigiRL: Training in-the-wild device-control agents with autonomous reinforcement learning. In *NeurIPS 2024*, 2024.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Ming-Hsuan Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

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. *arXiv:2204.05862*, 2022.

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. *arXiv:2605.26684*, 2026.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *NeurIPS 2025*, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Anna Harutyunyan, Will Dabney, Thomas Mesnard, Mohammad Gheshlaghi Azar, Bilal Piot, Nicolas Heess, Hado van Hasselt, Gregory Wayne, Satinder Singh, Doina Precup, and Rémi Munos. Hindsight credit assignment. In *NeurIPS 2019*, 2019.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. *arXiv:2503.09516*, 2025.

Kimi Team. Kimi k1.5: Scaling reinforcement learning with LLMs. *arXiv:2501.12599*, 2025.

Wouter Kool, Herke van Hoof, and Max Welling. Buy 4 reinforce samples, get a baseline for free! In *ICLR 2019 Workshop*, 2019.

Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven Chu-Hong Hoi. CodeRL: Mastering code generation through pretrained models and deep reinforcement learning. In *NeurIPS 2022*, 2022.

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 *ICLR 2024*, 2024.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML 1999*, 1999.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. *NeurIPS 2022*, 2022.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv:1910.00177*, 2019.

Pranav Putta, Edmund Mills, Naman Garg, Sumeet Ramesh Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. Agent Q: Advanced reasoning and learning for autonomous AI agents. *arXiv:2408.07199*, 2024.

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

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS 2023*, 2023.

Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy Lillicrap. Android in the Wild: A large-scale dataset for android device control. In *NeurIPS 2023 Datasets and Benchmarks Track*, 2023.

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 *NeurIPS 2023*, 2023.

Max-Philipp B. Schrader. gym-sokoban. `https://github.com/mpSchrader/gym-sokoban`, 2018.

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv:2402.03300*, 2024.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *NeurIPS 2023*, 2023.

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 *ICLR 2021*, 2021.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. Learning to summarize with human feedback. In *NeurIPS 2020*, 2020.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *NeurIPS 1999*, 1999.

Hui-Ze Tan, Xiao-Wen Yang, Hao Chen, Jie-Jing Shao, Yi Wen, Yuteng Shen, Weihong Luo, Xiku Du, Lan-Zhe Guo, and Yu-Feng Li. Hindsight credit assignment for long-horizon LLM agents. *arXiv:2603.08754*, 2026.

Gemini Team. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. AppWorld: A controllable world of apps and people for benchmarking interactive coding agents. In *ACL 2024*, 2024.

Jiawei Wang, Jiacai Liu, Yuqian Fu, Yingru Li, Xintao Wang, Yuan Lin, Yu Yue, Lin Zhang, Yang Wang, and Ke Wang. Harnessing uncertainty: Entropy-modulated policy gradients for long-horizon LLM agents. In *ICML 2026*, 2026.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS 2022*, 2022.

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, 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. *arXiv preprint arXiv:2412.15115*, 2024a.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *NeurIPS 2024*, 2024b.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *NeurIPS 2022*, 2022.

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 *ICLR 2023*, 2023.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. ArCHer: Training language model agents via hierarchical multi-turn RL. In *ICML 2024*, 2024.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul F. Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

APPENDIX

A ALGRITHOM

Algorithm 1 Pseudo-code for GraphHCA

Require: Initial policy π_θ, task distribution $p(X)$, propagation discount $\bar{\gamma}$, numerical floor ϵ, group size N, maximum horizon T

for each training step **do**
  $\theta_{\text{old}} \leftarrow \theta$
  // Multi-step rollout
  Sample task $x \sim p(X)$
  Roll out N trajectories under $\pi_{\theta_{\text{old}}}$ until termination or step T
  // Graph estimation
  Construct the pooled rollout graph and empirical weights $\hat{q}(a \mid s)$
  Compute $\hat{\Phi}$ by iterating Eq. (9)
  Compute $\hat{\Psi}(s) = \log \max\{\hat{\Phi}(s), \epsilon\}$
  // Hindsight credit assignment
  Compute $\hat{\ell}_t = \hat{\Psi}(s_{t+1}) - \hat{\Psi}(s_t)$
  Compute same-state advantages A_t^{s} by Eq. (11)
  // Policy optimization
  trajectory-level advantages A^{T}
  Compute $\hat{A}_t$ by Eq. (12)
  Update θ using the objective in Eq. (13)
end for

Algorithm 1 summarizes the GraphHCA training procedure. At each step, a group of rollouts is aggregated into a shared transition graph, from which the discounted graph value and transition-level hindsight credit are computed. The resulting same-state step advantage is combined with the episode-level advantage for policy optimization. GraphHCA requires neither a learned critic nor an auxiliary hindsight model or additional outcome-conditioned policy evaluation, as all step-level credit is derived from rollouts already collected for training. The computational overhead of these graph operations is negligible compared with rollout generation and policy optimization, as shown in Figure 4.

B PROOFS

This appendix provides proofs for the four propositions in Sec. 4: the closed-form hindsight ratio (Sec. B.1), the contraction and convergence of the graph backup (Sec. B.2), the hindsight KL identity and bounded graph credit (Sec. B.3), and the relation between expectation-based and best-case aggregation (Sec. B.4). We first fix the notation shared across these proofs.

Exact quantities. Consider a terminal-goal task with deterministic transitions $s'_a = f(s, a)$ and terminal reward $R(\tau) = \mathbf{1}[s_T \in \mathcal{G}]$. For the exact quantities, we adopt the state-sufficiency assumption in Sec. 4.1. Trajectories are generated by the behavior policy q, and $G = \{s_T \in \mathcal{G}\}$ denotes the success event. For any state s, define

$$\Phi(s) = P(G \mid s), \qquad \Phi(s) = 1 \text{ for } s \in \mathcal{G}, \quad \Phi(s) = 0 \text{ for } s \in \mathcal{F}. \tag{17}$$

The corresponding log-scale success potential is

$$\Psi(s) = \log \max\{\Phi(s), \epsilon\}, \qquad \epsilon \in (0, 1). \tag{18}$$

We write $h(a \mid s, G) = P(a \mid s, G)$ for the success-conditioned hindsight distribution.

Empirical graph quantities. For empirical graph quantities, identical environment states are pooled as in Sec. 4.2, and each observed action is weighted by its empirical frequency

$$\hat{q}(a \mid s) = \frac{n(s, a)}{n(s)}. \tag{19}$$

Let $\mathcal{V}$ denote the set of graph nodes. We define the backup operator $\mathcal{T} : \mathbb{R}^{\mathcal{V}} \to \mathbb{R}^{\mathcal{V}}$ by

$$(\mathcal{T}\phi)(s) = \begin{cases} 1, & s \in \mathcal{G}, \\ 0, & s \in \mathcal{F}, \\ \bar{\gamma} \sum_a \hat{q}(a \mid s)\phi(s'_a), & \text{otherwise.} \end{cases} \tag{20}$$

A fixed point of $\mathcal{T}$ therefore satisfies exactly the discounted graph recursion in Eq. (9) together with its terminal boundary conditions. We denote this fixed point by $\hat{\Phi}$ and define the corresponding graph success potential as $\hat{\Psi}(s) = \log \max\{\hat{\Phi}(s), \epsilon\}$.

B.1 Closed-form hindsight ratio (Proposition 4.1)

Deterministic transitions identify the successor state, whose continuation success probability is independent of the preceding history and time step under the state-sufficiency assumption.

Lemma B.1 (Successor identity)**.** *Under the assumptions of Proposition 4.1, for any non-terminal state s and action a with $q(a \mid s) > 0$,*

$$P(G \mid s, a) = \Phi(s'_a). \tag{21}$$

Proof. Under deterministic transitions, taking action a at state s leads to the unique successor $s'_a = f(s, a)$. By the state-sufficiency assumption, all histories reaching s'_a have the same conditional probability of eventual success under q, irrespective of the time step. Therefore,

$$P(G \mid s, a) = P(G \mid s'_a) = \Phi(s'_a). \tag{22}$$

□

Proof of Proposition 4.1. For any non-terminal state s, marginalizing over the first action gives

$$\Phi(s) = P(G \mid s) \tag{23}$$

$$= \sum_a q(a \mid s) P(G \mid s, a) \tag{24}$$

$$= \sum_a q(a \mid s)\Phi(s'_a) = \mathbb{E}_{a \sim q(\cdot \mid s)}\big[\Phi(s'_a)\big], \tag{25}$$

where the third equality follows from Lemma B.1. This proves Eq. (6).

For any state with $\Phi(s) > 0$, Bayes' rule gives

$$h(a \mid s, G) = P(a \mid s, G) = \frac{q(a \mid s) P(G \mid s, a)}{P(G \mid s)}. \tag{26}$$

For actions in the support of $q(\cdot \mid s)$, dividing by $q(a \mid s)$ and applying Lemma B.1 yields

$$\rho(s, a) = \frac{P(a \mid s, G)}{q(a \mid s)} = \frac{P(G \mid s, a)}{P(G \mid s)} = \frac{\Phi(s'_a)}{\Phi(s)}, \tag{27}$$

which proves Eq. (7). □

The corresponding log-scale credit is $\ell(s, a) = \Psi(s'_a) - \Psi(s)$. Whenever the floor in Eq. (5) is inactive at both states, Eq. (7) gives

$$\ell(s, a) = \log \rho(s, a). \tag{28}$$

Independently of whether the floor is active, the potential differences telescope along any realized trajectory:

$$\sum_{t=0}^{T-1} \ell(s_t, a_t) = \sum_{t=0}^{T-1} \big[\Psi(s_{t+1}) - \Psi(s_t)\big] = \Psi(s_T) - \Psi(s_0). \tag{29}$$

B.2 Contraction and convergence (Proposition 4.2)

Proof of Proposition 4.2. Let $\phi_1, \phi_2 \in \mathbb{R}^{\mathcal{V}}$. For terminal states, $\mathcal{T}\phi_1 = \mathcal{T}\phi_2$ by definition. For any non-terminal state s,

$$\big|(\mathcal{T}\phi_1)(s) - (\mathcal{T}\phi_2)(s)\big| = \bar{\gamma}\left|\sum_a \hat{q}(a \mid s)\big[\phi_1(s'_a) - \phi_2(s'_a)\big]\right| \tag{30}$$

$$\leq \bar{\gamma}\sum_a \hat{q}(a \mid s)\big|\phi_1(s'_a) - \phi_2(s'_a)\big| \tag{31}$$

$$\leq \bar{\gamma}\|\phi_1 - \phi_2\|_\infty. \tag{32}$$

Taking the supremum over $s \in \mathcal{V}$ gives

$$\|\mathcal{T}\phi_1 - \mathcal{T}\phi_2\|_\infty \leq \bar{\gamma}\|\phi_1 - \phi_2\|_\infty. \tag{33}$$

Since $\bar{\gamma} \in (0, 1)$, $\mathcal{T}$ is a contraction on the complete metric space $(\mathbb{R}^{\mathcal{V}}, \|\cdot\|_\infty)$. Banach's fixed-point theorem therefore guarantees a unique fixed point $\hat{\Phi}$, and for any initialization ϕ_0,

$$\|\mathcal{T}^k\phi_0 - \hat{\Phi}\|_\infty \leq \bar{\gamma}^k\|\phi_0 - \hat{\Phi}\|_\infty, \tag{34}$$

which establishes geometric convergence, independently of whether the graph contains cycles.

It remains to identify the fixed point. Let $\kappa = \inf\{t \geq 0 : s_t \in \mathcal{G} \cup \mathcal{F}\}$ be the termination time of a graph rollout generated by $a_t \sim \hat{q}(\cdot \mid s_t)$, and define

$$\tilde{\Phi}(s) = \mathbb{E}_{\hat{q}}[\bar{\gamma}^\kappa \mathbf{1}\{s_\kappa \in \mathcal{G}\}\mathbf{1}\{\kappa < \infty\} \mid s_0 = s]\,. \tag{35}$$

The terminal boundary gives $\tilde{\Phi} = 1$ on $\mathcal{G}$ and $\tilde{\Phi} = 0$ on $\mathcal{F}$. For any non-terminal s, conditioning on the first action yields

$$\tilde{\Phi}(s) = \bar{\gamma}\sum_a \hat{q}(a \mid s)\tilde{\Phi}(s'_a), \tag{36}$$

so $\mathcal{T}\tilde{\Phi} = \tilde{\Phi}$. By uniqueness of the fixed point, $\tilde{\Phi} = \hat{\Phi}$. Hence, $\hat{\Phi}$ is exactly the $\bar{\gamma}$-discounted expected terminal success under the empirical graph policy $\hat{q}$. □

B.3 Hindsight KL identity and bounded step credit (Proposition 4.3)

Proof of Proposition 4.3. Fix a state satisfying the conditions of Proposition 4.3. The floor in Eq. (5) is inactive at s and at every successor reached with positive probability under q. For every action in the support of $q(\cdot \mid s)$, Proposition 4.1 gives

$$\ell(s, a) = \log\frac{h(a \mid s, G)}{q(a \mid s)}. \tag{37}$$

Taking expectation under $q(\cdot \mid s)$ yields

$$\mathbb{E}_{a\sim q(\cdot|s)}[\ell(s, a)] = \sum_{a:q(a|s)>0} q(a \mid s)\log\frac{h(a \mid s, G)}{q(a \mid s)} \tag{38}$$

$$= -\mathrm{KL}(q(\cdot \mid s)\,\|\,h(\cdot \mid s, G)) \leq 0, \tag{39}$$

where equality holds if and only if $q(\cdot \mid s) = h(\cdot \mid s, G)$ almost everywhere.

Independently, for the empirical graph, the terminal boundary and the discounted backup imply $0 \leq \hat{\Phi}(s) \leq 1$ for every graph node. Whether or not the floor is active, it gives

$$\hat{\Psi}(s) = \log\max\{\hat{\Phi}(s), \epsilon\} \in [\log\epsilon, 0]. \tag{40}$$

Therefore, for every observed transition,

$$|\hat{\ell}(s, a)| = |\hat{\Psi}(s'_a) - \hat{\Psi}(s)| \leq \log(1/\epsilon), \tag{41}$$

which proves the boundedness of the graph credit. □

B.4 Expectation versus best-case aggregation (Proposition 4.4)

Under the unit-cost convention of Cheng et al. (2026), define the max-aggregation counterpart of $\mathcal{T}$ by

$$(\mathcal{T}^{\max}\phi)(s) = \begin{cases} 1, & s \in \mathcal{G}, \\ 0, & s \in \mathcal{F}, \\ \bar{\gamma} \max\limits_{a:n(s,a)>0} \phi(s'_a), & \text{otherwise.} \end{cases} \tag{42}$$

For any $\phi_1, \phi_2 \in \mathbb{R}^{\mathcal{V}}$, the terminal values cancel, while for any non-terminal state s,

$$\left|(\mathcal{T}^{\max}\phi_1)(s) - (\mathcal{T}^{\max}\phi_2)(s)\right| \le \bar{\gamma} \max_{a:n(s,a)>0} \left|\phi_1(s'_a) - \phi_2(s'_a)\right| \le \bar{\gamma}\|\phi_1 - \phi_2\|_\infty. \tag{43}$$

Hence, $\mathcal{T}^{\max}$ is a $\bar{\gamma}$-contraction in the sup norm and therefore admits a unique fixed point, denoted $\Phi^{\max}$.

Proof of Proposition 4.4. For any ϕ and non-terminal state s,

$$(\mathcal{T}\phi)(s) = \bar{\gamma} \sum_a \hat{q}(a \mid s)\phi(s'_a) \le \bar{\gamma} \max_{a:n(s,a)>0} \phi(s'_a) = (\mathcal{T}^{\max}\phi)(s). \tag{44}$$

Both operators are monotone. Starting from the same initialization therefore gives $\mathcal{T}^k\phi_0 \le (\mathcal{T}^{\max})^k\phi_0$ for every k, and taking $k \to \infty$ yields

$$\hat{\Phi}(s) \le \Phi^{\max}(s). \tag{45}$$

It remains to identify $\Phi^{\max}$. Let $d(s)$ denote the shortest-path distance from s to $\mathcal{G}$ on the empirical graph, with $d(s) = 0$ for $s \in \mathcal{G}$. For every goal-reachable non-terminal state,

$$d(s) = 1 + \min_{a:n(s,a)>0} d(s'_a). \tag{46}$$

Define $v(s) = \bar{\gamma}^{\,d(s)}$ for goal-reachable states and $v(s) = 0$ otherwise. The terminal conditions give $v = 1$ on $\mathcal{G}$ and $v = 0$ on $\mathcal{F}$. For every goal-reachable non-terminal state,

$$(\mathcal{T}^{\max}v)(s) = \bar{\gamma} \max_{a:n(s,a)>0} \bar{\gamma}^{\,d(s'_a)} \tag{47}$$

$$= \bar{\gamma}^{\,1+\min_{a:n(s,a)>0} d(s'_a)} \tag{48}$$

$$= \bar{\gamma}^{\,d(s)} = v(s), \tag{49}$$

where the second equality follows from $\bar{\gamma} \in (0,1)$. If s is not goal-reachable, none of its observed successors is goal-reachable, so $(\mathcal{T}^{\max}v)(s) = 0 = v(s)$. Thus, v is a fixed point of $\mathcal{T}^{\max}$, and uniqueness implies

$$\Phi^{\max}(s) = \bar{\gamma}^{\,d(s)} \qquad \text{for every goal-reachable } s. \tag{50}$$

To characterize equality, define $\Delta(s) = \Phi^{\max}(s) - \hat{\Phi}(s) \ge 0$. For any non-terminal state,

$$\Delta(s) = \bar{\gamma}\left[\max_a \Phi^{\max}(s'_a) - \sum_a \hat{q}(a \mid s)\hat{\Phi}(s'_a)\right] \tag{51}$$

$$= \bar{\gamma}\left[\max_a \Phi^{\max}(s'_a) - \sum_a \hat{q}(a \mid s)\Phi^{\max}(s'_a)\right] + \bar{\gamma}\sum_a \hat{q}(a \mid s)\Delta(s'_a). \tag{52}$$

Both terms are non-negative. Hence, $\Delta(s) = 0$ if and only if $\hat{q}(\cdot \mid s)$ is supported on maximizing successors and $\Delta(s'_a) = 0$ for every successor reached with positive probability. Recursively applying the same argument gives the stated equality condition at s and along its $\hat{q}$-reachable continuation.

Finally, for a fixed state s, $\hat{\Psi}(s)$ is independent of the action, so $\hat{\ell}(s,a) = \hat{\Psi}(s'_a) - \hat{\Psi}(s)$ orders the observed actions by their successor potential $\hat{\Psi}(s'_a)$. Under unit transition cost, GraphGPO uses

$$R^{\mathrm{G}}(s,a,s'_a) \propto \bar{\gamma}^{\,d(s'_a)+1}, \tag{53}$$

which favors smaller $d(s'_a)$ because $\bar{\gamma} \in (0,1)$. Since $\Phi^{\max}(s'_a) = \bar{\gamma}^{\,d(s'_a)}$, GraphGPO's shortest-path score is ranking-equivalent to the max-aggregation counterpart, whereas GraphHCA evaluates successors through the empirical expectation under $\hat{q}$. This establishes the expectation-versus-best-case distinction. □

C Benchmark Details And Compared Methods

ALFWorld. ALFWorld (Shridhar et al., 2021) aligns the ALFRED household task suite with an interactive text simulator, so the agent must carry out long-horizon, multi-step decision making from textual observations alone. It contains 3,827 task instances spanning six categories of common household activity, namely Pick, Clean, Cool, Look, Heat, and Pick2, which differ in how many subgoals a task requires and therefore in how long the chain of decisions to be credited is. At each turn the agent receives a text description of the room together with its inventory, and emits a template action for navigation, object manipulation, or interaction with a receptacle. Reward is granted only once the full instruction is satisfied, and we report the success rate on each of the six categories as well as the overall success rate.

WebShop. WebShop (Yao et al., 2022) is a large-scale web-based interactive environment that places the agent in a realistic online shopping scenario over more than 1.1 million real product pages and approximately 12,000 human-written user instructions. Given an instruction that names a target product with its desired attributes and a price constraint, the agent issues search queries, browses result pages, opens candidate products, selects attribute options, and finally purchases one item. An episode is scored by how well the purchased item matches the requested attributes, and it counts as a success when that score is maximal, so we report the average task score alongside the average success rate.

Sokoban. Sokoban (Schrader, 2018) is an interactive puzzle game in which the agent pushes boxes onto target squares of a grid. We adopt the 6×6 configuration and present each state as a rendered image, so the policy is a vision-language model that grounds its plan in pixels rather than in text. The game is a stringent test of multi-step planning because pushes are irreversible: a box driven against a wall or into a corner can make the level unsolvable, and the agent receives no signal until the episode ends. We report the average success rate.

Compared Methods. Under an identical agent framework and protocol, we compare Graph-HCA against *closed-source LLMs* (GPT-4o (Achiam et al., 2023), Gemini-2.5-Pro (Team, 2023)), *prompting-based agents* (the base Qwen2.5 checkpoint, ReAct (Yao et al., 2023), Reflexion (Shinn et al., 2023)), *critic-based RL* (PPO (Schulman et al., 2017)), and *critic-free group-based RL* (RLOO (Kool et al., 2019; Ahmadian et al., 2024), GRPO (Shao et al., 2024), EMPG (Wang et al., 2026), GiGPO (Feng et al., 2025), HCAPO (Tan et al., 2026), GraphGPO (Cheng et al., 2026)). The last family is the closest comparison, as GiGPO, HCAPO, and GraphGPO differ from GraphHCA only in how step-level credit is obtained. On Sokoban we compare against its group-based members GRPO, GiGPO, and GraphGPO.

D Implementation Details and Hyperparameter Sensitivity

D.1 Training configuration

All experiments are conducted for 150 training steps across all benchmarks and model configurations. For experiments using Qwen2.5-1.5B-Instruct and Qwen2.5-VL-3B-Instruct, training is performed on 4 NVIDIA A100 GPUs with 80 GB memory each. For experiments using the larger Qwen2.5-7B-Instruct model, we employ 8 NVIDIA A100 GPUs with 80 GB memory each to accommodate the increased computational and memory requirements.

D.2 More Ablations

Composition of the two advantage terms. Equation Eq. (12) combines the trajectory-level advantage A^{T} with the state-level advantage A^{s}. Table 4 examines their complementarity by removing each term in turn and varying the weight w_{s}. Setting $w_{\mathrm{s}} = 0$ recovers GRPO and reduces overall success from 95.7% to 71.1%, a drop of 24.6 points. Using A^{s} alone achieves 92.5%, but remains 3.2 points below the full objective, consistent with the role of A^{T} in providing trajectory-level supervision, including at singleton states where $A^{\mathrm{s}} = 0$. Among the mixed settings, performance increases from 93.4% at $w_{\mathrm{s}} = 0.5$ to 94.9% at $w_{\mathrm{s}} = 0.7$ and peaks at 95.7% with the default

Table 4: ALFWorld ablation on the composition of the combined advantage of Eq. (12) with Qwen2.5-1.5B-Instruct: per-subtask and overall success rate (%) over three random seeds. The upper block removes one of the two terms, where $w_{\mathrm{s}}=0$ recovers GRPO exactly (reproduced from Table 1). The lower block keeps both and varies their relative weight around the default $w_{\mathrm{s}}=1$ (highlighted). The per-column best is in **bold**.

Advantage	Pick	Clean	Cool	Look	Heat	Pick2	All
A^{T} only ($w_{\mathrm{s}}=0$, GRPO)	$88.2_{\pm 2.9}$	$72.6_{\pm 15.9}$	$67.5_{\pm 7.5}$	$43.3_{\pm 18.3}$	$76.6_{\pm 0.4}$	$51.6_{\pm 9.5}$	$71.1_{\pm 1.6}$
A^{step} only	$97.1_{\pm 0.05}$	$\mathbf{100.0}_{\pm 0.0}$	$82.5_{\pm 2.5}$	$81.25_{\pm 6.3}$	$\mathbf{97.6}_{\pm 2.4}$	$84.2_{\pm 5.3}$	$92.5_{\pm 0.4}$
$w_{\mathrm{s}}=0.5$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{100.0}_{\pm 0.0}$	$82.5_{\pm 2.5}$	$68.8_{\pm 6.3}$	$95.2_{\pm 0.0}$	$92.1_{\pm 2.6}$	$93.4_{\pm 0.5}$
$w_{\mathrm{s}}=0.7$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{95.0}_{\pm 5.0}$	$77.1_{\pm 10.4}$	$92.9_{\pm 7.2}$	$93.8_{\pm 0.9}$	$94.9_{\pm 0.4}$
$w_{\mathrm{s}}=1$ (default)	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{100.0}_{\pm 0.0}$	$85.7_{\pm 0.7}$	$\mathbf{89.9}_{\pm 2.4}$	$96.7_{\pm 3.3}$	$\mathbf{95.8}_{\pm 4.2}$	$\mathbf{95.7}_{\pm 1.2}$

Table 5: ALFWorld ablation over the propagation discount $\bar{\gamma}$ of Eq. (9) with Qwen2.5-1.5B-Instruct: per-subtask and overall success rate (%) over three random seeds. The default $\bar{\gamma}=0.95$ is highlighted. Small $\bar{\gamma}$ drives the potential toward the distance surrogate $\bar{\gamma}^{\,d(s)}$ of Eq. (15). The per-column best is in **bold**.

$\bar{\gamma}$	Pick	Clean	Cool	Look	Heat	Pick2	All
0.55	$96.4_{\pm 3.6}$	$95.5_{\pm 4.6}$	$85.8_{\pm 0.8}$	$79.2_{\pm 4.2}$	$\mathbf{97.6}_{\pm 2.4}$	$90.2_{\pm 4.5}$	$90.7_{\pm 3.2}$
0.75	$98.6_{\pm 1.5}$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{87.5}_{\pm 2.5}$	$87.5_{\pm 0.0}$	$\mathbf{97.6}_{\pm 2.4}$	$92.1_{\pm 2.7}$	$93.3_{\pm 1.6}$
0.95 (default)	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{100.0}_{\pm 0.0}$	$85.7_{\pm 0.7}$	$\mathbf{89.9}_{\pm 2.4}$	$96.7_{\pm 3.3}$	$\mathbf{95.8}_{\pm 4.2}$	$\mathbf{95.7}_{\pm 1.2}$

$w_{\mathrm{s}}=1$. These results support the complementary roles of the two advantage terms rather than either term subsuming the other.

Effect of the propagation discount. A small propagation discount $\bar{\gamma}$ in Eq. (9) can weaken credit assignment through two related mechanisms. First, stronger discounting gives greater relative weight to short successful continuations, allowing path length to outweigh differences in success probability. Even before clipping, the potential can therefore favor a shorter but less reliable route over a longer, more reliable one. Second, discounting interacts with the floor $\epsilon=10^{-2}$ in $\hat{\Psi}(s)=\log\max\{\hat{\Phi}(s),\epsilon\}$. By Eq. (15), $\hat{\Phi}(s)\le\bar{\gamma}^{d(s)}$, so states with $d(s)>\log\epsilon/\log\bar{\gamma}$ are necessarily clipped to $\log\epsilon$. This occurs from distances of 8 and 17 steps for $\bar{\gamma}=0.55$ and 0.75, respectively, well within the rollout horizon $T=50$, erasing potential differences between clipped states. For $\bar{\gamma}=0.95$, the discount factor remains above the floor over T steps ($\bar{\gamma}^{T}\approx 0.077>\epsilon$), although low success probabilities can still trigger clipping. Table 5 is consistent with these mechanisms: overall success rises from 90.7% to 93.3% and 95.7% as $\bar{\gamma}$ increases from 0.55 to 0.75 and 0.95. We therefore set $\bar{\gamma}=0.95$, which reduces distance bias and clipping caused by propagation depth while retaining the contraction guarantee of Proposition 4.2.

Sensitivity to the numerical floor. The floor ϵ keeps $\hat{\Psi}(s)=\log\max\{\hat{\Phi}(s),\epsilon\}$ finite when $\hat{\Phi}(s)$ approaches zero. Under the default $\bar{\gamma}=0.95$ and $T=50$, the attenuation over the full horizon is $\bar{\gamma}^{T}\approx 0.077$, which exceeds every tested floor. The floor therefore never clips a state because of its propagation depth, and it binds only states whose continuations rarely or never reach the goal. Changing ϵ thus shifts the potential of these failure states, while the potential of every state with $\hat{\Phi}(s)\ge 10^{-2}$ remains unchanged. Consistent with this analysis, Table 6 reports overall success of 95.7%, 94.1%, and 94.9% for $\epsilon=10^{-2}$, 10^{-4}, and 10^{-6}, a variation of at most 1.6 points that is comparable to the standard deviation across seeds. These results indicate that ϵ acts as a numerical safeguard rather than a sensitive hyperparameter, provided that it stays below $\bar{\gamma}^{T}$.

E Prompt Templates

Figures 5, 6, and 7 present the prompt template and an illustrative filled example for ALFWorld, WebShop, and Sokoban, respectively. All three instantiate a common schema: the agent is given

Table 6: ALFWorld ablation over the failure floor ϵ of Eq. (10) with Qwen2.5-1.5B-Instruct: per-subtask and overall success rate (%) over three random seeds. The default $\epsilon = 10^{-2}$ is highlighted. The per-column best is in **bold**.

ϵ	Pick	Clean	Cool	Look	Heat	Pick2	All
10^{-6}	**100.0**$_{\pm 0.0}$	**100.0**$_{\pm 0.0}$	**90.0**$_{\pm 0.0}$	81.3$_{\pm 6.3}$	95.3$_{\pm 4.8}$	89.5$_{\pm 5.3}$	94.9$_{\pm 2.0}$
10^{-4}	98.6$_{\pm 1.5}$	**100.0**$_{\pm 0.0}$	**90.0**$_{\pm 10.0}$	87.5$_{\pm 0.0}$	**97.6**$_{\pm 2.4}$	92.1$_{\pm 2.6}$	94.1$_{\pm 1.2}$
10^{-2} (default)	**100.0**$_{\pm 0.0}$	**100.0**$_{\pm 0.0}$	85.7$_{\pm 0.7}$	**89.9**$_{\pm 2.4}$	96.7$_{\pm 3.3}$	**95.8**$_{\pm 4.2}$	**95.7**$_{\pm 1.2}$

the task instruction, the step count, at most the two most recent observation-action pairs, and the current observation with its admissible actions, and is asked to reason before committing to a single action. They diverge only where the environment demands it, Sokoban adding a symbol legend and a warning that pushes are irreversible, and offering a fixed set of four moves rather than a state-dependent action set. Gold text denotes template placeholders, while green and magenta mark the reasoning and action delimiters, respectively.

Prompt Template for ALFWorld

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.

ALFWorld: Filled Example

You are an expert agent operating in the ALFRED Embodied Environment. Your task is to: **put a clean egg in microwave**. Prior to this step, you have already taken **3** step(s). Below are the most recent **2** observations and the corresponding actions you took:

[t=2] *obs*: The fridge 1 is closed.
[t=2] *act*: open fridge 1
[t=3] *obs*: You open the fridge 1. In it you see a egg 1, a lettuce 1, a mug 1, and a tomato 2.
[t=3] *act*: take egg 1 from fridge 1

You are now at step **4** and your current observation is: You pick up the egg 1 from the fridge 1. Your admissible actions of the current situation are: [go to sinkbasin 1, go to microwave 1, go to countertop 1, put egg 1 in/on fridge 1, examine egg 1].

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 5: Prompt template and illustrative filled example for ALFWorld.

Prompt Template for WebShop

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.

WebShop: Filled Example

You are an expert autonomous agent operating in the WebShop e-commerce environment. Your task is to: **i want a pre-seasoned 8-inch cast iron skillet made of carbon steel with a heat-resistant handle, and price lower than 40.00 dollars**. Prior to this step, you have already taken **2** step(s). Below are the most recent **2** observations and the corresponding actions you took:

[t=1] *obs*: WebShop [SEP] Instruction: i want a pre-seasoned 8-inch cast iron skillet ... [SEP] Search
[t=1] *act*: search[pre-seasoned 8 inch cast iron skillet carbon steel]
[t=2] *obs*: [Page 1 (Total results: 50)] [SEP] B07XQK6WZP [SEP] Pre-Seasoned Cast Iron Skillet 8-Inch with Heat-Resistant Silicone Handle [SEP] $29.99 [SEP] B08D8Q2JG9 [SEP] Lodge L5SK3 8-Inch Pre-Seasoned Cast-Iron Skillet [SEP] $19.90 [SEP] ...
[t=2] *act*: click[B07XQK6WZP]

You are now at step **3** and your current observation is: Pre-Seasoned Cast Iron Skillet 8-Inch with Heat-Resistant Silicone Handle [SEP] Price: $29.99 [SEP] Size: [7-inch] [8-inch] [10-inch] [SEP] Color: [black] [red] [SEP] Description: Heavy-duty cast iron, oven-safe, silicone sleeve included. [SEP] Buy Now. Your admissible actions of the current situation are: [click[8-inch], click[7-inch], click[10-inch], click[black], click[red], click[Description], click[Features], click[Reviews], click[Buy Now], click[< Prev]].

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 6: Prompt template and illustrative filled example for WebShop.

Prompt Template for Sokoban

You are an expert agent operating in the Sokoban environment. Your goal is to push every box onto a target square, and the level is solved once no box is left off a target.

The grid is drawn with the following symbols:

- `#` a wall, which neither you nor a box can enter.
- `_` an empty floor square, which you can walk on and push a box onto.
- `O` a target square, on which a box has to be placed.
- `X` a box.
- `P` you, the player.
- √ a box that already rests on a target.
- `S` you standing on a target.

Boxes can only be pushed and never pulled, so a box driven into a corner or flattened against a wall may become impossible to move again and can leave the level unsolvable. Plan ahead and keep every box on a square from which it can still reach a target.

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 are: [“up”, “down”, “left”, “right”].

Now it’s your turn to take one action for the current step. You should first reason step-by-step about the current situation, locating the boxes and the targets, planning a path that pushes one box toward a target, and checking that the push does not trap that box. 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.

Sokoban: Filled Example

You are an expert agent operating in the Sokoban environment. *(The symbol legend and the rules repeat the template above and are omitted here.)*

Prior to this step, you have already taken **2** step(s). Below are the most recent **2** observations and the corresponding actions you took:

[t=1] *obs*

```
# # # # # #
# _ _ O _ #
# _ _ X _ #
# _ _ P _ #
# _ X _ O #
# # # # # #
```

[t=1] *act*: up

[t=2] *obs*

```
# # # # # #
# _ _ √ _ #
# _ _ P _ #
# _ _ _ _ #
# _ X _ O #
# # # # # #
```

[t=2] *act*: left

You are now at step **3** and your current observation is:

```
# # # # # #
# _ _ √ _ #
# _ P _ _ #
# _ _ _ _ #
# _ X _ O #
# # # # # #
```

Your admissible actions are: [“up”, “down”, “left”, “right”].

Now it’s your turn to take one action for the current step. You should first reason step-by-step about the current situation, locating the boxes and the targets, planning a path that pushes one box toward a target, and checking that the push does not trap that box. 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: Prompt template and illustrative filled example for Sokoban.