REWARDFLOW: TOPOLOGY-AWARE REWARD PROPAGATION ON STATE GRAPHS FOR AGENTIC RL WITH LARGE LANGUAGE MODELS

Xiao Feng[1] **Bo Han**[1†] **Zhanke Zhou**[1] **Jiaqi Fan**[2]
Jiangchao Yao[3] **Ka Ho Li**[2] **Dahai Yu**[2] **Michael Kwok-Po Ng**[4]
[1]TMLR Group, Hong Kong Baptist University [2]TCL Corporate Research (HK) Co., Ltd
[3]CMIC, Shanghai Jiao Tong University [4]Department of Mathematics, Hong Kong Baptist University

ABSTRACT

Reinforcement learning (RL) holds significant promise for enhancing the agentic reasoning capabilities of large language models (LLMs) with external environments. However, the inherent sparsity of terminal rewards hinders fine-grained, state-level optimization. Although process reward modeling offers a promising alternative, training dedicated reward models often entails substantial computational costs and scaling difficulties. To address these challenges, we introduce REWARDFLOW, a lightweight method for estimating state-level rewards tailored to agentic reasoning tasks. REWARDFLOW leverages the intrinsic topological structure of states within reasoning trajectories by constructing state graphs. This enables an analysis of state-wise contributions to success, followed by topology-aware graph propagation to quantify contributions and yield objective, state-level rewards. When integrated as dense rewards for RL optimization, REWARDFLOW substantially outperforms prior RL baselines across four agentic reasoning benchmarks, demonstrating superior performance, robustness, and training efficiency. RewardFlow is publicly available at https://github.com/tmlr-group/RewardFlow.

1 INTRODUCTION

Large Language Models (LLMs) have demonstrated strong reasoning capabilities, making them compelling foundations for agents that solve real-world tasks by interacting with environments, including computer control (Gou et al., 2025) and GUI operation (Qin et al., 2025). In this paradigm, agentic reinforcement learning (RL) plays a central role, enhancing both capability and reliability by optimizing expected behavior under environment-provided rewards.

Such agent-environment interaction unfolds over multiple turns, producing long-horizon reasoning trajectories with many intermediate states. However, optimization is often hindered by the sparse-reward structure of agentic environments: most provide no state-wise feedback, yielding only a terminal evaluation upon task termination with completion, failure, or truncation. Consequently, agentic RL is driven by a coarse, trajectory-level signal rather than fine-grained, state-level guidance, weakening credit assignment and leading to insufficient training.

Prior work seeks to recover state-wise rewards but typically relies on training separate reward models with human-annotated data (Lightman et al., 2023; Wang et al., 2025a). This dependence incurs substantial data and computational costs, limiting optimization efficiency and scalability. This motivates our central question: *How can we objectively estimate process rewards for intermediate states in agentic tasks without training reward models?*

This work introduces REWARDFLOW to address this challenging question. The key idea is to exploit informative signals in the intrinsic topological relationships among states within reasoning trajectories (Fig. 1). We treat graph-based proximity to success as a surrogate process reward: states closer to a successful terminal state receive higher value, enabling objective estimation even for states in failed

†Correspondence to Bo Han (bhanml@comp.hkbu.edu.hk).

trajectories. Moreover, states that frequently appear in successful trajectories and serve as critical junctions in the state graph are identified as pivotal and rewarded accordingly.

Motivated by this idea, REWARDFLOW constructs a *state graph* for each task. As in the left panel of Fig. 2, the graph aggregates equivalent states collected from one agentic task into unique nodes, consolidating shared states and transitions across multiple trajectories for the same task. This structure reveals intrinsic state-wise properties, facilitating the analysis of each state's potential for success.

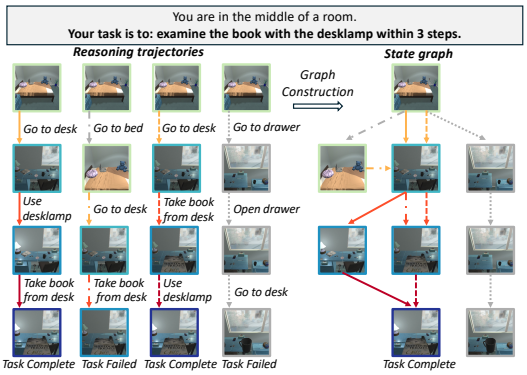


Figure 1: Trajectories sampled from the policy are merged into nodes by equivalent states, with directed edges representing observed actions. Node and edge colors encode distance to success (darker = closer to terminal success). Grey nodes and edges indicate no path to success. The resulting graph reveals the task's intrinsic topology and enables reliable process reward modeling.

Building on the state graph, REWARDFLOW employs graph-based propagation methods (*e.g.*, Breadth-First Search, BFS (Moore, 1959)) to estimate state-wise rewards. As shown in the middle panel of Fig. 2, outcome rewards from successful terminal states are propagated backward to intermediate states along observed transitions. These rewards capture topological signals, thereby quantifying the potential for success in each intermediate state. As a result, REWARDFLOW provides a principled and objective means of estimating process rewards without dependence on external reward models.

Leveraging state-wise rewards, REWARDFLOW optimizes the policy model through RL. As depicted in the right panel of Fig. 2, REWARDFLOW first converts propagated state values into dense action-level rewards via successor–current state value differences. It then computes synergistic advantages by combining these local advantages with global trajectory-level advantages. The policy is finally updated via a PPO-style clipped objective, promoting high-synergy actions while preserving stability.

We evaluate REWARDFLOW on four agentic benchmarks spanning text and visual modalities: Sokoban, ALFWorld, WebShop, and DeepResearch. Compared to strong RL baselines, REWARDFLOW achieves relative success-rate gains of 27.7% on Sokoban, 12.3% on ALFWorld, and 11.2% on DeepResearch (vs. Search-R1). It also provides more informative supervision, greater robustness under imperfect conditions, and minimal additional training overhead.

In summary, this work presents three main contributions:

- **State Graph Modeling:** We model agentic reasoning as state graphs that aggregate states and action-induced transitions across trajectories for analysis, revealing task-oriented properties (Sec. 3).
- **Policy Optimization with Estimated Process Rewards:** We propagate rewards in success states to intermediate states using state graphs, yielding objective, fine-grained supervision (Sec. 4).
- **Empirical Validation:** Extensive experiments across benchmarks and models show that RewardFlow consistently improves performance, robustness, and computational efficiency (Sec. 5).

2 PRELIMINARY

Problem formulation. Considering an agentic setting where an agent interacts with an environment to solve tasks with description $x \sim P(X)$. A trajectory is the sequence $\tau = (s_0, a_0, s_1, a_1, \ldots, a_{T-1}, s_T)$, where $s_t \in \mathcal{S}$ denotes the state (which includes the task description x and environmental observations) and $a_t \in \mathcal{A}$ the action chosen by the policy $\pi_\theta(a_t \mid s_t)$. In these environments, rewards are highly sparse: the reward $r(s_{t-1}, a_{t-1}, s_t) = 0$ for all non-terminal actions ($t < T$), and a non-zero reward is assigned only upon reaching the terminal state s_T. Because trajectories are often long and contain a mixture of helpful, irrelevant, and counterproductive actions, making accurate credit assignment from the sparse, trajectory-level signal remains a core challenge.

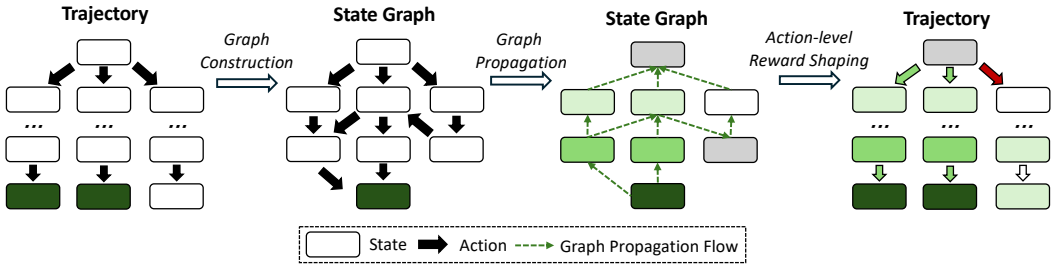


Figure 2: Overview of REWARDFLOW. Each rectangular node represents a state; color indicates reward level. Given sampled agentic trajectories (sequences of states and actions), REWARDFLOW estimates action-wise rewards via three steps: (1) **Graph Construction:** Build state graph by aggregating equivalent states. (2) **Graph Propagation:** Backpropagate rewards from success nodes. (3) **Action-level Reward Shaping:** Map propagated state rewards back to trajectories and compute action rewards as the reward difference (gain) between post-action and pre-action states.

Group sampling RL. Recent RL algorithms extend PPO (Schulman et al., 2017) with group-based advantage estimation, replacing the critic model for improved efficiency and scalability on verifiable-reward tasks. For a given task description x, the LLM samples a group of G trajectories $\{\tau^{(1)}, \tau^{(2)}, ..., \tau^{(G)}\}$. In a sparse reward setting, each completion $\tau^{(i)}$ receives a scalar reward $r^{(i)}$ in the terminal turn T_i, and the advantage of k-th token in t-th turn is computed via group normalization: $A^{(i)}_{t,k} = (r^{(i)} - \text{mean}(\{r^{(i)}\}_{i=1}^{G}))/\text{std}(\{r^{(i)}\}_{i=1}^{G}))$. Representative algorithms like GRPO (Shao et al., 2024) and RLOO (Ahmadian et al., 2024; Kool et al., 2019) assign the same terminal reward $r^{(i)}$ uniformly to every token in a trajectory. This weakens fine-grained credit assignment and hinders precise optimization in agentic tasks. In contrast, Group-in-Group Policy Optimization (GiGPO) (Feng et al., 2025b) estimates advantages at the state level by propagating rewards backward along trajectories, but does not explicitly exploit the objective topology among states.

3 THE STATE GRAPH OF AGENTIC REASONING

This section qualitatively evaluates the importance of individual states to task completion by analyzing their intrinsic relationships within trajectories. We (1) demonstrate the high repeatability of states across sampled trajectories to validate graph-based modeling, (2) construct state graphs by aggregating states from multiple trajectories and filtering out noisy actions, and (3) identify key structural properties in the resulting state graph that reveal each state's contribution to overall task success.

3.1 RETHINKING STATES IN AGENTIC REASONING

In agentic environments, states exhibit an intrinsic structure with multiple actions that transition to other states, similar to those in Markov chains, thereby forming an underlying conceptual state graph. From the perspective of an LLM, the sampled trajectories constitute partial observations (subgraphs) of this graph, leading to frequent re-occurrence of identical states across sampled trajectories.

$$\left| \bigcup_{i=1}^{G} \{ s_t \mid s_t \in \tau^{(i)} \} \right| \leq \sum_{i=1}^{G} T_i, \tag{1}$$

where $\bigcup_{i=1}^{G}\{\cdot\}$ denotes the set of unique states across G sampled trajectories, and T_i is the number of interaction steps (i.e., state visits) of trajectory $\tau^{(i)}$. This redundancy intensifies as the sampling size G increases. Empirical evidence in Fig. 3 supports this: the number of unique states is substantially smaller than the cumulative state visits, with distinct state-action transitions exhibiting similar compression. These findings motivate the construction of a unified, global state graph by aggregating information from multiple trajectories. Relative to modeling each trajectory independently, this consolidated representation offers a more comprehensive view of the underlying agentic task structure.

3.2 STATE GRAPH CONSTRUCTION

We construct the state graph leveraging the above insight. Firstly, we samples trajectories $\{\tau^{(i)}\}_{i=1}^{G}$ with a group size G through the LLM-driven policy π_θ: $\tau^{(i)} = \left(s_0, a_0^{(i)}, s_1^{(i)}, a_1^{(i)}, \ldots, a_{T_i-1}^{(i)}, s_{T_i}^{(i)}\right)$.

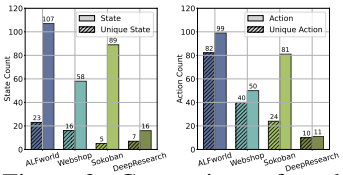


Figure 3: Comparison of total vs. unique states and actions in sampled trajectories.

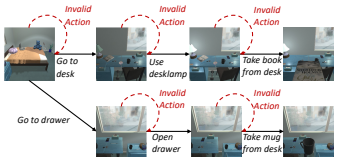


Figure 4: Invalid actions produce spurious edges in graphs, injecting unexpected noise.

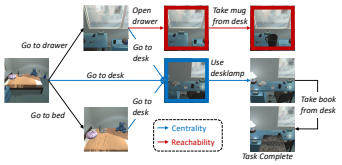


Figure 5: Example of constructed state graphs with **Reachability** and **Centrality**.

Prior to graph construction, we apply a normalization operation f for states and actions in trajectories to ensure that the resulting subgraph faithfully approximates the conceptual state graph.

- **States normalization:** [1] Raw environment states in trajectories frequently exhibit representational variability (e.g., identical underlying states may appear in different forms across observations). To address this, we canonicalize each observed state $s_t^{(i)}$ to produce a normalized representation: $\hat{s}_t^{(i)} = f(s_t^{(i)})$ that maps semantically equivalent states to a consistent canonical representation.
- **Pruning noisy transitions:** LLMs may propose syntactically plausible yet environment-invalid actions, adding spurious edges to the graph (see Fig. 4). To mitigate this problem, we construct canonical representations of actions $f(a_t^{(i)}) = \hat{a} \cdot \mathbb{1}(\textsc{valid}(s_t^{(i)}, a, s_{t+1}^{(i)}))$ that aggregate equivalent actions and remove hallucinated actions to ensure a substantial cleaner graph representation. [2]

We construct a state graph $\mathcal{G}_{\text{state}} = (\mathcal{S}, \mathcal{A}, \mathcal{T})$ from the sampled trajectories, where $\mathcal{S}$ is the set of distinct observed states, $\mathcal{A}$ the set of observed actions, and $\mathcal{T}$ the set of observed transitions. Nodes represent distinct states that appear in the environment, while directed edges represent actions observed in the sampled trajectories that transition from one state to another. The graph is constructed by taking the union of all states, actions, and transitions appearing in the collected set of trajectories:

$$\mathcal{S} = \bigcup_{i=1}^{G} \bigcup_{t=0}^{T_i} \{\hat{s}_t^{(i)}\}, \; \mathcal{A} = \bigcup_{i=1}^{G} \bigcup_{t=0}^{T_i-1} \{\hat{a}_t^{(i)}\}, \; \mathcal{T} = \bigcup_{i=1}^{G} \bigcup_{t=0}^{T_i-1} \{(\hat{s}_t^{(i)}, \hat{a}_t^{(i)}, \hat{s}_{t+1}^{(i)}) : P(\hat{s}' \mid \hat{s}, \hat{a}) > 0\}. \quad (2)$$

Each triple $(\hat{s}, \hat{a}, \hat{s}') \in \mathcal{T}$ encodes a directed, action-labeled edge from $\hat{s}$ to $\hat{s}'$, which is valid in the environment. Through construction, $\mathcal{G}_{\text{state}}$ approximates the underlying state graph of the environment; increasing the group size G expands its coverage of the underlying conceptual graph.

3.3 Revealing States' Properties via State Graphs

The global structure of the state graph $\mathcal{G}_{\text{state}}$ enables topological analysis of state dependencies, exposing prerequisites for success. We identify the following structural properties:

- **Reachability.** A state is reachable to success if there exists an observed path from it to a successful terminal state. In Fig. 5, states with no outgoing paths to a success (highlighted in red) are deemed unreachable. Moreover, proximity to success correlates with higher likelihood of reaching success.
- **Centrality.** In successful trajectories, states with high in-degree and out-degree often act as bottlenecks or pivotal junctures for problem-solving. For example, in Fig. 5, the state reached by the action "Go to desk" is critical: it has high out-degree, providing access to essential objects (desklamp and book) required for task success.

4 Learning with the State Graph

To address the sparse-reward problem in agentic RL, this section exploits the state graph to construct dense, meaningful per-action rewards for effective optimization. We propose: (1) shaping process rewards via graph propagation and projecting the propagated rewards onto trajectories, (2) estimating synergistic advantages that integrates action- and trajectory-level supervision, and (3) updating the policy via a clipped surrogate objective using the resulting advantages.

[1] Please see Appendix E.1 for implementation details.

[2] We denote trajectory states/actions by s, a and state-graph nodes/edges by $\hat{s}, \hat{a}$.

4.1 PROCESS REWARD SHAPING

The rewards of intermediate states are deduced by backward-propagating success over the state graph, projecting success-derived rewards onto states appearing in trajectories, and assigning each transition the difference. The resulting action-level rewards substantially improve credit assignment in RL.

Reward propagation. Let $\mathcal{S}_{\text{succ}} \subseteq \mathcal{S}$ be the set of success terminals. We perform multi-source inverse BFS from all $\hat{s}^\star \in \mathcal{S}_{\text{succ}}$, traversing $\mathcal{T}$ backwards, to obtain the shortest-hop distance to the nearest success node. Each node is assigned a propagated process reward based on this distance.

$$R(\hat{s}) \;=\; \gamma^{\,d(\hat{s})}, \gamma \in (0,1], \qquad d(\hat{s}) \;:=\; \min_{\hat{s}^\star \in \mathcal{S}_{\text{succ}}} \operatorname{dist}_{\text{hop}}(\hat{s} \rightsquigarrow \hat{s}^\star), \tag{3}$$

with the conventions $d(\hat{s}) = 0$ for $\hat{s} \in \mathcal{S}_{\text{succ}}$ and $d(\hat{s}) = \infty$ if no success is reachable from $\hat{s}$, thus $R(\hat{s}) = 1$ for success states, and $R(\hat{s})$ decreases monotonically as $\hat{s}$ gets farther from success. Intuitively, the propagated reward measures the promise of a state to solve the agentic task successfully, with states closer to success nodes receiving higher rewards.

Action-level rewards. Using the inverse preprocessing map $f^{-1} : \hat{s} \mapsto s, \hat{a} \mapsto a$, we project the propagated process rewards in state graphs back onto the raw states observed in trajectories: $R(s) := R(\hat{s}), \quad \forall\, s \in f^{-1}(\hat{s})$. For any transition (s_t, a_t, s_{t+1}) in the sampled trajectory, we define the action-shaped reward as the potential difference:

$$\tilde{r}(s_t, a_t) \;=\; R(s_{t+1}) - R(s_t). \tag{4}$$

Intuitively, $\tilde{R} > 0$ if a_t moves the agent closer to success, $\tilde{R} < 0$ if it moves it farther away, and $\tilde{R} = 0$ on equally-good plateaus. This yields dense, globally consistent feedback: every state receives a progress signal $R(s)$, and every observed action obtains immediate, interpretable credit $\tilde{r}$ reflecting its true contribution to task completion — substantially stabilizing and accelerating downstream RL.

4.2 ADVANTAGE ESTIMATION

We form advantages with reward signals at the action-level (local) and trajectory-level (global). We first collect action-reward pairs in trajectories through each $\hat{\boldsymbol{s}} \in \mathcal{S}$:

$$\text{Group}(\hat{\boldsymbol{s}}) := \bigcup_{i=1}^{G} \bigcup_{t=0}^{T_i - 1} \big\{(a_t^{(i)}, \tilde{r}_t^{(i)})\big\}, \text{ s.t. } (s_t^{(i)}, a_t^{(i)}, s_{t+1}^{(i)}) \in \mathcal{T}, f(s_t^{(i)}) = \hat{\boldsymbol{s}}. \tag{5}$$

Then the state-wise baseline and normalization factor are

$$\mu(\hat{\boldsymbol{s}}) := \operatorname{mean}\big\{\tilde{r}_t^{(j)} \mid (a_t^{(j)}, \tilde{r}_t^{(j)}) \in \text{Group}(\hat{\boldsymbol{s}})\big\}, \sigma(\hat{\boldsymbol{s}}) := F_{\text{norm}}\big\{\tilde{r}_t^{(j)} \mid (a_t^{(j)}, \tilde{r}_t^{(j)}) \in \text{Group}(\hat{\boldsymbol{s}})\big\}, \tag{6}$$

where μ measures the baseline of the policy given a state $\hat{\boldsymbol{s}}$, F_{norm} is a positive function (e.g., sample standard deviation with small positive ε). Then the advantage for an action $a_t^{(i)}$ taken in state $\hat{\boldsymbol{s}}$ is

$$A_{t,k}^{\text{action}}(\hat{\boldsymbol{s}}, a_t^{(i)}) \;=\; (\tilde{r}_t^{(i)} - \mu(\hat{\boldsymbol{s}}))/\sigma(\hat{\boldsymbol{s}}). \tag{7}$$

This produces an action-specific advantage signal that quantifies an action's performance relative to alternatives in the same state, supporting more precise and stable credit assignment in agentic RL.

Action-level advantages provide fine-grained local guidance but become uninformative (always zero) in single-action states where the group size is 1. To address this, we then incorporate trajectory-level advantages to construct synergistic advantages. Let $r^{(i)} \in \{0, 1\}$ indicate whether trajectory $\tau^{(i)}$ achieved success. The trajectory-level advantage is then defined as

$$A_{t,k}^{\text{traj}}(\tau^{(i)}) \;=\; (r^{(i)} - \overline{r})/\hat{\sigma}_r, \tag{8}$$

where $\overline{r}$ is the mean success rate and $\hat{\sigma}_r$ is the standard deviation of $\{r^{(i)}\}_{i=1}^{G}$ over G trajectories. We then combine both signals to obtain a robust final advantage:

$$A_{t,k}^{(i)} \;=\; \alpha_{\text{action}}\, A_{t,k}^{\text{action}} + \alpha_{\text{traj}}\, A_{t,k}^{\text{traj}}, \tag{9}$$

where $\alpha_{\text{action}}, \alpha_{\text{traj}} \geq 0$ are hyperparameters that control the relative strength of action-local versus trajectory-global supervision. This synergistic design ensures that the trajectory-level term provides guidance in low-data regimes (e.g., states with few or no alternative actions), while the action-level term delivers precise, local discrimination, resulting in dense and robust credit assignment.

4.3 POLICY UPDATE

We update the policy using a clipped surrogate objective (in the style of PPO) that favors actions with higher synergistic advantages. Given G trajectories $\{\tau^{(i)}\}_{i=1}^{G}$ sampled from the behavior policy $\pi_{\theta_{\text{old}}}$, where each trajectory $\tau^{(i)}$ contains T_i transitions, the per-step importance ratio is defined as $\rho_{i,t} = \pi_\theta(o_{t,k}^{(i)} \mid s_{t,k}^{(i)}, o_{t,<k}^{(i)}) / \pi_{\theta_{\text{old}}}(o_{t,k}^{(i)} \mid s_{t,k}^{(i)}, o_{t,<k}^{(i)})$. The objective of REWARDFLOW is defined as

$$J_{\text{REWARDFLOW}}(\theta) = \mathbb{E}_{\substack{s_0 \sim \mathcal{S}, \\ \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}}} \Bigg[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{T_i} \sum_{t=1}^{T_i} \frac{1}{|o_t^{(i)}|} \sum_{k=1}^{|o_t^{(i)}|} \Bigg(\min \Big[\rho_{t,k}^{(i)} A_{t,k}^{(i)}, \text{clip}(\rho_{t,k}^{(i)}, 1-\varepsilon, 1+\varepsilon) A_{t,k}^{(i)} \Big] - \beta \mathbb{D}_{\text{KL}}(\pi_\theta || \pi_{\text{ref}}) \Bigg) \Bigg], \tag{10}$$

with clipping parameter $\epsilon > 0$. Together, these components steer the policy toward reasoning with actions that are both locally superior and globally effective for task completion.

5 EXPERIMENTS

5.1 EXPERIMENT SETUP

Datasets. We evaluate REWARDFLOW on the text-based environment: **ALFWorld** (Shridhar et al., 2021), **WebShop** (Yao et al., 2022), and **DeepResearch** (Jin et al., 2025) tasks and the visual environment **Sokoban** (Schrader, 2018). Brief descriptions of these environments are provided below.

- **ALFWorld:** A household task with six type: *Pick & Place* (Pick), *Examine in Light* (Look), *Clean & Place* (Clean), *Heat & Place* (Heat), *Cool & Place* (Cool), and *Pick Two & Place* (Pick2).
- **WebShop:** Large-scale, text-based web-shopping environment where agents follow natural-language instructions to shop using text commands (search, click, etc).
- **Sokoban:** A classic spatial puzzle where agents push boxes into goal locations on randomly generated 6×6 grids, requiring understanding spatial constraints and long-horizon planning.
- **DeepResearch:** The knowledge-intensive QA with retrieval systems. We evaluate on single-hop tasks: NarrativeQA (NQ) (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023), and multi-hop tasks: HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (2Wiki) (Ho et al., 2020), Musique (Trivedi et al., 2022), and Bamboogle (Press et al., 2023)

Baselines. We compare the performance of REWARDFLOW with three competitive RL training baselines on ALFWorld, WebShop, and Sokoban: RLOO (Ahmadian et al., 2024), GRPO (Shao et al., 2024), that have proven effective for language reasoning. We also include GiGPO (Feng et al., 2025b), an agent-tailored RL with state-wise advantage estimation via terminal-to-initial reward propagation along trajectories. On DeepResearch, we follow the experimental setting with that of GiGPO and compare REWARDFLOW against: R1-Instruct, Search-R1 (Jin et al., 2025), ZeroSearch (Sun et al., 2025), StepSearch (Wang et al., 2025b), and GiGPO.

Training settings. We train Qwen2.5-VL-3B/7B-Instruct on Sokoban and Qwen2.5-1.5B/3B/7B-Instruct on ALFWorld and WebShop. For DeepResearch, we train Qwen2.5-3B/7B-Instruct. State embeddings are obtained using E5 (Wang et al., 2022) for retrieval and similarity measurement. Validation is performed on a randomly sampled set of 128 tasks, reporting the checkpoint's performance on this validation split. Full training settings and hyperparameters are provided in Appendix G.1.

5.2 MAIN RESULTS

Experimental results appear in Tabs. 1 and 2. REWARDFLOW outperforms prior RL methods across model sizes from 1.5B to 7B on all benchmarks. These gains stem primarily from REWARDFLOW's accurate estimation of dense process rewards. Key findings per benchmark are summarized below.

- **ALFWorld:** REWARDFLOW achieves the highest overall success rate (89.8% at 7B, +7.0% over GiGPO) and ranks first or second in nearly all subtasks across all model sizes. It establishes SOTA results in Pick (100% at 7B, +7.7% over GRPO) and Clean (100% at 7B, +4.0% over GiGPO).

Table 1: Performance comparison of REWARDFLOW and baselines. Results show success rates (%) before and after RL training. For ALFWorld, we show per-subtask success rates and the overall average. For WebShop and Sokoban, we report both score and success rate.

Type	Method	ALFWorld							WebShop		Sokoban	
		Pick	Look	Clean	Heat	Cool	Pick2	All	Score	Succ.	Score	Succ.
Qwen2.5-1.5B-Instruct												
Prompting	Base	5.9	5.5	3.3	9.7	4.2	0	4.1	23.1	5.2	-	-
RL Training	RLOO	56.2	46.7	62.5	50.0	43.5	27.8	49.2	**80.9**	54.7	-	-
RL Training	GRPO	62.9	53.3	50.0	40.0	45.8	38.1	50.0	73.7	42.2	-	-
RL Training	GiGPO	59.4	46.7	62.5	44.4	43.5	**56.3**	53.1	75.4	55.5	-	-
RL Training	REWARDFLOW	**77.4**	**64.3**	**84.0**	**62.5**	**86.4**	25.0	**68.8**	78.3	**60.9**	-	-
Qwen2.5-(VL)-3B-Instruct												
Prompting	Base	36.4	42.9	9.1	7.1	5.3	4.5	16.4	8.0	1.6	0.5	14.1
RL Training	RLOO	78.1	46.7	75.0	31.3	43.5	33.3	55.5	75.2	59.4	1.0	22.7
RL Training	GRPO	79.8	57.1	75.8	21.4	31.6	36.4	56.2	69.5	53.9	1.3	26.6
RL Training	GiGPO	82.1	50.0	76.9	**53.3**	60.9	50.0	64.8	80	59.4	1.2	21.9
RL Training	REWARDFLOW	**94.6**	**70.0**	**90.0**	36.4	**70.0**	**70.0**	**78.9**	**81.8**	**60.9**	**2.2**	**49.2**
Qwen2.5-(VL)-7B-Instruct												
Prompting	Base	33.3	13.3	10.7	0	4.3	0	10.9	26.4	7.8	0.9	18.8
RL Training	RLOO	90.0	**85.7**	88.0	50.0	**85.7**	37.5	75.0	84.2	72.7	1.0	21.9
RL Training	GRPO	92.3	53.3	90.5	77.8	69.6	47.6	75.0	70.3	54.0	1.0	23.4
RL Training	GiGPO	84.6	80.0	96.0	71.4	73.9	84.0	82.8	**88.4**	72.7	1.4	34.4
RL Training	REWARDFLOW	**100**	78.6	**100**	**81.3**	81.8	**85.0**	**89.8**	84.4	**73.4**	**3.0**	**62.4**

Table 2: Performance comparison of REWARDFLOW on RewardFlow performance on DeepResearch QA benchmarks (trained on NarrativeQA + HotpotQA following GiGPO (Feng et al., 2025b). Reported as average accuracy (%). †in-distribution, * out-of-distribution.

Type	Method	Single-hop QA			Multi-hop QA				Avg.
		NQ†	TriviaQA*	PopQA*	HotpotQA†	2Wiki*	Musique*	Bamboogle*	
Qwen2.5-3B-Instruct									
RL Training	R1-Instruct	27.0	53.7	19.9	23.7	29.2	7.2	29.3	27.1
RL Training	Search-R1	34.1	54.5	37.8	32.4	31.9	10.3	26.4	32.5
RL Training	ZeroSearch	41.4	57.4	**44.8**	27.4	30.0	9.8	11.1	31.7
RL Training	StepSearch	-	-	-	34.5	32.0	**17.4**	34.4	-
RL Training	GiGPO	42.0	59.5	42.4	36.9	37.0	12.6	**64.1**	42.1
RL Training	REWARDFLOW	**44.4**	**60.9**	44.1	**40.3**	**41.2**	15.2	63.7	**44.3**
Qwen2.5-7B-Instruct									
RL Training	R1-Instruct	21.0	44.9	17.1	20.8	27.5	6.0	19.2	22.4
RL Training	Search-R1	39.3	61.0	39.7	37.0	40.1	14.6	36.8	38.5
RL Training	ZeroSearch	43.6	61.8	**51.5**	34.6	35.2	18.4	27.8	39.1
RL Training	StepSearch	-	-	-	38.6	36.6	**22.6**	40.0	-
RL Training	GiGPO	46.4	64.7	46.1	41.6	43.6	18.9	68.9	47.2
RL Training	REWARDFLOW	**47.4**	**65.2**	48.1	**44.7**	**47.1**	19.9	**71.4**	**49.1**

Notably large gains on smaller models (1.5B and 3B) highlight how REWARDFLOW's fine-grained rewards reduce optimization difficulty and enhance agentic reasoning in lower-capacity models.

- **WebShop:** Similarly, REWARDFLOW frequently achieves the best or second-best scores and success rates across all model sizes, with particularly strong gains for the 1.5B model (+5.4% over GiGPO), underscoring its effectiveness to improve agentic reasoning for weaker models.
- **Sokoban:** REWARDFLOW achieves the largest gain, reaching 62.4% success at 7B — a 28.0% improvement over the previous best (GiGPO at 34.4%). This demonstrates strong generalization beyond text-only settings and highlights its promise for visual agentic reasoning tasks.
- **DeepResearch:** While states remain stochasticity, REWARDFLOW delivers the best average performance in both 3B/7B models (44.3% at 3B, +2.2% over GiGPO; 49.1% at 7B, +1.9% over GiGPO). Across subtasks, it consistently ranks first or second in results compared to baselines, indicating its applicability to broad agentic tasks with appropriate state aggregation strategies.

5.3 ROBUSTNESS UNDER IMPERFECT SCENARIOS

We test the robustness of REWARDFLOW under imperfect scenarios: (1) reasoning in out-of-distribution (OOD) tasks and (2) resiliency with insufficient exploration.

Out-of-distribution (OOD) environments. We evaluate the OOD generalization of REWARDFLOW on ALFWorld and DeepResearch (see Tabs. 2 and 3). In ALFWorld, prior methods suffer substantial performance degradation under OOD conditions, whereas REWARDFLOW exhibits strong robustness, with only a 3.9% drop for the 3B model and 2.4% for the 1.5B model. On DeepResearch, REWARDFLOW achieves strong unseen-domain performance: 60.9% on TriviaQA (+1.4% over GiGPO) and 41.2% on 2Wiki (+4.2% over GiGPO) at the 3B scale, despite never encountering these tasks during training. These indicate that REWARDFLOW genuinely enhances general agentic reasoning capabilities rather than overfitting to specific samples.

Table 3: OOD evaluation on ALFWorld. The agent must solve household tasks with familiar objects from training, but in entirely novel environments (different rooms, layouts, and furniture).

Method	ALFWorld						
	Pick	Look	Clean	Heat	Cool	Pick2	All
Qwen2.5-1.5B-Instruct							
Base	8.0	5.3	2.8	10.0	7.1	0	5.5
RLOO	50.0	44.4	36.7	28.6	40.7	29.4	38.3
GRPO	52.0	26.3	33.3	60.0	21.4	21.4	37.5
GiGPO	50.0	55.6	40.0	38.1	63.0	23.5	45.3
REWARDFLOW	**64.0**	**78.9**	**61.1**	**70.0**	**71.4**	**57.1**	**66.4**
Qwen2.5-3B-Instruct							
Base	56.0	15.8	5.6	10.0	7.1	0	17.2
RLOO	72.0	36.8	44.4	35.0	57.1	35.7	47.7
GRPO	70.8	33.3	33.3	28.6	51.9	35.3	43.7
GiGPO	66.7	**55.6**	36.7	28.5	48.1	41.2	45.3
REWARDFLOW	**79.2**	**55.6**	**76.7**	**57.1**	**92.6**	**70.6**	**75.0**
Qwen2.5-7B-Instruct							
Base	28.0	26.3	5.6	20.0	7.1	0	14.8
RLOO	79.2	77.8	66.7	47.6	85.2	64.7	70.3
GRPO	72.0	63.2	72.2	**65.0**	**100.0**	14.3	66.4
GiGPO	**83.3**	63.2	**75.0**	57.1	76.9	72.2	71.9
REWARDFLOW	75.0	**100.0**	66.7	61.9	**100.0**	**82.4**	**78.9**

Table 4: Ablation on number of rollouts per training step using Qwen2.5-1.5B-Instruct in ALFWorld. Success rate (%) reported; RewardFlow includes average graph statistics.

Method	Rollouts	Avg. Nodes	Avg. Edges	Success Rate (%)
GiGPO	4	-	-	28.9
	6	-	-	51.6
	8	-	-	53.1
REWARDFLOW	4	17.1	28.8	**42.2**
	6	22.9	41.3	**53.1**
	8	28.8	55.9	**68.8**

Table 5: Performance of REWARDFLOW with or without each normalization operation on ALFWorld using Qwen2.5-3B-Instruct.

Method	Success Rate (%)
Base	16.4
REWARDFLOW (w/o state normalization)	53.9
REWARDFLOW (w/o noisy transition pruning)	60.2
REWARDFLOW	**78.9**

Insufficient exploration We evaluate robustness under poor exploration by comparing REWARDFLOW with GiGPO under limited sampling budgets. In REWARDFLOW, fewer trajectories provide sparser graphs (fewer nodes and edges), which can impair the induction of agentic environment structure and compromise the reliability of process reward estimation. In Tab. 4, REWARDFLOW consistently outperforms GiGPO with a +13.3% improvement even with only 4 trajectories. Increasing the sampling budgets yields disproportionately larger gains for REWARDFLOW, further widening its advantage over baselines. This indicates REWARDFLOW effectively extracts reliable process rewards from minimal data, demonstrating strong robustness to poor exploration (see Appendix F.1).

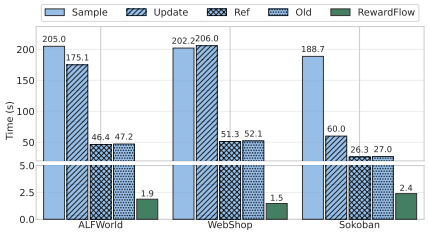


Figure 6: Time breakdown per training step in REWARDFLOW. Blue: shared RL stages (trajectory sampling, probability computation, policy update). Green: REWARDFLOW stages (graph construction + process reward shaping).

5.4 COMPUTATIONAL EFFICIENCY

We compare the runtime of REWARDFLOW's core components (state graph construction and backward reward propagation) against other major parts of the RL training. Figure 6 shows that graph construction and process reward shaping take at most 2.39 seconds across the three evaluated environments — a small and highly efficient overhead compared to the much more expensive phases of trajectory collection and policy gradient updates. Detailed timing results are provided in Appendix G.2.

5.5 ABLATION STUDY

Ablation studies (Tab. 5) show that both state normalization and noisy transition pruning are critical to REWARDFLOW. Removing state normalization reduces performance by 25%, while excluding noisy transition pruning causes an 18.7% drop, demonstrating that these components are essential for high-quality graph construction, clean reward propagation, and accurate advantage estimation in partially observable settings. Details are provided in Appendices E.1 and E.2.

6 CONCLUSIONS

We introduce REWARDFLOW, a process reward modeling framework that estimates per-state contributions in agentic trajectories. It uncovers topological relationships among states and applies graph-based propagation to distribute terminal rewards, producing principled process evaluation. Combining action-level rewards with trajectory-level supervision yields substantial performance improvements in both text-based and visual domains. REWARDFLOW demonstrates strong robustness under imperfect conditions and exceptional training efficiency.

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. *arXiv preprint arXiv:2402.14740*, 2024.

Xiaocong Chen, Siyu Wang, Julian McAuley, Dietmar Jannach, and Lina Yao. On the opportunities and challenges of offline reinforcement learning for recommender systems. *ACM TOIS*, 2024.

Nicola De Cao and Thomas Kipf. Molgan: An implicit generative model for small molecular graphs. *arXiv preprint arXiv:1805.11973*, 2018.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, et al. Agentic reinforced policy optimization. *arXiv preprint arXiv:2507.19849*, 2025.

Jiazhan Feng, Shijue Huang, Xingwei Qu, Ge Zhang, Yujia Qin, Baoquan Zhong, Chengquan Jiang, Jinxin Chi, and Wanjun Zhong. Retool: Reinforcement learning for strategic tool use in llms. *arXiv preprint arXiv:2504.11536*, 2025a.

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

Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. Navigating the digital world as humans do: Universal visual grounding for GUI agents. In *ICLR*, 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.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*, 2020.

Jian Hu. Reinforce++: A simple and efficient approach for aligning large language models. *arXiv preprint arXiv:2501.03262*, 2025.

Dongfu Jiang, Yi Lu, Zhuofeng Li, Zhiheng Lyu, Ping Nie, Haozhe Wang, Alex Su, Hui Chen, Kai Zou, Chao Du, et al. Verltool: Towards holistic agentic reinforcement learning with tool use. *arXiv preprint arXiv:2509.01055*, 2025.

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

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*, 2017.

Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

Wouter Kool, Herke van Hoof, and Max Welling. Buy 4 REINFORCE samples, get a baseline for free!, 2019.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *ACL*, 2019.

Xuefeng Li, Haoyang Zou, and Pengfei Liu. Torl: Scaling tool-integrated rl. *arXiv preprint arXiv:2503.23383*, 2025.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *ICLR*, 2023.

Yen-Ting Lin, Di Jin, Tengyu Xu, Tianhao Wu, Sainbayar Sukhbaatar, Chen Zhu, Yun He, Yun-Nung Chen, Jason E Weston, Yuandong Tian, et al. Step-kto: Optimizing mathematical reasoning through stepwise binary feedback. In *Proceedings of The 3rd Workshop on Mathematical Natural Language Processing (MathNLP 2025)*, 2025.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *ACL*, 2023.

Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. In *NeurIPS*, 2024.

E. F. Moore. The shortest path through a maze. In *Proc. International Symposium on the Theory of Switching, Part II*. harvard, 1959.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *EMNLP*, 2023.

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

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, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *CVPR*, 2020.

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

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

Hao Sun, Zile Qiao, Jiayan Guo, Xuanbo Fan, Yingyan Hou, Yong Jiang, Pengjun Xie, Yan Zhang, Fei Huang, and Jingren Zhou. Zerosearch: Incentivize the search capability of llms without searching. *arXiv preprint arXiv:2505.04588*, 2025.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *IROS*, 2012.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single-hop question composition. *ACL*, 2022.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *nature*, 575(7782):350–354, 2019.

Hanlin Wang, Chak Tou Leong, Jiashuo Wang, Jian Wang, and Wenjie Li. Spa-rl: Reinforcing llm agents via stepwise progress attribution. *arXiv preprint arXiv:2505.20732*, 2025a.

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

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

Yuxi Xie, Anirudh Goyal, Wenyue Zheng, Min-Yen Kan, Timothy P Lillicrap, Kenji Kawaguchi, and Michael Shieh. Monte carlo tree search boosts reasoning via iterative preference learning. *arXiv preprint arXiv:2405.00451*, 2024.

Fengli Xu, Qianyue Hao, Zefang Zong, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, Tianjian Ouyang, Fanjin Meng, et al. Towards large reasoning models: A survey of reinforced reasoning with large language models. *arXiv preprint arXiv:2501.09686*, 2025.

Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Xiaosen Zheng, Zejun Ma, and Bo An. Simpletir: End-to-end reinforcement learning for multi-turn tool-integrated reasoning. *arXiv preprint arXiv:2509.02479*, 2025.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*, 2018.

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

Jiaxuan You, Bowen Liu, Zhitao Ying, Vijay Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. *NeurIPS*, 2018.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, YuYue, 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 *NeurIPS*, 2025.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. ReST-MCTS*: LLM self-training via process reward guided tree search. In *NeurIPS*, 2024.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. *arXiv preprint arXiv:2504.03160*, 2025.

Aojun Zhou, Ke Wang, Zimu Lu, Weikang Shi, Sichun Luo, Zipeng Qin, Shaoqing Lu, Anya Jia, Linqi Song, Mingjie Zhan, et al. Solving challenging math word problems using gpt-4 code interpreter with code-based self-verification. In *ICLR*, 2024.

APPENDIX

A Ethics Statement

This study does not involve human subjects, dataset releases, potentially harmful insights, applications, conflicts of interest, external sponsorship, discrimination, bias, fairness concerns, privacy or security issues, legal compliance issues, or research integrity concerns.

B LLM Usage Disclosure

This manuscript was prepared with the assistance of LLMs, which were used for refining content and ensuring grammatical accuracy. The authors take full responsibility for the entire content of the manuscript and confirm that no LLM is listed as an author.

C Impact Statement

The primary objective of this research is to investigate limitations in the active reasoning capabilities of large language models, contributing to the advancement of trustworthy language model reasoning. We emphasize that any references to violence in the datasets are purely fictional without involving any practical details. This work neither represents nor endorses real-world violence. Furthermore, this research was conducted independently, free from conflicts of interest or external sponsorship. The study adheres to ethical research principles, addressing considerations of discrimination, bias, fairness, privacy, security, and legal compliance while maintaining research integrity.

D Related Work

RL for LLM reasoning. Reinforcement learning (RL) (Kaelbling et al., 1996) optimizes policies to maximize rewards and has recently enhanced LLM reasoning (Xu et al., 2025). Approaches fall into two paradigms: off-policy methods train from data collected by other policies—e.g., DPO (Rafailov et al., 2023), often combined with MCTS (Zhang et al., 2024; Xie et al., 2024), though paired preferences are costly; newer variants reduce this burden via single-instance or stepwise feedback (KTO (Zhou et al., 2024), Step-KTO (Lin et al., 2025)) and by removing the reference model (SimPO (Meng et al., 2024)). On-policy methods (e.g., PPO (Schulman et al., 2017), Reinforce++ (Hu, 2025)) optimize using data from the current policy, typically with an auxiliary reward model that increases compute and risks reward hacking (Guo et al., 2025); GRPO (Shao et al., 2024), and RLOO (Ahmadian et al., 2024; Kool et al., 2019) mitigates this via group-based sampling and relative advantage estimation, with extensions such as DAPO (Yu et al., 2025) and Dr. GRPO (Liu et al., 2025) advancing optimization for complex reasoning (Guo et al., 2025).

RL for agentic scenarios. Recent advancements in RL have extended its application to train LLMs in agentic scenarios, enabling agents to interact with tools and external environments to tackle complex problems. Frameworks such as Search-R1 (Jin et al., 2025), R1-Searcher (Song et al., 2025), and DeepResearcher (Zheng et al., 2025) integrate LLMs with retrieval tools, including document vector databases and search engines, using GRPO to enhance both reasoning and tool usage capabilities. For computationally intensive tasks like mathematical reasoning, frameworks such as ReTool (Feng et al., 2025a), ToRL (Li et al., 2025), and Verl-Tool (Jiang et al., 2025) leverage Python environments to optimize agents' abilities to solve problems through code-based complex calculations. Despite their promise, these frameworks struggle with long-horizon reasoning. To address this, ARPO (Dong et al., 2025) employs an entropy-guided strategy during the rollout process to encourage exploration in high-entropy states, leading to improved trajectory collection and optimization. Similarly, SimpleTIR (Xue et al., 2025) identifies and filters out trajectories containing "void turn", which can destabilize multi-turn agentic training, ensuring more stable optimization.

E Further Method Details

E.1 State Preprocessing

The state preprocessing in RewardFlow effectively handles stochasticity and ambiguity **by aggregating semantically equivalent states using embeddings or enriching the information of states.** Here, we outline the specific challenges posed by stochasticity and ambiguity in the ALFWorld and DeepResearch environments, followed by the targeted state-preprocessing strategies we employed (described in Section 3) and applied consistently in both settings.

For ALFWorld:

- Challenge: Although raw text observations are usually distinct, ALFWorld is partially observable and often omits critical object property changes. For example, after the agent cleans an apple, the observation may still read "You are carrying an apple" without indicating that it is now cleaned, creating ambiguous states that can degrade the quality of the state graph.
- Solution: We enrich the raw observation by automatically detecting transformative actions (e.g., clean, heat, cool) and appending the resulting property changes to the text (e.g., "You are carrying an apple [cleaned]"). This ensures that states before and after such actions are distinguishable, yielding accurate node representations and reliable aggregation.

For DeepResearch:

- Challenge: DeepResearch is inherently highly stochastic and ambiguous. Semantically similar search queries frequently return substantially different document sets, causing (1) high stochasticity (from similar actions to divergent raw states) and (2) ambiguity (functionally equivalent research progress represented by distinct observations). This leads to severe graph fragmentation if raw states are used directly.
- Solution: We perform embedding-based node aggregation using a sentence transformer. States s and s' are merged into a single super-node if their cosine similarity exceeds a threshold (default 0.9). This eliminates near-duplicate nodes, mitigates fragmentation from query paraphrasing, preserves meaningful transitions, and produces a compact, semantically coherent graph that supports stable reward propagation via BFS.

$$f(s) = \begin{cases} \text{cluster representative}(s) & \text{if } s \text{ is semantically equivalent to others,} \\ s \oplus \text{critical features} & \text{if } s \text{ is ambiguous,} \\ s & \text{otherwise.} \end{cases}$$

E.2 Invalid Actions Filtering

Invalid actions arise from the LLM policy's outputs during interaction with the environment, but they are distinct in nature. Invalid actions include cases where (1) the policy model outputs responses from which no valid action can be extracted (e.g., malformed or nonsensical text that fails parsing), or (2) the policy outputs an action that is not among the admissible actions in the current environment state (e.g., attempting an unavailable command). These definitions align with the VALID predicate, which checks for successful execution and state change.

Filtering invalid actions ensures a cleaner state graph, improving the accuracy of reward propagation. In practice, invalid actions can lead to erroneous states that do not exist in the true agentic environment; for instance, in ALFWorld, they trigger a fallback state like "Nothing happens," which introduces noise and distorts the graph structure by creating inaccurate nodes or edges. Our ablation study in Tab. 5 confirms that filtering invalid actions improves overall performance by reducing noise in credit assignment.

Given invalid actions, invalid states occur when the agent executes an invalid action, prompting the environment to return feedback that does not reflect a genuine state update. In many agentic environments, such as ALFWorld, the agent may attempt actions that are not feasible given the current context, resulting in responses like "Nothing happens." This feedback is erroneously interpreted as a new state during rollout collection, even though it carries no meaningful environmental information. For instance, consider the following exemplar trajectory from ALFWorld:

An example of invalid actions occured in ALFWorld

STEP: 1
STATE: -= Welcome to TextWorld, ALFRED! =-

You are in the middle of a room. Looking quickly around you, you see a cabinet 10, a cabinet 9, a cabinet 8, a cabinet 7, a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 1, a diningtable 1, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1.

Your task is to: put a cool apple in microwave.
ACTION: go to cabinet 1
—
STEP: 2
STATE: You arrive at cabinet 1. The cabinet 1 is closed.
ACTION: go to cabinet 1 [INVALID ACTION]
—
STEP: 3
STATE: Nothing happens. [INVALID STATE]
ACTION: open cabinet 1
—
STEP: 4
STATE: You open the cabinet 1. The cabinet 1 is open. In it, you see nothing.

If left unfiltered, this string "Nothing happens" would be treated as an ordinary node in the state graph, creating noisy nodes and edges that distort the propagated rewards. We remove these states with this strategy to ensure accurate graph construction and reward propagation. The ablation study of Tab. 5 supports the efficacy of this strategy.

E.3 Prompt Templates

In this section, we provide the prompt used in RL training to guide the model to do basic agentic reasoning. The prompt with no history denotes the initial prompt, while the prompt with history denotes the prompt in intermediate states.

Prompt of ALFWorld without history

You are an expert agent operating in the ALFRED Embodied Environment.
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.

Prompt of ALFWorld with history

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.

Prompt of WebShop without history

You are an expert autonomous agent operating in the WebShop e-commerce environment.
Your task is to: {task_description}.
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.

Prompt of WebShop with history

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.

Prompt of Sokoban

You are an expert agent operating in the Sokoban environment. Your goal is to push all the boxes onto the target spots. Once all boxes are on the targets, you win!

Rules
You can only push boxes. You can't pull them, so plan ahead to avoid getting stuck.
You can't walk through or push boxes into walls.
To avoid traps, do not push boxes into corners or against walls where they can't be moved again.

Visual Elements in the Image:
Character: A small, green alien-like figure with two antennae and black eyes. It represents you.
Box: A yellow crate marked with an orange "X" across its front. It is the box you need to push.
Target: A black tile outlined in red, with a small red diamond shape in the center. It marks the destination where a box should be pushed.

Current Step
Your current observation is shown in the image: <image>
Your admissible actions are ["up", "down", "left", "right"].

Now it's your turn to make a move (choose ONE action only for the current step).
You should first reason step-by-step about the current situation — observe the positions of boxes and targets, plan a path to push a box toward a target, and avoid traps like corners or walls. 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.

Prompt of DeepResearch without history

You are an expert agent tasked with answering the given question step-by-step.
Your question: {task_description}

Now it's your turn to respond for the current step.
You should first conduct reasoning process. This process MUST be enclosed within <think> </think> tags.
After completing your reasoning, choose only one of the following actions (do not perform both):
(1) If you find you lack some knowledge, you can call a search engine to get more external information using format: <search> your query </search>.
(2) If you have enough knowledge to answer the question confidently, provide your final answer within <answer> </answer> tags, without detailed illustrations. For example, <answer>Beijing</answer>.

Prompt of DeepResearch with history

You are an expert agent tasked with answering the given question step-by-step.
Your question: {task_description}

Prior to this step, you have already taken {step_count} step(s). Below is the interaction history where <search> </search> wrapped your past search queries and <information> </information> wrapped the corresponding search results returned by the external search engine. History:
{memory_context}

Now it's your turn to respond for the current step.
You should first conduct reasoning process. This process MUST be enclosed within <think> </think> tags.
After completing your reasoning, choose only one of the following actions (do not perform both):
(1) If you find you lack some knowledge, you can call a search engine to get more external information using format: <search> your query </search>.
(2) If you have enough knowledge to answer the question confidently, provide your final answer within <answer> </answer> tags, without detailed illustrations. For example, <answer>Beijing</answer>.

F Further Discussion

F.1 Does RewardFlow Rely on the High-quality Exploration?

RewardFlow does not perform exhaustive exploration and scales to massive state/action spaces. RewardFlow derives states, actions, and transitions directly from RL rollout sampling, and no exhaustive exploration is required. The induced graph is a compact subset of visited trajectories, and reward propagation operates solely on this subgraph. In large agentic environments, RewardFlow remains applicable as long as the LLM agent can sample at least one successful trajectory per batch. This avoids "exhaustive visitation" and ensures tractability, even in continuous or high-dimensional spaces, by clustering states via embeddings or hashing (as in our stochastic handling below) without assuming pure discreteness.

The accuracy of the state graph that is critical for reward modeling is not obviously affected by the poor exploration. RewardFlow ensures accurate state graph construction by building graphs solely from actual environment interactions (states and actions), as induced from LLM-generated rollouts. This guarantees that the graph is a faithful topological representation of visited trajectories, without hallucinations or invalid edges. As long as at least one successful trajectory is sampled in a group, propagation methods like BFS reliably assign state-wise rewards reflecting task-centric metrics, such as the minimum actions needed to reach success from each state (shortest path in the graph). Poor exploration (e.g., low diversity) merely results in a sparser but still accurate graph. modeling a subset of the environment without introducing errors or misguiding propagation. This contrasts with trajectory-level methods, where sparse rewards can dilute credit assignment across entire paths. Our filtering of invalid/self-examination actions further enhances reliability by removing noise, ensuring propagation focuses on meaningful transitions. The ablation study in Tab. 4 supports this, where RewardFlow consistently outperforms GiGPO, even reducing rollouts number, indicating that RewardFlow is robust to poor exploration.

We further measure the entropy of the policy during different training steps and investigate how the entropy affects graph density. We measure policy entropy as $H(p) = -\sum_{i=1}^{|V|} p(x_i) \log p(x_i)$, where $|V|$ denotes the length of the policy's vocabulary and x_i denotes each token in the vocabulary. To study the interplay between entropy and graph structure, we evaluate three checkpoints of the Qwen-2.5-1.5B-Instruct model during RewardFlow training on ALFWorld: step 0 (initial supervised model), step 50, and step 100. For each checkpoint, we collect K = 8 rollouts per task using group sampling, compute the entropy, and record graph statistics along with validation performance.

Table 6: Training progress of Qwen-2.5-1.5B-Instruct on the target task. Entropy, invalid rates, graph size, and success rate across training steps.

Training Step	Entropy	Invalid State Rate (%)	Invalid Action Rate (%)	Avg. Node	Avg. Edge	Success Rate (%)
0	1.099	33.1	41.0	24.1	46.9	4.1
50	0.555	0.3	0	33.1	55.9	44.5
100	0.295	0.1	0	19.9	30.6	68.8

At step 0, the policy has very high entropy (1.099) and generates many invalid actions, which are subsequently pruned. As a result, even though the policy is highly exploratory, a large fraction of transitions do not contribute valid edges, yielding only moderately dense graphs and poor reward signal propagation. At step 50, entropy has decreased significantly (0.555), the model almost never produces invalid actions, and exploration remains sufficient to discover diverse valid paths. This produces the densest state graphs (33.1 nodes and 55.9 edges on average), maximizing the coverage of the refined graph and enabling the most effective BFS-based reward backpropagation, which explains the rapid performance improvement at this stage. At step 100, entropy is lowest (0.295), and the policy has converged toward near-deterministic optimal behavior. It now focuses almost exclusively on high-reward paths to success states, resulting in the sparsest graphs (19.9 nodes and 30.6 edges on average). Despite having the fewest nodes and edges, these compact graphs are highly efficient: almost every observed transition reduces the shortest-hop distance to a success state, making reward shaping extremely precise and leading to the highest validation success rate (65.6%). These results reveal a clear and insightful trend: excessively high entropy harms graph density due to invalid actions, moderate entropy maximizes graph coverage and reward densification

during learning, and low entropy ultimately yields minimal yet highly informative graphs that support optimal performance.

Furthermore, the state-graph visualization case studies in ALFWorld (Appendix G.6) demonstrate that, even with a single successful sampling rollout, the constructed state graph accurately captures the topological relationships among states, enabling precise reward assignment to intermediate states. With additional sampling rollouts, the graph modeling is further refined, yielding even more reliable rewards.

F.2 Comparing RewardFlow with LLM-based PRM Methods

Existing PRMs and preference learning approaches are largely confined to static, single-turn reasoning tasks (e.g., math problems (Lightman et al., 2023)), where paired preferences or step-level scores can be obtained relatively easily. In contrast, long-horizon agentic tasks involve dynamic state transitions and environment feedback, making human annotation of intermediate preferences prohibitively costly (20-40 steps per episode in ALFWorld). Recent agentic RL methods (e.g., ARPO (Dong et al., 2025), SimpleTIR (Xue et al., 2025)) rely on trajectory- or group-level supervision and do not employ stepwise PRMs. GiGPO (Feng et al., 2025b), the closest related work, combines state- and trajectory-level advantages but ignores topological relationships across states. This gap underscores RewardFlow's novelty: it delivers dense, topology-aware supervision without requiring human annotations or auxiliary reward models. We evaluate RewardFlow against two baselines in Tab. 7: state-wise GRPO using Qwen2.5-7B-Instruct as a process reward model (PRM) for per-action quality scoring, and standard PPO with its implicit token-wise reward model. RewardFlow substantially outperforms both in efficacy and efficiency: it achieves +22.6% over PPO and +34.3% over GRPO+PRM. Computationally, PPO and GRPO+PRM incur heavy overhead from reward/critic model inference, whereas RewardFlow derives verifiable, task-aligned dense rewards directly via graph propagation from terminal signals and observed topology, enabling precise credit assignment at minimal cost. Detailed analysis is in Appendix F.2.

Table 7: Comparison of RewardFlow with process reward modeling baselines on ALFWorld using Qwen2.5-1.5B-Instruct. We report success rate (%) and average training time per step.

Training Method	Success Rate	Time/Step (s)
GRPO + PRM	31.3	513.0
PPO	43.0	502.5
RewardFlow	**68.8**	**320.3**

F.3 The Potential to Use GNN for Reward Propagation

GNN-based reward propagation methods demonstrate challenges in on-policy RL training. In ALFWorld, Sokoban, WebShop, and DeepResearch, GNN-based propagation is less applicable due to practical constraints. GNNs require collecting and labeling additional graph data (e.g., node/action features and ground-truth rewards), followed by separate training phases, which introduce overhead in data quality dependence and generalization challenges on dynamically growing, large-scale graphs from online rollouts. Moreover, GNNs demand additional GPU resources and computation time for inference during each training step, reducing overall efficiency in on-policy RL. Heuristic methods like BFS/PageRank, by contrast, are lightweight, interpretable, and directly leverage the induced graph topology without external training, ensuring reliable propagation of verifiable terminal rewards. This aligns with RewardFlow's goal of simple, scalable credit assignment in sparse-reward, multi-turn agentic scenarios, where exploration is ongoing, and graphs evolve per batch.

GNN-based propagation methods have the potential to be applied in semantically rich environments. GNNs offer advantages in incorporating semantic information from node (state) and edge (action) embeddings, potentially yielding more nuanced reward propagation strategies that account for contextual similarities beyond pure topology. This could be particularly beneficial in fixed-task environments with abundant offline data, such as: (1) video game domains like StarCraft (Vinyals et al., 2019), where state graphs are predefined and GNNs can learn from massive replay buffers to predict value functions; (2) robotics tasks in simulated worlds (e.g., MuJoCo (Todorov et al., 2012)), where physical semantics (e.g., joint angles, object interactions) enable GNNs to generalize across similar trajectories; (3) molecular design (De Cao & Kipf, 2018) or drug discovery graphs (You et al.,

2018), where node features (e.g., atom types) allow learned propagation to optimize sparse rewards like binding affinity; or (4) offline RL in recommendation systems (Chen et al., 2024), with static user-item graphs enabling pre-trained GNNs for reliable, semantics-aware credit assignment. In these cases, a trained GNN could provide task-specific reliability once deployed, and this is a promising extension for future work in RewardFlow variants tailored to such domains.

F.4 Do Reward Contradictions Occur in RewardFlow?

RewardFlow's graph construction avoids such conflicts in deterministic environments due to the inherent consistency of transitions. In ALFWorld, WebShop, and Sokoban, the transition function $P(s'|s, a)$ is deterministic, and states are distinct (textual configurations in ALFWorld and WebShop; grid layouts in Sokoban). Thus, any two rollouts that visit the same state-action pair necessarily reach the identical next state. The induced exploration graph is therefore free of conflicting edges. For reward propagation, our BFS/PageRank algorithms conservatively take the maximum propagated reward per node (equivalent to the shortest-path distance to the nearest successful terminal in BFS), yielding a unique, consistent value for every reachable state regardless of how many or which trajectories discovered it.

The semantic similarity clustering strategy avoids the contradiction of reward assignment. In the highly stochastic DeepResearch environment, semantically equivalent queries can surface different retrieved passages, leading to superficially distinct but functionally identical states. To prevent conflicting reward assignments, we cluster states whose embeddings have high cosine similarity, merge them into a single super-node, and assign the max-pooled propagated reward (or averaged, as ablation shows negligible difference). This simple yet effective deduplication ensures that near-identical research states receive identical credit, eliminating contradictions while preserving the underlying topology. Empirical studies in DeepResearch (Tab. 2) demonstrate that this clustering contributes critically to about 11.2% average gain over Search-R1 (Jin et al., 2025) under high stochasticity.

G Further Experiments

G.1 Detailed Experimental Setting

Evaluation Benchmarks. We show details of chosen benchmarks as follows:

- **ALFWorld** is a synthetic text-based simulator aligned with the embodied benchmark ALFRED (Shridhar et al., 2020), comprising 3,827 household tasks spanning six types: *Pick & Place* (Pick), *Examine in Light* (Look), *Clean & Place* (Clean), *Heat & Place* (Heat), *Cool & Place* (Cool), and *Pick Two & Place* (Pick2).
- **WebShop** is a large-scale benchmark where the agent must fulfill natural-language shopping instructions by issuing text commands such as "search" and "click" to web pages with over 1.18 million real Amazon products.
- **Sokoban** is a classic puzzle game in which agents must observe and analyze the grid layout and push boxes onto target locations, taxing spatial understanding and long-horizon planning. In this work, we randomly generate 6×6 Sokoban boards for training and evaluation.
- **DeepResearch** is an agentic framework for knowledge-intensive QA that uses a search engine to retrieve documents. The environment state consists of retrieved documents, and the agent's actions are generated search queries. A key challenge is the high sensitivity of retrieval: semantically similar queries often return substantially different document sets, which complicates trajectory modeling, knowledge graph construction, and reward modeling. We evaluate on single-hop tasks: NarrativeQA (NQ) (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023), and multi-hop tasks: HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (2Wiki) (Ho et al., 2020), Musique (Trivedi et al., 2022), and Bamboogle (Press et al., 2023).

Training Configuration. We show detailed configurations in our experiments in Tab. 8, including the training part and the algorithm part, for full reproduction. All experiments are conducted using the Verl-Agent (Feng et al., 2025b) framework.

Table 8: Details of training configuration of experiments.

Type	Hyperparameter	ALFWorld	Webshop	Sokoban	DeepResearch
		Qwen2.5-(VL)-1.5B/3B-Instruct			
Learning	Training batch size	16	16	16	256
Learning	Minibatch size for policy update	256	64	256	512
Learning	Max interactive steps	25	15	15	4
Learning	Max prompt length	2048	4096	1024	4096
Learning	Max response length	512	512	512	512
Learning	Training step	100	100	100	200
Learning	Temperature	0.4	0.4	0.4	0.4
Learning	Learning rate	1e-6	1e-6	1e-6	1e-6
Learning	Invalid action penalty	0.1	0.1	0.1	0.01
Learning	Computational cost	2x(A100 & h20)	2x(A100 & h20)	2x(A100 & h20)	2x(A100 & h20)
		Qwen2.5-(VL)-7B-Instruct			
Learning	Training batch size	16	16	16	256
Learning	Minibatch size for policy update	256	64	256	512
Learning	Max interactive steps	25	15	15	4
Learning	Max prompt length	2048	4096	1024	4096
Learning	Max response length	512	512	512	512
Learning	Training step	100	100	100	200
Learning	Temperature	0.4	0.4	0.4	0.4
Learning	Learning rate	1e-6	1e-6	1e-6	1e-6
Learning	Invalid action penalty	0.1	0.1	0.1	0.01
Learning	Computational cost	4x(A100 & h20)	4x(A100 & h20)	4x(A100 & h20)	4x(A100 & h20)
		GRPO			
Algorithm	Group size	8	8	8	5
Algorithm	KL-divergence loss coefficient	0.01	0.01	0.01	0.01
		RLOO			
Algorithm	Group size	8	8	8	5
Algorithm	KL-divergence loss coefficient	0.01	0.01	0.01	0.001
		GiGPO			
Algorithm	Decay weight γ	0.95	0.95	0.95	0.95
Algorithm	weighting coefficient w	1	1	1	1
Algorithm	Group size	8	8	8	5
Algorithm	KL-divergence loss coefficient	0.01	0.01	0.01	0.01
		RewardFlow			
Algorithm	Decay weight γ	0.90	0.90	0.90	0.90
Algorithm	Max graph propagation iteration	1000	1000	1000	1000
Algorithm	Action-level advantage weight α_{action}	1	1	1	1
Algorithm	Trajectory-level advantage weight α_{traj}	1	1	1	1
Algorithm	Group size	8	8	8	5
Algorithm	KL-divergence loss coefficient	0.01	0.01	0.01	0.01

G.2 Training Efficiency

We show average numbers of nodes and edges in the induced graph of rollout sampling (Tab. 9), where we collect the data from ALFWorld and WebShop via the sampling from Qwen2.5-1.5B-Instruct with 8 rollouts, where ALFWorld takes 25 interactive steps, and WebShop takes 15 interactive steps. For Sokoban, we use Qwen2.5-VL-3B-Instruct with 8 rollouts and 15 interactive steps. The results show that for all three agentic environments, the nodes and edges are no more than 42 and 77, respectively, indicating that the induced graph through rollout does not involve numerous nodes and edges, where the graph construction and reward propagation process would not take much time. We also show the average time consumption of each training stage per step to demonstrate that RewardFlow is a lightweight and efficient reward modeling method.

Table 9: Average and maximum size of induced state graphs from 8 on-policy rollouts (Qwen2.5-1.5B/VL-3B-Instruct). Even in the worst case, graphs remain extremely compact.

Environment	**Avg. Nodes**	**Avg. Edges**	**Max Nodes**	**Max Edges**
ALFWorld	28.8	55.9	42	77
WebShop	36.8	48.2	54	65
Sokoban	14.0	21.0	27	39

G.3 Training Until Convergence

We continued training the exact same checkpoints of Sokoban from the main experiments for an additional 100 steps (total 200 steps) under identical hyperparameters. All methods plateau between 150 and 200 steps, confirming full convergence. As shown in Tab 10, RewardFlow's lead over the strongest baseline grows from +22.6% at 100 steps to +28.9% at 200 steps. Notably, while baselines improve by at most +18.7% with the extra budget, RewardFlow gains +22.1%, showing that its denser credit assignment continues to extract meaningful signal even in later training phases. These results

Table 10: Extended training to 200 steps on Sokoban (Qwen2.5-VL-3B-Instruct). All methods fully converge well before 200 steps, and RewardFlow's advantage further widens after convergence.

Method	Success @100 steps (%)	Success @200 steps (%)	Δ (200-100)
RLOO	22.7	41.4	+18.7
GRPO	26.6	39.1	+12.5
GiGPO	21.9	33.6	+11.7
RewardFlow (ours)	**49.2**	**70.3**	**+22.1**

confirm that our reported advantages are not artifacts of early stopping and remain robust (and even strengthen) under fully converged conditions.

G.4 Propagation Strategy

In this section, we try another strategy in BFS: the average distance $d(s) = \frac{1}{|S_{\text{succ}}|}\sum_{s^\star \in S_{\text{succ}}} \text{dist}_{\text{hop}}(s \rightsquigarrow s^\star)$ to compare with the original propagation strategy $d(s) = \min_{s^\star \in S_{\text{succ}}} \text{dist}_{\text{hop}}(s \rightsquigarrow s^\star)$. As shown in Tab. 11, using minimum hop distance for reward propagation demonstrates consistently higher performance than the mean distance. Especially in Sokoban, using minimum distance shows a 15.6% improvement in success rate compared with mean distance. Intuitively, farther routes would dilute the signal, assigning a lower value $R(s)$ to a state with one short path but many long ones, even if the short path is viable. This might yield $\tilde{r}(s_t, a_t) < 0$ for actions leading to such states, discouraging promising transitions and destabilizing the state-wise advantages.

However, while simply integrate mean distance performs worse, alternatives like weighted mean distances could mitigate these issues but would require additional design and empirical support. For example, assigning higher weights to shorter paths (e.g., exponential decay based on hop length) might better aggregate topological information, but this introduces hyperparameters and risks overfitting to rollout noise, complicating the multi-source reverse BFS. Our choice of min distance keeps the approach simple and topology-aware, ensuring reliable propagation, which improves reachability without such overhead. Our results in Table 1 indicate that the min-based method already provides robust, task-centric signals across model sizes.

In addition, incorporating uncertainty into reward propagation is an insightful suggestion, as it could refine the signal by accounting for policy variability, though it introduces subjective elements distinct from objective graph topology. Uncertainty, such as entropy over $\pi_\theta(a|s)$, reflects the LLM policy's confidence rather than inherent environment structure, potentially enhancing exploration in high-uncertainty states. However, integrating it (e.g., via entropy-regularized distances) would require careful adaptation to avoid biasing the objective distance metric.

Table 11: Ablation on propagation strategies.

Model	Environment	Propagation	Success Rate (%)
Qwen-2.5-1.5B-Instruct	ALFWorld	Mean Distance	62.5
		Min Distance	**65.6**
Qwen-2.5-VL-3B-Instruct	Sokoban	Mean Distance	33.6
		Min Distance	**49.2**

G.5 Training dynamics

We show the training dynamics of ALFWorld and Sokoban with Qwen2.5-7B-Instruct in Fig. 7. RewardFlow consistently outperforms prior RL methods, demonstrating objective and accurate state-wise reward modeling.

G.6 State Graph Cases

To illustrate the construction of state graphs, we visualize the graphs derived from varying numbers of rollout trajectories sampled from checkpoints after 100-step training in WebShop and Sokoban. In

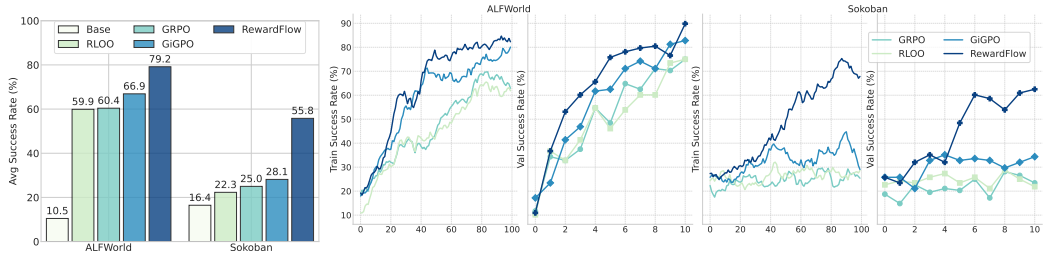


Figure 7: **Performance overview of RewardFlow. Left:** Average success rate (%) on agentic tasks. Each bar represents the average performance over different model sizes using that method. RewardFlow consistently outperforms all baselines, delivering the strongest optimization gains.**Right:** Training dynamics of curves for training and validation success rate, Further details are provided in Sec. 5.

these visualizations, node rewards propagated via BFS are highlighted in green, with deeper shades indicating higher values. Edge rewards are colored red for positive gains, blue for negative gains, and white for zero gains.

The state graphs for WebShop, constructed using 1, 2, and 3 trajectories, are depicted in Figs. 8, 9, and 10, respectively. Corresponding textual descriptions of the states and actions are provided in Figs. 11 and 12.

The state graphs for Sokoban, constructed using 1, 2, and 3 trajectories, are depicted in Figs. 13, 14, and 15, respectively. Corresponding textual and visual descriptions of the states and actions are provided in Figs. 16 and 17.

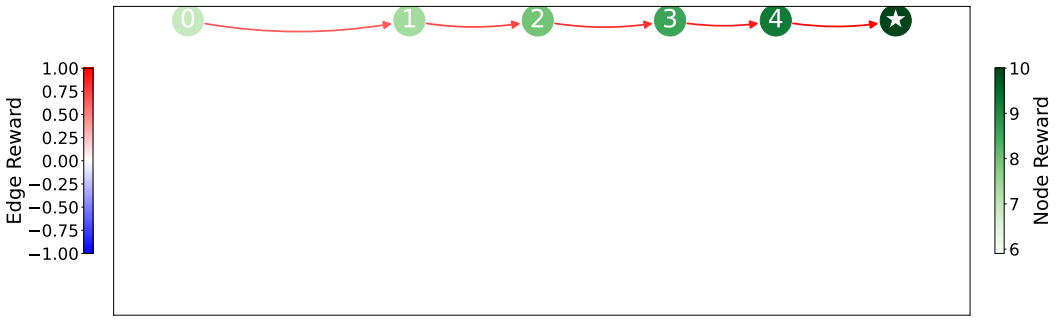


Figure 8: The constructed state graph of the trajectory sampled from WebShop with 1 sampling rollout. The deeper color indicates a higher assigned reward. The initial state is in Node 0, and the success terminal state is in Node ★. The corresponding text representation of nodes and edges can be found in Figs. 11 and 12.

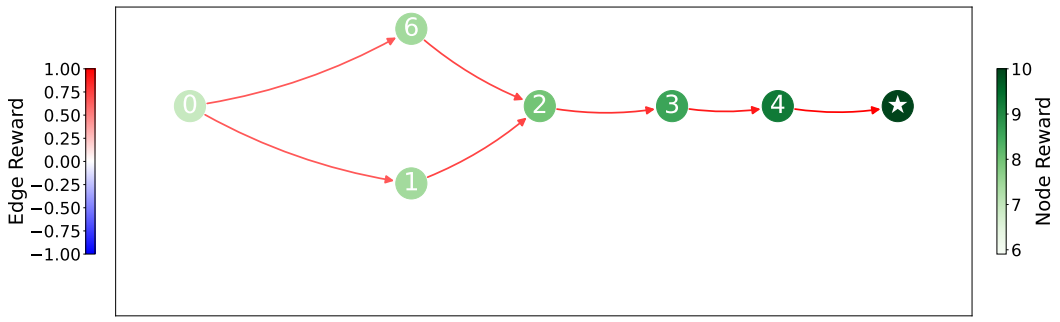


Figure 9: The constructed state graph of the trajectories sampled from WebShop with 2 sampling rollouts. The deeper color indicates a higher assigned reward. The initial state is in Node 0, and the success terminal state is in Node ★. The corresponding text representation of nodes and edges can be found in Figs. 11 and 12.

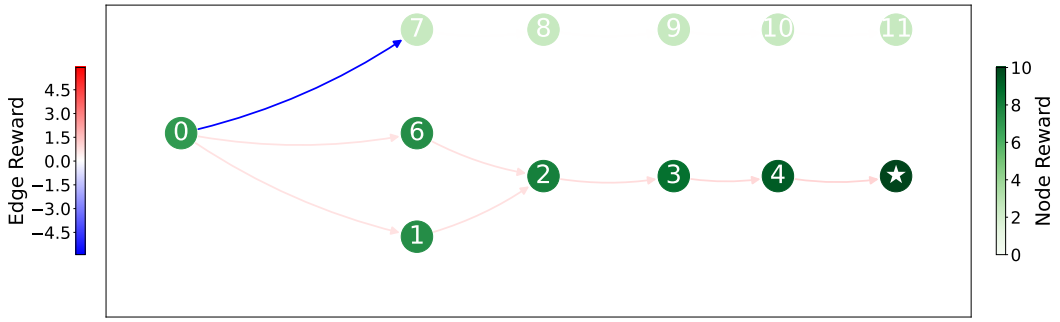


Figure 10: The constructed state graph of the trajectory sampled from WebShop with 3 sampling rollouts. The deeper color indicates a higher assigned reward. The initial state is in Node 0, and the success terminal state is in Node ★. The corresponding text representation of nodes and edges can be found in Figs. 11 and 12.

Node list of examples in WebShop

Node 0: 'Search'
Node 1: 'Back to Search' [SEP] 'Page 1 (Total results: 50)' [SEP] 'Next >' [SEP] 'B07S7HDC88' [SEP] 'VanciLin Mens Casual Leather Fashion Slip-on Loafers' [SEP] '$33.99 to $35.99' [SEP] 'B00ITCBD3Y' [SEP] 'Laredo Womens Spellbound Studded Square Toe Dress Boots Mid Calf Low Heel 1-2" - Black' [SEP] '$132.0 to $150.26' [SEP] 'B09N9T673Q' [SEP] 'Women's Heeled Sandals Sexy Minimalistic Cut out Rear Zipper Wedge Roman Sandals Casual Solid Stiletto Pump for Night Party Comfy Lightweight Summer Shoes' [SEP] '$100.0' [SEP] 'B07KK7G5YJ' [SEP] 'Laeacco Steampunk Clock Backdrops 6.5x6.5ft Vinyl Photography Background Shabby Clock Photo Backdrop Old Clock Punk Grunge Culture Adults Brave Boy Portraits Photo Studio Props' [SEP] '$24.59' [SEP] 'B09P1HJ32G' [SEP] 'VDSOIUTYHFV Infrared Digital Night Vision Device, High-Definition Day and Night Dual-use, Outdoor Photos and Videos are All Black, for Adults Bird Watching, Traveling, Hiking' [SEP] '$840.99' [SEP] 'B089KGBJGW' [SEP] 'ZINUS 10 Inch Cooling Copper ADAPTIVE Pocket Spring Hybrid Mattress / Moisture Wicking Cover / Cooling Foam / Pocket Innersprings for Motion Isolation / Mattress-in-a-Box, Full' [SEP] '$289.0' [SEP] 'B09P57ZKGP' [SEP] '10 Pack 512MB USB Flash Drives Gift Stick Business Bidding Thumb Drive USB 2.0 high Speed Pen Drives (512MB, Black)' [SEP] '$100.0' [SEP] 'B001C6JXJU' [SEP] 'Premier HiDef Grey Electric Projection Screen with Low Voltage Motor Viewing Area: 50" H x 80" W' [SEP] '$2721.34' [SEP] 'B085HGD5Z3' [SEP] 'haoricu Womens Platform Sandals Summer Fashion Ladies Bow Tie Flats Rome Casual Basic Slipper' [SEP] '$14.98 to $25.98' [SEP] 'B09QKJHF4Q' [SEP] 'LIUEONG Lady Metal Buckle Platform Shoes Booties Comfortable High Top Boots Casual Shoes Comfort Women's Closed Toe Ankle Strap' [SEP] '$37.99'
Node 2: 'Back to Search' [SEP] '< Prev' [SEP] 'size' [SEP] '6.5' [SEP] '7' [SEP] '8' [SEP] '8.5' [SEP] '9' [SEP] '9.5' [SEP] '10' [SEP] '10.5' [SEP] '11' [SEP] '11.5' [SEP] '12' [SEP] '13' [SEP] '14' [SEP] 'color' [SEP] '1877black' [SEP] '1877blue' [SEP] '228black' [SEP] 'black137' [SEP] 'blue137' [SEP] 'l.brown137' [SEP] 'r.brown137' [SEP] '1877r.brown' [SEP] '228blue' [SEP] 'r.brown-hole228' [SEP] '1877brown' [SEP] '228r.brown-a' [SEP] 'y.brown-hole228' [SEP] 'black-hole228' [SEP] '228l.brown' [SEP] 'r.brown2070' [SEP] '228r.brown' [SEP] 'black2070' [SEP] 'r.brown2626' [SEP] 'r.brown-hole1887' [SEP] 'black2626' [SEP] 'black1901' [SEP] 'r.brown2060' [SEP] 'l.brown2060' [SEP] 'VanciLin Mens Casual Leather Fashion Slip-on Loafers' [SEP] 'Price: $33.99 to $35.99' [SEP] 'Rating: N.A.' [SEP] 'Description' [SEP] 'Features' [SEP] 'Reviews' [SEP] 'Buy Now'
Node 3: 'Back to Search' [SEP] '< Prev' [SEP] 'size' [SEP] '6.5' [SEP] '7' [SEP] '8' [SEP] '8.5' [SEP] '9' [SEP] '9.5' [SEP] '10' [SEP] '10.5' [SEP] '11' [SEP] '11.5' [SEP] '12' [SEP] '13' [SEP] '14' [SEP] 'color' [SEP] '1877black' [SEP] '1877blue' [SEP] '228black' [SEP] 'black137' [SEP] 'blue137' [SEP] 'l.brown137' [SEP] 'r.brown137' [SEP] '1877r.brown' [SEP] '228blue' [SEP] 'r.brown-hole228' [SEP] '1877brown' [SEP] '228r.brown-a' [SEP] 'y.brown-hole228' [SEP] 'black-hole228' [SEP] '228l.brown' [SEP] 'r.brown2070' [SEP] '228r.brown' [SEP] 'black2070' [SEP] 'r.brown2626' [SEP] 'r.brown-hole1887' [SEP] 'black2626' [SEP] 'black1901' [SEP] 'r.brown2060' [SEP] 'l.brown2060' [SEP] 'VanciLin Mens Casual Leather Fashion Slip-on Loafers' [SEP] 'Price: $33.99 to $35.99' [SEP] 'Rating: N.A.' [SEP] 'Description' [SEP] 'Features' [SEP] 'Reviews' [SEP] 'Buy Now' clicked attributes: [228r.brown-a]
Node 4: 'Back to Search' [SEP] '< Prev' [SEP] 'size' [SEP] '6.5' [SEP] '7' [SEP] '8' [SEP] '8.5' [SEP] '9' [SEP] '9.5' [SEP] '10' [SEP] '10.5' [SEP] '11' [SEP] '11.5' [SEP] '12' [SEP] '13' [SEP] '14' [SEP] 'color' [SEP] '1877black' [SEP] '1877blue' [SEP] '228black' [SEP] 'black137' [SEP] 'blue137' [SEP] 'l.brown137' [SEP] 'r.brown137' [SEP] '1877r.brown' [SEP] '228blue' [SEP] 'r.brown-hole228' [SEP] '1877brown' [SEP] '228r.brown-a' [SEP] 'y.brown-hole228' [SEP] 'black-hole228' [SEP] '228l.brown' [SEP] 'r.brown2070' [SEP] '228r.brown' [SEP] 'black2070' [SEP] 'r.brown2626' [SEP] 'r.brown-hole1887' [SEP] 'black2626' [SEP] 'black1901' [SEP] 'r.brown2060' [SEP] 'l.brown2060' [SEP] 'VanciLin Mens Casual Leather Fashion Slip-on Loafers' [SEP] 'Price: $33.99 to $35.99' [SEP] 'Rating: N.A.' [SEP] 'Description' [SEP] 'Features' [SEP] 'Reviews' [SEP] 'Buy Now' clicked attributes: [228r.brown-a, 6.5]
Node ★: Thank you for shopping with us! [SEP] Your code: [SEP] None [SEP] (Paste it in your MTurk interface.) [SEP] Purchased [SEP] asin [SEP] B07S7HDC88 [SEP] options [SEP] "color": "228r.brown-a", "size": "6.5" [SEP] attrs [SEP] None [SEP] category [SEP] None [SEP] query [SEP] None [SEP] product category [SEP] None [SEP] Target [SEP] asin [SEP] options [SEP] attrs [SEP] price upper [SEP] instuction text [SEP] category [SEP] product category [SEP] query [SEP] Goal [SEP] None [SEP] Reward [SEP] Your score (min 0.0, max 1.0) [SEP] 1.0 [SEP] Reward Details [SEP] None
Node 6: 'Back to Search' [SEP] 'Page 1 (Total results: 23)' [SEP] 'Next >' [SEP] 'B07S7HDC88' [SEP] 'VanciLin Mens Casual Leather Fashion Slip-on Loafers' [SEP] '$33.99 to $35.99' [SEP] 'B07YCGBPRD' [SEP] 'OTAO Privacy Screen Protector for iPhone 11 Pro Max/iPhone Xs Max 6.5 Inch True 28°Anti Spy Tempered Glass Full-Coverage (2-pack)' [SEP] '$9.98' [SEP] 'B085HGD5Z3' [SEP] 'haoricu Womens Platform Sandals Summer Fashion Ladies Bow Tie Flats Rome Casual Basic Slipper' [SEP] '$14.98 to $25.98' [SEP] 'B07KK7G5YJ' [SEP] 'Laeacco Steampunk Clock Backdrops 6.5x6.5ft Vinyl Photography Background Shabby Clock Photo Backdrop Old Clock Punk Grunge Culture Adults Brave Boy Portraits Photo Studio Props' [SEP] '$24.59' [SEP] 'B09J95S478' [SEP] 'My Sanity Question Giraffe Christmas Pattern Black Ugly Wool Christmas Sweater Pullover Long Sleeve Sweater for Men Women, Couple Matching, Friends' [SEP] '$39.99' [SEP] 'B09N9T673Q' [SEP] 'Women's Heeled Sandals Sexy Minimalistic Cut out Rear Zipper Wedge Roman Sandals Casual Solid Stiletto Pump for Night Party Comfy Lightweight Summer Shoes' [SEP] '$100.0' [SEP] 'B07TN3Y9H1' [SEP] 'Maxim 32478SWSBRBP Finn Mid-Century Modern Satin White Glass Ball Bath Vanity Wall Mount, 5-Light 200 Total Watts, 7"H x 44"W, Satin Brass/Brushed Platinum' [SEP] '$219.99' [SEP] 'B09QKJHF4Q' [SEP] 'LIUEONG Lady Metal Buckle Platform Shoes Booties Comfortable High Top Boots Casual Shoes Comfort Women's Closed Toe Ankle Strap' [SEP] '$37.99' [SEP] 'B0912FQSMT' [SEP] '450 Bluetooth Marine Gauge Receiver and 6.5-Inch Speaker Package' [SEP] '$224.02' [SEP] 'B088TWNTRB' [SEP] 'MIA Delena' [SEP] '$21.89 to $55.08'
Node 7: 'Back to Search' [SEP] 'Page 1 (Total results: 50)' [SEP] 'Next >' [SEP] 'B07RPMCNKD' [SEP] 'ULTRAIDEAS Women's Comfy Lightweight Slippers Non-Slip House Shoes for Indoor & Outdoor' [SEP] '$19.99 to $21.99' [SEP] 'B07HP6LVRS' [SEP] 'MCICI Mens Loafers Moccasin Driving Shoes Premium Genuine Leather Casual Slip On Flats Fashion Slipper Breathable Big Size' [SEP] '$15.99 to $32.99' [SEP] 'B07S7HDC88' [SEP] 'VanciLin Mens Casual Leather Fashion Slip-on Loafers' [SEP] '$33.99 to $35.99' [SEP] 'B09HMCKZQW' [SEP] 'Copercn Dress Ankle Booties For Women Ladies Fashion Casual Chunky Block High Heels Dress Pump Thermal Short Boots Winter Fall Dressy Short Boots For Business Work Wedding Party Dresses' [SEP] '$17.99 to $23.99' [SEP] 'B09F2MFG1Q' [SEP] 'Womens Mens house slippers Memory Foam Garden Plush Lining Slip On Rubber sole Indoor Outdoor House Casual Shoes (Black adult-10.5-11)' [SEP] '$100.0' [SEP] 'B09S6VN97V' [SEP] 'Skechers Women's, Relaxed Fit: Arch Fit - Commute Clog Taupe 11 M' [SEP] '$74.95' [SEP] 'B008QYOGZ2' [SEP] 'OluKai MEA Ola - Men's Supportive Sandal Charcoal/Dkjava - 14' [SEP] '$100.0' [SEP] 'B07XDRVVYM' [SEP] 'Clarks Women's Un Adorn Sling Sandal' [SEP] '$34.99 to $99.95' [SEP] 'B01LXTLLVG' [SEP] 'PUMA V1.08 Tricks Top Trainer Mens Soccer Sneakers/Boots-Grey-5.5' [SEP] '$55.5' [SEP] 'B09QXF3V3X' [SEP] 'DEUVOUM Summer Trend Mesh Shoes Men's Sports Shoes Solid Color Lace-Up Sneakers Fashion All-Match Walking Shoes Outdoor Hiking Shoes Non-Slip Shock-Absorbing Casual Sports Shoes' [SEP] '$100.0'
Node 8: 'Back to Search' [SEP] '< Prev' [SEP] 'size' [SEP] '6.5' [SEP] '7' [SEP] '8' [SEP] '8.5' [SEP] '9.5' [SEP] '10' [SEP] '10.5' [SEP] '11' [SEP] '11.5' [SEP] 'color' [SEP] 'black' [SEP] 'blue' [SEP] 'brown' [SEP] 'white' [SEP] 'yellow' [SEP] 'black-b' [SEP] 'blue-a' [SEP] 'yellow-a' [SEP] 'MCICI Mens Loafers Moccasin Driving Shoes Premium Genuine Leather Casual Slip On Flats Fashion Slipper Breathable Big Size' [SEP] 'Price: $15.99 to $32.99' [SEP] 'Rating: N.A.' [SEP] 'Description' [SEP] 'Features' [SEP] 'Reviews' [SEP] 'Buy Now'
Node 9: 'Back to Search' [SEP] '< Prev' [SEP] 'size' [SEP] '6.5' [SEP] '7' [SEP] '8' [SEP] '8.5' [SEP] '9.5' [SEP] '10' [SEP] '10.5' [SEP] '11' [SEP] '11.5' [SEP] 'color' [SEP] 'black' [SEP] 'blue' [SEP] 'brown' [SEP] 'white' [SEP] 'yellow' [SEP] 'black-b' [SEP] 'blue-a' [SEP] 'yellow-a' [SEP] 'MCICI Mens Loafers Moccasin Driving Shoes Premium Genuine Leather Casual Slip On Flats Fashion Slipper Breathable Big Size' [SEP] 'Price: $15.99 to $32.99' [SEP] 'Rating: N.A.' [SEP] 'Description' [SEP] 'Features' [SEP] 'Reviews' [SEP] 'Buy Now' clicked attributes: [brown]
Node 10: 'Back to Search' [SEP] '< Prev' [SEP] 'size' [SEP] '6.5' [SEP] '7' [SEP] '8' [SEP] '8.5' [SEP] '9.5' [SEP] '10' [SEP] '10.5' [SEP] '11' [SEP] '11.5' [SEP] 'color' [SEP] 'black' [SEP] 'blue' [SEP] 'brown' [SEP] 'white' [SEP] 'yellow' [SEP] 'black-b' [SEP] 'blue-a' [SEP] 'yellow-a' [SEP] 'MCICI Mens Loafers Moccasin Driving Shoes Premium Genuine Leather Casual Slip On Flats Fashion Slipper Breathable Big Size' [SEP] 'Price: $15.99 to $32.99' [SEP] 'Rating: N.A.' [SEP] 'Description' [SEP] 'Features' [SEP] 'Reviews' [SEP] 'Buy Now' clicked attributes: [brown, 6.5]
Node 11: Thank you for shopping with us! [SEP] Your code: [SEP] None [SEP] (Paste it in your MTurk interface.) [SEP] Purchased [SEP] asin [SEP] B07HP6LVRS [SEP] options [SEP] "color": "brown", "size": "6.5" [SEP] attrs [SEP] None [SEP] category [SEP] None [SEP] query [SEP] None [SEP] product category [SEP] None [SEP] Target [SEP] asin [SEP] options [SEP] attrs [SEP] price upper [SEP] instuction text [SEP] category [SEP] product category [SEP] query [SEP] Goal [SEP] None [SEP] Reward [SEP] Your score (min 0.0, max 1.0) [SEP] 0.7142857142857143 [SEP] Reward Details [SEP] None

Figure 11: The node list of cases in WebShop in Figs. 8, 9, and 10.

Edge list of examples in WebShop

Node 0 -> Node 6: search[228r.brown-a 6.5]
Node 0 -> Node 1: search[228r.brown-a, size 6.5, price lower than 70.00]
Node 0 -> Node 7: search[228r.brown-a 6.5 rubber outsole men slipper non slip]
Node 1 -> Node 2: click[b07s7hdc88]
Node 2 -> Node 3: click[228r.brown-a]
Node 3 -> Node 4: click[6.5]
Node 4 -> Node ★: click[buy now]
Node 6 -> Node 2: click[b07s7hdc88]
Node 7 -> Node 8: click[b07hp6lvrs]
Node 8 -> Node 9: click[brown]
Node 9 -> Node 10: click[6.5]
Node 10 -> Node 11: click[buy now]

Figure 12: The edge list of cases in WebShop in Figs. 8, 9, and 10.

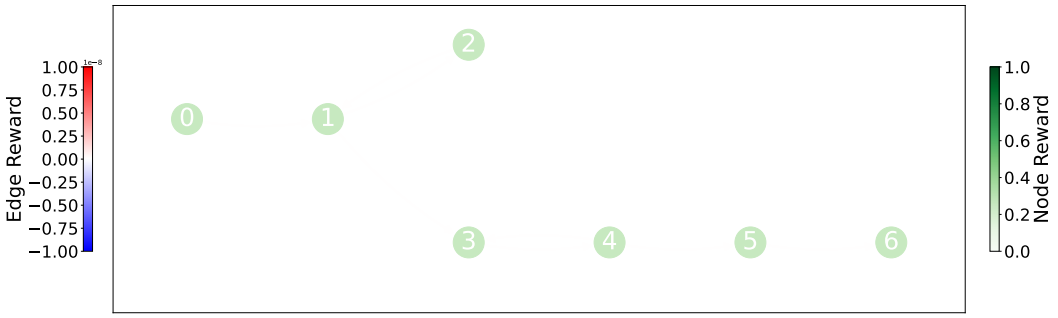


Figure 13: The constructed state graph of the trajectory sampled from Sokoban with 1 sampling rollout. The deeper color indicates a higher assigned reward. The initial state is in Node 0, and the success terminal state is in Node ★. The corresponding text representation of nodes and edges can be found in Figs. 16 and 17.

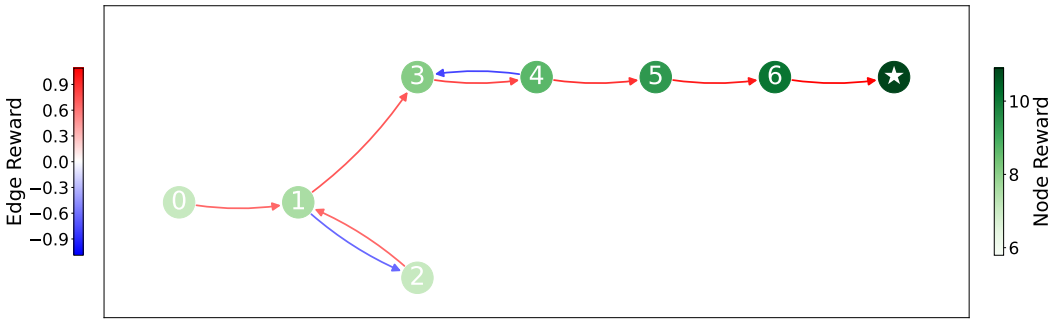


Figure 14: The constructed state graph of the trajectories sampled from Sokoban with 2 sampling rollout. The deeper color indicates a higher assigned reward. The initial state is in Node 0, and the success terminal state is in Node ★. The corresponding text representation of nodes and edges can be found in Figs. 16 and 17.

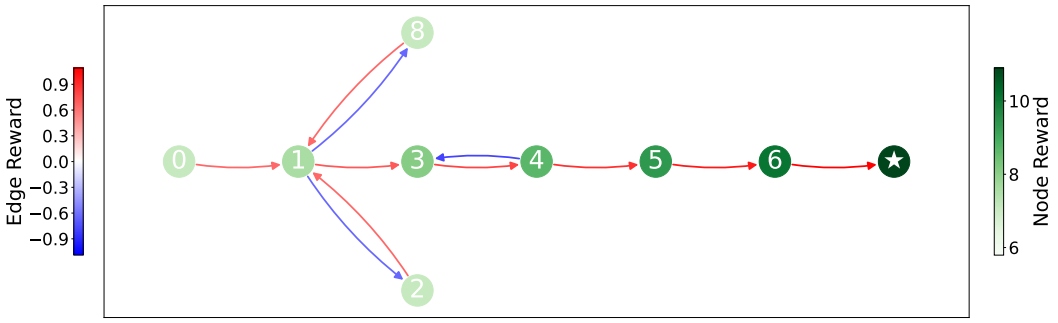
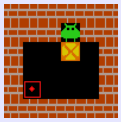


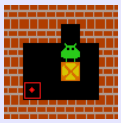
Figure 15: The constructed state graph of the trajectories sampled from Sokoban with 3 sampling rollouts. The deeper color indicates a higher assigned reward. The initial state is in Node 0, and the success terminal state is in Node ★. The corresponding text representation of nodes and edges can be found in Figs. 16 and 17.

Node list of examples in Sokoban

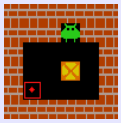
Node 0:



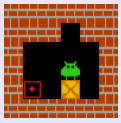
Node 1:



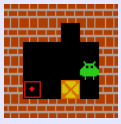
Node 2:



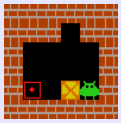
Node 3:



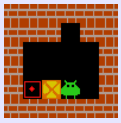
Node 4:



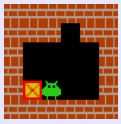
Node 5:



Node 6:



Node ★:



Node 8:

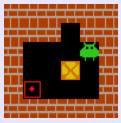


Figure 16: The node list of cases in Sokoban in Figs. 13, 14, and 15.

Edge list of examples in Sokoban

Node 0 -> Node 1: down
Node 1 -> Node 3: down
Node 1 -> Node 2: up
Node 1 -> Node 8: right
Node 2 -> Node 1: down
Node 3 -> Node 4: right
Node 4 -> Node 5: down
Node 4 -> Node 3: left
Node 5 -> Node 6: left
Node 6 -> Node ★: left
Node 8 -> Node 1: left

Figure 17: The edge list of cases in Sokoban in Figs. 13, 14, and 15.