MATCHED-CONTEXT GROUP POLICY OPTIMIZATION FOR LLM AGENT TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Training long-horizon language model agents requires assigning credit to intermediate decisions from sparse terminal feedback. Within a finite group of sibling rollouts, useful action comparisons depend on their context, evidence for both alternatives, and how that evidence enters learning. We propose *Matched-Context Group Policy Optimization (MCGPO)*, a procedure for constructing weighted action preferences from already collected trajectories. MCGPO defines comparison contexts with pre-action state summaries and observed rubric progress, annotated by task rules. Two-action support selects among ordered exact and coarse matching rules. Fixed folds separate candidate decisions from the terminal outcomes used to orient their preferences, and support-dependent evidence weights scale an auxiliary pairwise objective alongside shared GraphGPO task learning. MCGPO reuses rollout data and each response's original history, preserving critic-free training. Across six environment–backbone settings with three training seeds each, MCGPO improves mean success over matched GraphGPO: by 0.47–1.46 percentage points on ALFWorld and WebShop with Qwen2.5-1.5B and 7B, and by 3.96 and 5.16 points on WebShop with Qwen3-4B and 8B. The ALFWorld gains build on baseline success above 92%, where remaining headroom is limited. Integrated success improves in three of four Qwen2.5 settings. Three-seed factorial controls support opposite-fold evidence and evidence weighting; joint state–rubric matching achieves the highest mean in all four representation-control settings.

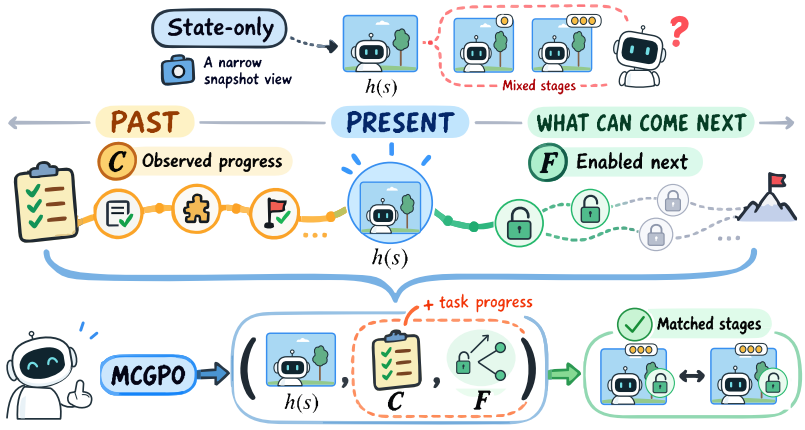

Figure 1: Task context for local action comparisons. The same state summary $h(s)$ can occur at different stages of a task. MCGPO groups decisions using the state summary and observed rubric progress C, with enabled criteria annotated by the rule-derived frontier F.

1 INTRODUCTION

Training large language model agents requires *credit assignment*: sparse terminal rewards must guide the intermediate actions that shape task outcomes (Zhou et al., 2024; Chen et al., 2025; Wang et al.,

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

2025). In ALFWorld (Shridhar et al., 2021) and WebShop (Yao et al., 2022), success depends on a sequence of interdependent decisions. Group-relative reinforcement learning offers critic-free updates by comparing outcomes within a group of sampled responses or trajectories (Shao et al., 2024; Ahmadian et al., 2024; Yu et al., 2025). For multi-turn agents, GiGPO localizes these comparisons at recurring anchor states (Feng et al., 2025), while trajectory-graph methods extract step credit from collected transitions (Li et al., 2026b; Cheng et al., 2026). However, grouping by repeated states or transitions can compare decisions with different prior task progress, obscuring what their outcome contrast represents.

This mismatch arises when *the same state summary describes different remaining decision problems*. A page, location, or inventory summary may omit progress established before the current choice. Consequently, an action taken after more prerequisites have been met can inherit a higher success rate, while a useful action taken earlier can receive weaker relative feedback. Their outcome contrast can mix prior progress with the contribution of the current action. A captured ALFWorld task illustrates this representational issue: two empty-handed visits to the same countertop share a coarse state key, although one follows acquiring and relocating a target object (Appendix D). Adding context can separate such decisions, but stricter conditions also reduce the repeated evidence available for comparing different actions.

Recent work enriches comparison context to address this tension. HGPO combines groups at multiple history depths, and BiPACE uses policy representations and action-conditioned peer baselines (He et al., 2026; Wang et al., 2026a). Task structure offers another source of context: explicit criteria guide reward aggregation (Gunjal et al., 2025; Lv et al., 2026), policy stages and memory (Li et al., 2026a), and milestone-level credit (Wang et al., 2026d; Qian et al., 2026). However, an explicit preference between different actions also needs evidence for both alternatives. A richer context can leave one action poorly supported, so a larger outcome gap may reflect only a few trajectories. The remaining challenge is to select comparisons and weight their influence using the evidence available within a finite rollout group. This leads to our research question:

RESEARCH QUESTION

How can a finite group of existing rollouts supply supported local action preferences for policy learning?

We introduce MCGPO (Matched-Context Group Policy Optimization) to construct auxiliary action preferences from collected sibling rollouts. The core idea is to select a comparison context supported by both actions before forming a weighted preference. **Match** groups pre-action decisions by state summaries and observed rubric progress, with a rule-derived frontier annotating progress. It tries ordered exact and coarse state rules, selecting the first with evidence for both actions. **Label** uses terminal outcomes from the opposite fixed fold to orient pairs and assign support-discounted evidence weights. **Learn** optimizes a weighted pairwise objective alongside the shared GraphGPO task objective. The labels express observational outcome contrasts under the rollout policy; their weights control each pair's optimization contribution. MCGPO reuses rollout data and preserves critic-free training and each response's original history.

We evaluate MCGPO on ALFWorld and WebShop with Qwen2.5-1.5B and 7B, using three training runs per setting. Mean success improves over matched GraphGPO in all four settings by 0.47–1.46 percentage points, including ALFWorld baselines already above 92% success. A three-seed WebShop extension yields gains of 3.96 and 5.16 points with Qwen3-4B and 8B, respectively. Integrated success over the full training horizon improves in three Qwen2.5 settings, with positive paired differences in each of their three seeds. Three-seed factorial controls support opposite-fold evidence and evidence weighting. Across all four representation-control settings, joint state–rubric conditioning raises three-seed mean success over both State-only and Rubric-only matching. These results demonstrate performance gains across the tested backbones and support joint state–rubric conditioning as an effective comparison design.

Our contributions are threefold:

- **Supported local preference construction.** We combine explicit comparison contexts, two-action support checks, and opposite-fold terminal evidence within an existing rollout group.

- **Evidence-weighted policy learning.** We connect these observational preferences to an auxiliary pairwise objective while retaining the shared task objective and original response histories.
- **Evaluation of the construction and its scope.** Matched comparisons, supervision controls, and representation diagnostics assess overall utility and setting-dependent component effects.

2 Related Work

Agent training and group-based credit. Agent RL learns from interaction outcomes (Zhou et al., 2024; Chen et al., 2025; Wang et al., 2025; Luo et al., 2025), with GRPO, RLOO, and DAPO estimating within-group advantages (Shao et al., 2024; Ahmadian et al., 2024; Yu et al., 2025). GiGPO groups anchor states; HGPO combines historical-context depths (Feng et al., 2025; He et al., 2026). SALT matches recent interaction histories, while GraphGPO and RTMC aggregate credit through transition graphs and shared-state trees (Li et al., 2026b; Cheng et al., 2026; Wang et al., 2026b). In the same sparse-reward, long-horizon setting, G^2PO merges identical observations into a global state-transition graph and standardizes temporal-difference errors across edges to assign action credit (Wang et al., 2026c). AT-GRPO adapts tree-expansion depth to dialogue stage (Peng et al., 2026); STAPO selects steps using state-normalized entropy and combines trajectory-aware rewards with trajectory-independent penalties (Qi et al., 2026). Other designs use peer baselines, progress signals, or hierarchical updates (Wang et al., 2026a; Fan & Liu, 2026; Yang et al., 2026; Tan et al., 2026; Jin et al., 2026). ECPO calibrates anchor-state credit using action-return shrinkage and reliability gates (Li et al., 2026c). MCGPO instead constructs action-pair supervision: pre-action state and rubric progress define comparability, two-action support selects the matching resolution, and opposite-fold outcomes label an auxiliary pairwise objective.

Task-structured supervision. Task structure supports rubric rewards, exploration, and finite-state reward representations (Gunjal et al., 2025; Zhou et al., 2026; Toro Icarte et al., 2022). RubricEM organizes stages and memory; GEAR models criterion dependencies (Li et al., 2026a; Lv et al., 2026). SRaR, TRCA, and DRACO provide rubric-based step feedback (Xie et al., 2026; Zhang et al., 2026; Gandhi et al., 2026). BEACON combines milestone groups with reward shaping (Wang et al., 2026d); MileGPO infers milestones and traps from rollout graphs (Qian et al., 2026). MCGPO uses task-specific progress extractors to condition comparisons, with terminal outcomes supplying preference labels. SALT and MileGPO derive their comparison structure from histories and observed graphs.

Process, preference, and value supervision. Process supervision extends terminal feedback (Uesato et al., 2022; Lightman et al., 2024). DPO, REBEL, SimPO, and Contrastive Preference Learning optimize response or sequential-policy comparisons (Rafailov et al., 2023; Gao et al., 2024; Meng et al., 2024; Hejna et al., 2024); Step-DPO uses step preferences (Lai et al., 2024). Agent Q, VinePPO, OmegaPRM, and RTPO obtain intermediate evidence through search or additional continuations (Putta et al., 2024; Kazemnejad et al., 2025; Luo et al., 2024; Li et al., 2026d). POISE and SAPO learn value baselines or joint policy–value models (Choi et al., 2026; Liang et al., 2026). MCGPO reuses collected sibling rollouts, separating candidate decisions from label contributors with fixed folds. Appendix Table 4 compares these designs.

3 Methodology

For each task x, π_{old} collects eight sibling trajectories τ_i containing observations $s_{i,t}$, responses $z_{i,t}$, canonical actions $a_{i,t}$, and terminal success $y_i \in \{0, 1\}$. Rollout slots define fixed folds $\mathcal{I}_0, \mathcal{I}_1$ of four trajectories each. MCGPO turns these rollouts into local preferences through **Match–Label–Learn** (Figure 2). In each direction, one fold supplies candidate decisions and the other supplies outcome evidence; swapping roles uses both halves.

3.1 Task-conditioned matching (Match)

A comparison context combines the pre-action state summary with observed task progress:

$$g_{i,t}^{(\ell)} = \big(h^{(\ell)}(s_{i,t}), C_{i,t}, F_{i,t}\big), \qquad \ell \in \{\text{exact}, \text{coarse}\}. \tag{1}$$

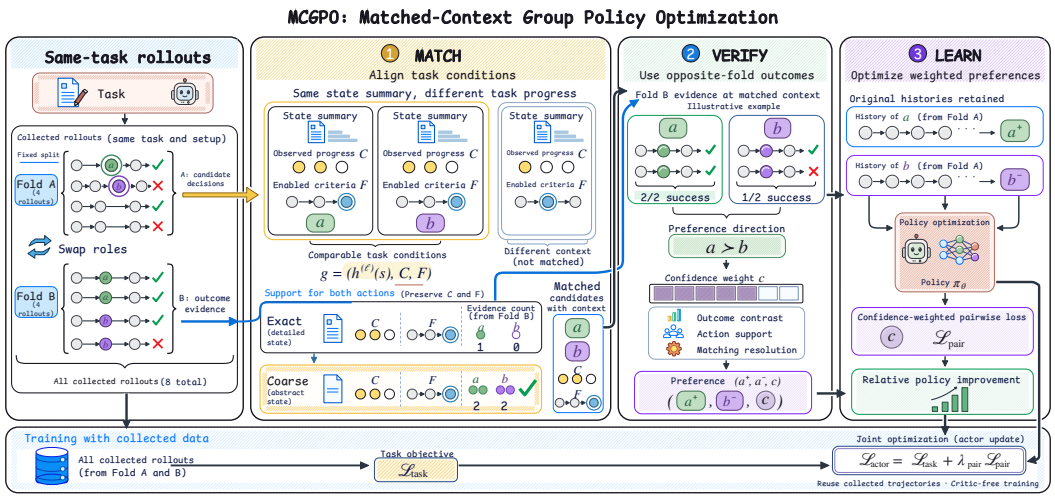

Figure 2: Overview of MCGPO. The Verify block labels matched actions using opposite-fold terminal outcomes. The resulting evidence-weighted pairwise loss complements the shared task objective.

Here $C_{i,t}$ contains rubric channels with finite positive pre-action status, and $F_{i,t}$ annotates inactive criteria enabled by task rules. Exact and coarse are ordered state-matching rules that preserve (C, F); coarse abstracts identity fields and can include task-relevant bins. Appendix A.1 specifies the fields and extractors.

At each rule, each trajectory contributes its first eligible visit to a context. Candidate row pairs $p = ((i,t),(j,u)) \in \mathcal{C}_c^{(\ell)}$ come from distinct trajectories in fold c, share a key $g_p^{(\ell)}$, and take distinct actions a_p, b_p. Let $\mathcal{E}_{1-c}^{(\ell)}(g,a)$ contain opposite-fold trajectories whose retained visit to g takes action a, and let $n_a^{(\ell)}(g) = |\mathcal{E}_{1-c}^{(\ell)}(g,a)|$. The balanced support is $n_\star^{(\ell)}(p) = \min\{n_{a_p}^{(\ell)}(g_p^{(\ell)}), n_{b_p}^{(\ell)}(g_p^{(\ell)})\}$, giving

$$\ell^\star(p) = \min_{\text{exact} \prec \text{coarse}} \{\ell : p \in \mathcal{C}_c^{(\ell)},\ n_\star^{(\ell)}(p) \geq m\}. \tag{2}$$

We use $m = 1$. Pairs without a supported rule are omitted. This gate precedes the outcome test: a supported exact match retains that rule even if its later contrast falls below threshold. Appendix A.1 gives the first-visit and eligibility definitions.

3.2 Cross-rollout preferences (Label)

Let $\widehat{p}_{1-c}^{(\ell)}(g,a)$ be mean terminal success over $\mathcal{E}_{1-c}^{(\ell)}(g,a)$. At the selected key and resolution,

$$\widehat{u}_p = \widehat{p}_{1-c}^{(\ell^\star)}(g,a_p) - \widehat{p}_{1-c}^{(\ell^\star)}(g,b_p), \tag{3}$$

$$c_p = |\widehat{u}_p| \frac{n_\star}{n_\star + \kappa} q_{\ell^\star}, \qquad n_\star = n_\star^{(\ell^\star(p))}(p). \tag{4}$$

The sign of $\widehat{u}_p$ orders the rows as (p^+, p^-); we retain contrasts with $|\widehat{u}_p| \geq \epsilon$. The evidence weight combines contrast magnitude, support shrinkage, and a preset resolution factor. These labels express observational outcome contrasts under the rollout policy; Appendix A.2 details their conditional outcome provenance.

For each sibling trajectory pair, we retain the earliest qualified comparison, minimizing $\max(t_i, t_j)$ and breaking time ties by c_p. A disjoint allocator keeps up to two preferences per fold, using each selected decision row once. This yields a bounded preference set $\mathcal{P}$ from the collected batch. Algorithm 1 and the parameter settings appear in Appendix A.

3.3 EVIDENCE-WEIGHTED POLICY LEARNING (LEARN)

Each preferred response should receive a larger policy improvement than its paired alternative. Using the original token histories $H_{i,t,k}$ and response mask $m_{i,t,k}$, define the length-normalized log-probability change

$$d_\theta(i,t) = \frac{\sum_k m_{i,t,k}[\log \pi_\theta(z_{i,t,k} \mid H_{i,t,k}) - \log \pi_{\text{old}}(z_{i,t,k} \mid H_{i,t,k})]}{\max(1, \sum_k m_{i,t,k})}. \tag{5}$$

For pairs colocated in actor microbatch $\mathcal{B}$, the auxiliary loss is

$$\mathcal{L}_{\text{pair}} = \frac{\sum_{(+,-)\in\mathcal{P}_\mathcal{B}} c_{+,-} \operatorname{softplus}(-[d_\theta(+) - d_\theta(-)])}{\max(1, \lfloor |\mathcal{B}|/2 \rfloor)}. \tag{6}$$

Normalization by the microbatch's pairing capacity makes the contribution scale with supervision supply; $c_{+,-}$ controls each pair's influence. The complete rollout group also supplies GraphGPO (Cheng et al., 2026) task advantages, giving

$$\mathcal{L}_{\text{actor}}(\mathcal{B}) = \mathcal{L}_{\text{task}}(\mathcal{B}) + \lambda_{\text{pair}}\mathcal{L}_{\text{pair}}. \tag{7}$$

Here $\mathcal{L}_{\text{task}}$ is the shared clipped policy loss with reference-policy and entropy terms (Appendix A.3). Tasks without preferences still receive this task update. The construction reuses sampled trajectories and policy log probabilities, preserving critic-free training. Appendix A.2 derives the local pairwise update.

4 EXPERIMENTS

We organize the evaluation around four questions: whether MCGPO improves task completion and learning progress (RQ1), how comparison conditions, outcome evidence, and preference learning contribute (RQ2), how matched contexts produce weighted supervision on WebShop (RQ3), and whether the WebShop gains extend to Qwen3 backbones (RQ4).

4.1 EXPERIMENTAL SETUP

4.1.1 BENCHMARKS AND MODELS

We evaluate household interaction in ALFWorld (Shridhar et al., 2021) and product search and purchase in WebShop (Yao et al., 2022). The main experiments use Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct (Yang et al., 2024), with training seeds 0, 1, and 2. GraphGPO supplies the shared task-credit objective and matched baseline, measuring the effect of the complete rubric-conditioned preference branch. WebShop rubric extraction uses structured training information from the goal scorer, including attribute, option, and price matches and a pre-purchase preview score (Appendix A.1). Controls include GraphGPO with rubric-event shaping. For Qwen3-4B/8B, *GraphGPO + shaping* uses the same scorer information as MCGPO through training rewards, with three seeds per backbone.

4.1.2 IMPLEMENTATION DETAILS

Within each backbone, MCGPO and GraphGPO use the same rollout, optimizer, and environment settings. Following the agent protocol of Feng et al. (2025), the Qwen2.5 policy conditions on the two most recent interaction steps and generates reasoning inside `<think>` tags followed by an action inside `<action>` tags. Each response is limited to 512 generated tokens. The environment horizons are 50 turns for ALFWorld and 15 for WebShop.

Each policy update uses 16 task groups with eight sibling rollouts per group. We train for 250 policy updates with rollout temperature 1.0, a constant learning rate of 10^{-6}, weight decay 0.01, and KL coefficient 0.01. The 1.5B and 7B runs use two and four GPUs, respectively, with tensor-parallel sizes of two and four. MCGPO uses the fixed preference-construction settings in Table 6. The Qwen3 extension uses action-only responses with thinking disabled. Resource settings, environment prompts, and an illustrative rollout appear in Appendices B, F, and G, respectively.

Table 1: Task success (%) on ALFWorld and graded score/success (%) on WebShop. All and Succ. are the primary metrics. GraphGPO and MCGPO report mean ± sample SD over three seeds using the evaluation protocol in Section 4.1. Results for the other methods are from Feng et al. (2025); Tan et al. (2026).

Type	Method	ALFWorld							WebShop	
		Pick	Look	Clean	Heat	Cool	Pick2	All	Score	Succ.
Closed-source models										
Prompting	GPT-4o	75.3	60.8	31.2	56.7	21.6	49.8	**48.0**	31.8	**23.7**
Prompting	Gemini-2.5-Pro	92.8	63.3	62.1	69.0	26.6	58.7	**60.3**	42.5	**35.9**
Base: Qwen2.5-1.5B-Instruct										
Prompting	Qwen2.5	5.9	5.5	3.3	9.7	4.2	0.0	**4.1**	23.1	**5.2**
Prompting	ReAct	17.4	20.5	15.7	6.2	7.7	2.0	**12.8**	40.1	**11.3**
Prompting	Reflexion	35.3	22.2	21.7	13.6	19.4	3.7	**21.8**	55.8	**21.9**
RL	PPO	64.8±3.5	40.5±6.9	57.1±4.9	60.6±6.6	46.4±4.0	47.4±1.9	**54.4**±3.1	73.8±3.0	**51.5**±2.9
RL	RLOO	88.3±3.0	52.8±8.6	71.0±5.9	62.8±8.7	66.4±5.5	56.9±4.7	**69.7**±2.5	73.9±5.6	**52.1**±6.7
RL	GRPO	85.3±1.5	53.7±8.0	84.5±6.8	78.2±7.9	59.7±5.0	53.5±5.6	**72.8**±3.6	75.8±3.5	**56.8**±3.8
RL	GiGPO$_{\text{w/std}}$	94.4±5.9	67.5±4.6	94.8±3.8	94.4±7.8	79.8±4.7	76.4±5.4	**86.7**±1.7	83.1±1.6	**65.0**±3.2
RL	GiGPO$_{\text{w/o std}}$	96.0±1.4	76.5±3.9	91.8±5.5	91.3±6.3	71.7±8.4	79.5±7.7	**86.1**±4.7	83.5±1.8	**67.4**±4.5
RL	HCAPO	88.6±7.0	75.0±0.0	**97.6**±1.8	90.7±6.9	84.2±0.0	74.2±6.9	**87.0**±4.1	83.8±0.7	**68.5**±1.0
RL	GRAPHGPO	96.1±1.2	79.5±4.7	96.8±2.8	92.9±4.1	93.6±1.6	**89.6**±5.4	**92.60**±1.93	**88.79**±1.26	**77.03**±1.22
RL	**MCGPO**	**96.5**±4.2	**84.6**±0.9	97.3±2.0	**95.8**±2.9	**94.6**±3.2	88.8±2.7	**93.65**±1.40	86.83±3.29	**78.13**±2.97
Base: Qwen2.5-7B-Instruct										
Prompting	Qwen2.5	33.4	21.6	19.3	6.9	2.8	3.2	**14.8**	26.4	**7.8**
Prompting	ReAct	48.5	35.4	34.3	13.2	18.2	17.6	**31.2**	46.2	**19.5**
Prompting	Reflexion	62.0	41.6	44.9	30.9	36.3	23.8	**42.7**	58.1	**28.8**
RL	PPO	92.3±4.0	64.0±8.4	92.5±2.4	89.5±7.0	80.3±2.0	68.8±8.3	**80.4**±2.7	81.4±3.1	**68.7**±5.1
RL	RLOO	87.6±4.3	78.2±8.3	87.3±5.8	81.3±7.6	71.9±5.2	48.9±8.4	**75.5**±4.6	80.3±3.2	**65.7**±4.0
RL	GRPO	90.8±5.1	66.1±6.7	89.3±5.4	74.7±6.9	72.5±5.4	64.7±7.3	**77.6**±5.2	79.3±2.8	**66.1**±3.7
RL	GiGPO$_{\text{w/std}}$	97.7±1.6	82.7±7.9	98.8±1.6	83.7±7.2	89.3±8.2	79.2±6.6	**90.8**±1.3	84.4±2.9	**72.8**±3.2
RL	GiGPO$_{\text{w/o std}}$	91.8±5.4	88.6±6.3	95.9±3.2	90.2±2.6	86.5±5.5	85.2±7.5	**90.2**±2.3	86.2±2.6	**75.2**±3.8
RL	HCAPO	99.1±1.3	90.3±2.0	97.3±1.9	81.8±8.8	90.8±6.6	81.9±10.0	**91.4**±2.3	85.1±1.3	**73.8**±2.8
RL	GRAPHGPO	99.2±1.5	91.6±5.6	**99.8**±0.4	90.7±1.1	90.2±5.2	92.2±6.6	**94.84**±2.11	90.23±0.66	**80.73**±1.52
RL	**MCGPO**	**99.8**±0.4	**98.6**±1.5	98.3±0.5	**93.1**±1.0	**94.0**±3.7	**92.3**±2.8	**96.30**±0.86	**90.28**±2.58	**81.20**±5.02

Evaluation. Every five updates, we sample one rollout for each of the same 128 validation tasks at temperature 0.4. To reduce sensitivity to individual checkpoints and sampled evaluation rollouts, our primary metric is mean success over the last five scheduled evaluations (L_5). All matched runs use the same end-of-budget window. We first average within seed and then report the mean and sample SD across seeds. Appendix C provides precise definitions, seed-level results, and summaries of learning progress.

4.2 RQ1: Does MCGPO Improve Task Completion and Learning Progress?

Table 1 reports ALFWorld and WebShop results, including published references, task-category success, and graded scores. Appendix C (Table 10) provides paired uncertainty estimates and complementary readouts for the matched runs.

Task completion. MCGPO improves mean success over matched GraphGPO in all four settings: by 1.04/1.46 percentage points on ALFWorld and 1.09/0.47 points on WebShop at 1.5B/7B. ALFWorld success rises from 92.60%/94.84% to 93.65%/96.30%, reducing the already-small remaining failure rate by approximately 14%/28%.

Where the gains appear. ALFWorld's largest category gains occur on Look (5.07/6.93 points at 1.5B/7B); Heat and Cool also improve at both sizes. On WebShop, success counts completed tasks, while the graded score also credits partial matches. At 1.5B, higher success accompanies a graded-score decrease from 88.79 to 86.83; at 7B, the scores are nearly equal.

Learning progress. Figure 3 shows an earlier rise in success for the 1.5B models. At update 100, the smoothed advantage is 7.60 points on WebShop and 3.54 on ALFWorld. A retrospective summary of the complete, unsmoothed curves integrates success over updates 0–250 and divides by 250 (Table 10). The resulting gains are 1.25/1.46 points on ALFWorld and 2.98/−0.02 on WebShop at 1.5B/7B; the first three settings improve in each of their three seeds. This comparison measures learning progress along the policy-update axis. We also compare task success at matched interaction budgets below. Appendix C reports uncertainty across the three training seeds; Appendix C.5 gives all four curves.

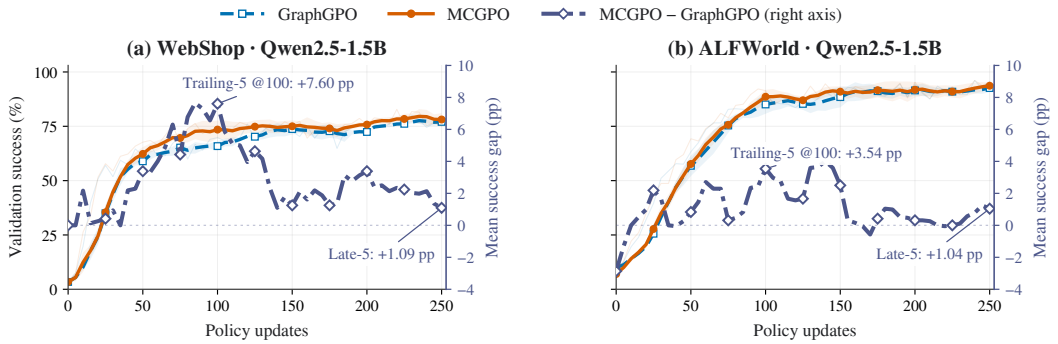

Figure 3: Learning curves and paired gains with Qwen2.5-1.5B. **Left axes:** validation success for MCGPO and GraphGPO; bands show the three-seed sample SD and faint lines retain raw means. **Right axes:** the deep-blue curve shows the mean paired difference (MCGPO minus GraphGPO), in percentage points. Each seed is averaged over up to five trailing evaluations, using the available evaluations at the beginning so curves start at update 0; complete five-point windows apply from update 20. The update-100 annotation uses the corresponding smoothed values. Shaded bands apply only to success curves.

Matched interaction budgets. We complement the update-axis curves with a retrospective budget comparison on Qwen2.5-1.5B. For each training seed, we select the scheduled checkpoint pair with the largest shared training-transition budget among pairs whose relative budget difference is below 1%. Selection uses budgets alone. Figure 5 in Appendix C.2 reports raw checkpoint success: MCGPO raises the mean from 92.97% to 95.57% on ALFWorld and from 76.04% to 79.43% on WebShop, gains of 2.60 and 3.39 points. WebShop improves in all three paired seeds; ALFWorld's differences vary across seeds. Appendix C.2 gives the selected checkpoints, counted interactions, and uncertainty.

4.3 RQ2: Which Parts of the Supervision Design Matter?

We test three design choices: task conditions for matching, the outcome evidence used to orient preferences, and the objective that transfers those preferences to the policy. Table 2(a) compares Full with four Qwen2.5-1.5B variants, using one training run per variant in each environment.

Comparison conditions and outcome evidence. *w/ Shuffled Rubric* permutes pre-action (C, F) labels among same-task decision rows, disrupting their alignment with actual progress. Success falls by 2.34 points on WebShop and 0.63 on ALFWorld. The *Candidate-progress labels* variant uses the candidate trajectories' progress evidence in place of opposite-fold terminal outcomes to label comparisons, reducing success by 1.41 and 3.13 points. This joint replacement changes evidence source, label content, and available support. Its result favors the complete terminal-outcome preference design over the candidate-progress alternative.

How the supervision enters training. *w/o Pairwise Loss* sets $\lambda_{\text{pair}} = 0$ while retaining preference diagnostics; its success drops by 1.41 and 0.78 points. *GraphGPO + Rubric Shaping* adds positive-progress rewards and regression penalties to the task rewards, retaining GraphGPO credit assignment with the pairwise loss disabled. This event-shaping baseline scores 8.28 and 2.66 points below Full. It compares two uses of the rubric channels under the stated shaping rule and coefficient. The three-seed

Table 2: Component ablations of MCGPO. (a) One Qwen2.5-1.5B run per configuration; parentheses give decreases from Full (pp). (b) Qwen3-4B/WebShop mean ± sample SD over three seeds. Entries are success rates (%) under the common evaluation protocol (Section 4.1); differences are computed before rounding. Protocols and paired analyses: Appendix C.3.

(a) Qwen2.5-1.5B: supervision controls (one run)

Component	Configuration	WebShop ↑	ALFWorld ↑
MCGPO (Full)		**78.13**	**94.53**
Match	w/ Shuffled Rubric	75.78 (↓2.34)	93.91 (↓0.63)
Label	Candidate-progress labels	76.72 (↓1.41)	91.41 (↓3.13)
Learn	w/o Pairwise Loss	76.72 (↓1.41)	93.75 (↓0.78)
	GraphGPO + Rubric Shaping	69.84 (↓8.28)	91.88 (↓2.66)

(b) Qwen3-4B / WebShop: 2×2 controls (three seeds)

Configuration	Opposite fold	Evidence weights	Success (%) ↑
A: MCGPO (Full)	Yes	Yes	$\mathbf{82.14} \pm 3.52$
B: Same-fold terminal	No	Yes	79.79 ± 3.37
C: Uniform weighting	Yes	No	80.52 ± 3.75
D: Same-fold + uniform	No	No	78.13 ± 3.36

Qwen3-4B/WebShop factorial controls in Table 2(b) show gains of 2.34–2.40 points from opposite-fold evidence and 1.61–1.67 points from evidence weighting, with all four contrasts positive in every seed. Appendix C.3 reports configurations and paired intervals. The Qwen3 experiments additionally compare MCGPO with GraphGPO + shaping under the same scorer access (Appendix C.3).

Comparison representation. Joint state–rubric conditioning gives the highest three-seed mean success in all four settings, exceeding State-only by 3.96–9.90 percentage points and Rubric-only by 5.15–9.01 points (Appendix C.4). The gains persist at the terminal evaluation in all four settings. With preference construction and the learning objective fixed, these gains support conditioning action comparisons and their terminal evidence jointly on state and progress. WebShop State-only already retains task-match score bins in its coarse state, placing this comparison within a scorer-assisted training setting.

4.4 RQ3: What Preferences and Weights Does the Construction Supply?

We examine preference construction on WebShop with Qwen2.5-1.5B, following supported comparisons through outcome-based labeling and evidence-weighted learning. Figure 4 covers all 750 updates from three runs: the matching resolution that supplies pairs, agreement with candidate outcomes, and the weight assigned to the evidence.

Matching supplies comparisons under sparse action support. Of 1,990 accepted preferences, 1,805 (90.70%) use coarse matching, which relaxes state detail while retaining rubric progress and dependency coordinates. Average supply declines from 3.99 pairs per update in updates 26–50 to 2.08 in 226–250, with coarse matching dominant throughout. The weaker-supported action has one evidence trajectory in 1,547 pairs (77.74%); the remaining 443 have two for each action.

Outcome evidence determines direction and strength. On WebShop, the preferred candidate succeeds while its alternative fails in 1,093 pairs; the reverse occurs in 124, and 773 have equal outcomes. This gives 74.35% pooled agreement when ties receive half credit. Every plotted window has a three-seed mean above 50%, showing a persistent association between preferences inferred from the evidence fold and outcomes in the separate candidate fold. The evidence fold's mean absolute outcome contrast is 0.781; accounting for action support and matching resolution yields a mean evidence weight of 0.208 (Figure 4c). Agreement measures association with candidate terminal outcomes, which also reflect subsequent actions.

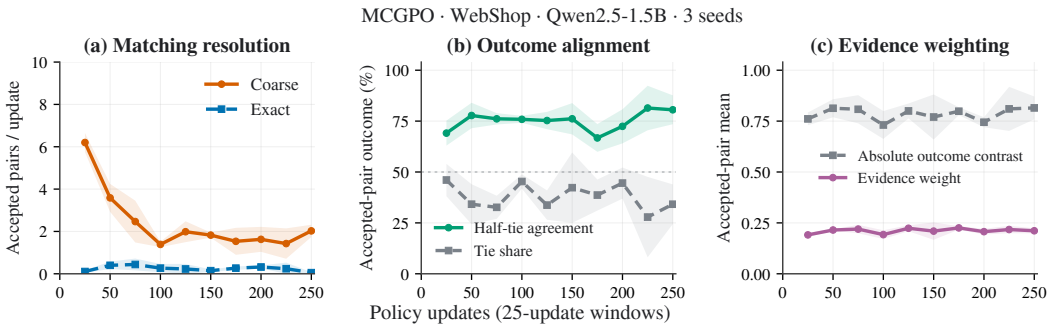

Figure 4: Preference construction on WebShop with Qwen2.5-1.5B. **(a)** Accepted pairs from exact and coarse matching. **(b)** Agreement of preference directions with candidate outcomes (ties receive half credit), together with the tie share. **(c)** Absolute outcome contrast and the resulting evidence weight, averaged over accepted pairs. Each point summarizes 25 updates; lines and bands show the three-seed mean and ± 1 sample SD. Ratios and pair-level means pool observations within each seed/window. Appendix C.5 defines the statistics and aggregation.

Connection to policy learning. The auxiliary loss is active in all 511 updates with accepted preferences (68.13% of training updates). RQ2 complements these supervision diagnostics with loss-removal, shuffled-rubric, and representation controls of policy performance.

4.5 RQ4: Do the WebShop Gains Extend to Qwen3 Backbones?

We extend the matched GraphGPO–MCGPO comparison to Qwen3-4B and 8B on WebShop with seeds 0, 1, and 2. These runs share the training budget and evaluation schedule and use action-only responses with thinking disabled.

Table 3: Task success on WebShop with Qwen3 backbones. Entries are mean $\pm$ sample SD over three training seeds under the evaluation protocol in Section 4.1. Gain is the paired mean difference in percentage points, computed before rounding.

Backbone	Mean success (%)		Gain (pp)
	GraphGPO	MCGPO	
Qwen3-4B	78.18 ± 2.62	$\mathbf{82.14} \pm 3.52$	+3.96
Qwen3-8B	79.69 ± 1.95	$\mathbf{84.84} \pm 1.96$	+5.16

MCGPO raises mean success from 78.18% to 82.14% at 4B and from 79.69% to 84.84% at 8B, gains of 3.96 and 5.16 percentage points, respectively. All six paired runs improve, with gains of 2.66–5.78 points at 4B and 2.50–9.22 points at 8B (Table 9). This consistency across seeds extends the evidence to both Qwen3 model sizes under action-only responses. A scorer-matched reward-shaping comparison appears in Appendix C.3.

5 Conclusion

MCGPO constructs action preferences by matching state and rubric progress within existing rollout groups. Opposite-fold terminal outcomes supply preference directions, and an evidence-weighted pairwise loss complements the GraphGPO task objective.

Across six environment–backbone settings, MCGPO improves three-seed mean success over matched GraphGPO, including ALFWorld baselines above 92% and Qwen3 WebShop gains of 3.96 and 5.16 points. Representation and factorial controls support joint context matching, opposite-fold evidence, and evidence weighting; scorer-matched comparisons favor MCGPO over reward shaping. Together, these results support extracting useful action-level supervision from existing rollout groups.

AI USE STATEMENT

Generative AI tools assisted with code development, experiment-log organization and analysis, manuscript drafting, and language editing. The authors are responsible for independently verifying all algorithmic claims, references, experimental results, and final manuscript content.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.662. URL https://aclanthology.org/2024.acl-long.662/.

Kevin Chen, Marco Cusumano-Towner, Brody Huval, Aleksei Petrenko, Jackson Hamburger, Vladlen Koltun, and Philipp Krähenbühl. Reinforcement learning for long-horizon interactive LLM agents. *arXiv preprint arXiv:2502.01600*, 2025. URL https://arxiv.org/abs/2502.01600.

Xin Cheng, Shuo He, Lang Feng, HaiYang Xu, Ming Yan, Lei Feng, and Bo An. Beyond trajectory-level attribution: Graph-based credit assignment for agentic reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2605.26684.

Yunho Choi, Jongwon Lim, Woojin Ahn, Minjae Oh, Jeonghoon Shim, and Yohan Jo. Your language model is its own critic: Reinforcement learning with value estimation from actor's internal states. *arXiv preprint arXiv:2605.07579*, 2026. URL https://arxiv.org/abs/2605.07579v2. Version 2.

Mingxuan Fan and Peiyang Liu. Progress- and reliability-oriented group policy optimization for agentic reinforcement learning. *arXiv preprint arXiv:2607.04242*, 2026. URL https://arxiv.org/abs/2607.04242.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2505.10978.

Shubham Gandhi, Saurabh Goyal, Kiran Kate, and Yara Rizk. DRACO: Fine-grained credit assignment with dynamic rubrics for long-horizon agent training. *arXiv preprint arXiv:2609.04094*, 2026. URL https://arxiv.org/abs/2609.04094.

Zhaolin Gao, Jonathan D. Chang, Wenhao Zhan, Owen Oertell, Gokul Swamy, Kianté Brantley, Thorsten Joachims, J. Andrew Bagnell, Jason D. Lee, and Wen Sun. REBEL: Reinforcement learning via regressing relative rewards. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2404.16767.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. *arXiv preprint arXiv:2507.17746*, 2025. URL https://arxiv.org/abs/2507.17746.

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

Joey Hejna, Rafael Rafailov, Harshit Sikchi, Chelsea Finn, Scott Niekum, W. Bradley Knox, and Dorsa Sadigh. Contrastive preference learning: Learning from human feedback without RL. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.13639.

Hongbo Jin, Rongpeng Zhu, Jiayu Ding, Guibo Luo, and Ge Li. HiMAC: Hierarchical macro-micro learning for long-horizon LLM agents. *arXiv preprint arXiv:2603.00977*, 2026. URL https://arxiv.org/abs/2603.00977v2. Version 2.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. VinePPO: Refining credit assignment in RL training of LLMs. In *Proceedings of the International Conference on Machine Learning*, 2025. URL https://arxiv.org/abs/2410.01679.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. Step-DPO: Step-wise preference optimization for long-chain reasoning of LLMs. *arXiv preprint arXiv:2406.18629*, 2024. URL https://arxiv.org/abs/2406.18629.

Gaotang Li, Bhavana Dalvi Mishra, Zifeng Wang, Jun Yan, Yanfei Chen, Chun-Liang Li, Long T. Le, Rujun Han, George Lee, Hanghang Tong, Chen-Yu Lee, and Tomas Pfister. RubricEM: Meta-RL with rubric-guided policy decomposition beyond verifiable rewards. *arXiv preprint arXiv:2605.10899*, 2026a. URL https://arxiv.org/abs/2605.10899.

Jiazheng Li, Yawei Wang, David Yan, Yijun Tian, Zhichao Xu, Huan Song, Panpan Xu, and Lin Lee Cheong. SALT: Step-level advantage assignment for long-horizon agents via trajectory graph. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 4709–4725. Association for Computational Linguistics, 2026b. doi: 10.18653/v1/2026.findings-eacl.247. URL https://aclanthology.org/2026.findings-eacl.247.pdf.

Yuanfan Li, Qi Zhou, Wenjing Duan, and Lu Chen. When denser credit is not enough: Evidence-calibrated policy optimization for long-horizon LLM agent training. *arXiv preprint arXiv:2606.05885*, 2026c. URL https://arxiv.org/abs/2606.05885v1. Version 1.

Yugu Li, Zehong Cao, Jianglin Qiao, and Siyi Hu. RTPO: Reverse-turn policy optimization for stabilizing agentic RL training. *arXiv preprint arXiv:2608.18682*, 2026d. URL https://arxiv.org/abs/2608.18682v3. Version 3.

Dayang Liang, Lang Feng, Bo An, and Yunlong Liu. SAPO: Single-rollout autoregressive policy optimization for agentic reinforcement learning. *arXiv preprint arXiv:2608.19842*, 2026. URL https://arxiv.org/abs/2608.19842v2. Version 2.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, and Abhinav Rastogi. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*, 2024. URL https://arxiv.org/abs/2406.06592.

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

Can Lv, Mingju Chen, Heng Chang, and Shiji Zhou. Mitigating false credit propagation: Probabilistic graphical reward aggregation for rubric-based reinforcement learning. *arXiv preprint arXiv:2606.03361*, 2026. URL https://arxiv.org/abs/2606.03361.

Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2405.14734.

Kun Peng, Conghui Tan, Yu Liu, Guohua Tang, Zhongqian Sun, Wei Yang, Zining Zhu, Lei Jiang, Yanbing Liu, and Hao Peng. Dialogue model optimization via agent game and adaptive tree-based GRPO. *arXiv preprint arXiv:2602.08533*, 2026. URL https://arxiv.org/abs/2602.08533.

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

Qiuyi Qi, Tian Liang, Mutian Bao, Jinjian Zhang, Dongnan Liu, Wei Zhou, Linjian Mo, Ming Kong, Jie Liu, Feng Zhang, and Qiang Zhu. STAPO: Selective trajectory-aware policy optimization for LLM agent training. *arXiv preprint arXiv:2607.04963*, 2026. URL https://arxiv.org/abs/2607.04963.

Bo Qian, Yuting Wu, Shuang Zeng, Huaiyu Wan, Dalin Zhang, and Jiqiang Liu. MileGPO: Milestone inference with local evidence for graph-based policy optimization of long-horizon LLM agents. *arXiv preprint arXiv:2608.19803*, 2026. URL https://arxiv.org/abs/2608.19803.

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 *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/a85b405ed65c6477a4fe8302b5e06ce7-Abstract-Conference.html.

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

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2010.03768.

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

Rodrigo Toro Icarte, Toryn Q. Klassen, Richard Valenzano, and Sheila A. McIlraith. Reward machines: Exploiting reward function structure in reinforcement learning. *Journal of Artificial Intelligence Research*, 73:173–208, 2022. doi: 10.1613/jair.1.12440. URL https://jair.org/index.php/jair/article/view/12440.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022. URL https://arxiv.org/abs/2211.14275.

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

Tao Wang, Suhang Zheng, and Xiaoxiao Xu. RTMC: Step-level credit assignment via rollout trees. *arXiv preprint arXiv:2604.11037*, 2026b. URL https://arxiv.org/abs/2604.11037v1. Version 1.

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

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, Yiping Lu, Kyunghyun Cho, Jiajun Wu, Li Fei-Fei, Lijuan Wang, Yejin Choi, and Manling Li. RAGEN: Understanding self-evolution in LLM agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*, 2025. URL https://arxiv.org/abs/2504.20073v2.

Zixuan Wang, Yuchen Yan, Hongxing Li, Teng Pan, Dingming Li, Ruiqing Zhang, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. Milestone-guided policy learning for long-horizon language agents. *arXiv preprint arXiv:2605.06078*, 2026d. URL https://arxiv.org/abs/2605.06078.

Weichu Xie, Haozhe Zhao, Wenpu Liu, Yongfu Zhu, Liang Chen, Minghao Ye, Zirong Chen, Yuqi Xu, Shuai Dong, Ziyue Wang, Xinbo Xu, Kean Shi, Ruoyu Wu, Xiaoying Zhang, Wenqi Shao, Baobao Chang, Nan Duan, and Jiaqi Wang. Step-wise rubric rewards for LLM reasoning. *arXiv preprint arXiv:2605.17291*, 2026. URL https://arxiv.org/abs/2605.17291.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/abs/2412.15115.

Kaibing Yang, Guangfeng Cai, Shengtian Yang, Shuo He, Yu Li, Mengyi Liu, Pengwei Chen, Jun Xu, and Lei Feng. Progress-conditioned group policy optimization for long-horizon agentic tasks. *arXiv preprint arXiv:2607.22724*, 2026. URL https://arxiv.org/abs/2607.22724.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, 2022. URL https://arxiv.org/abs/2207.01206.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025. URL https://arxiv.org/abs/2503.14476v2.

Huan Zhang, Mingju Chen, Dongxu Zhou, Can Lv, Heng Chang, Sen Cui, Faguo Wu, and Shiji Zhou. TRCA: Transition-wise rubric credit assignment for long-horizon LLM agents. *arXiv preprint arXiv:2608.16156*, 2026. URL https://arxiv.org/abs/2608.16156.

Yang Zhou, Sunzhu Li, Shunyu Liu, Wenkai Fang, Kongcheng Zhang, Jiale Zhao, Jingwen Yang, Yihe Zhou, Jianwei Lv, Tongya Zheng, Hengtong Lu, Wei Chen, Yan Xie, and Mingli Song. Breaking the exploration bottleneck: Rubric-scaffolded reinforcement learning for open-ended LLM reasoning. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*. PMLR, 2026. URL https://arxiv.org/pdf/2508.16949v7.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. ArCHer: Training language model agents via hierarchical multi-turn RL. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 62178–62209. PMLR, 2024. URL https://proceedings.mlr.press/v235/zhou24t.html.

A ALGORITHM AND COMPARISON CONSTRUCTION

This section specifies the comparison sets, evidence weights, and actor objective used in Section 3. Algorithm 1 summarizes the complete update. The same eight sibling trajectories supply the task objective and the auxiliary preferences.

Algorithm 1 MCGPO: Match–Label–Learn

Require: Policy π_θ, tasks $\mathcal{D}$, support m, contrast threshold ϵ, shrinkage κ, pair weight λ_{pair}.

for each training iteration **do**
  $\pi_{\text{old}} \leftarrow \pi_\theta$; $\mathcal{I}_0 \leftarrow \{0,2,4,6\}$, $\mathcal{I}_1 \leftarrow \{1,3,5,7\}$
  $\mathcal{T} \leftarrow \textsc{Collect}(\pi_{\text{old}}, \mathcal{D}, 8)$; $\mathcal{P} \leftarrow \emptyset$
  $A^{\text{task}} \leftarrow \textsc{TaskCredit}(\mathcal{T})$
  for each admitted task x and candidate fold $c \in \{0,1\}$ **do**
    $g_{i,t}^{(\ell)} \leftarrow (h^{(\ell)}(s_{i,t}), C_{i,t}, F_{i,t})$ ▷ Match
    $\mathcal{C}_c \leftarrow \textsc{FirstSupportedMatch}_{\text{exact}\to\text{coarse}}(\mathcal{I}_c, \mathcal{I}_{1-c}; g, m)$
    for each matched decision-row pair $p \in \mathcal{C}_c$ with context g, actions a, b, and level ℓ **do**
      $\widehat{u}_p \leftarrow \widehat{p}_{1-c}^{(\ell)}(g,a) - \widehat{p}_{1-c}^{(\ell)}(g,b)$ ▷ Label
      $n_p \leftarrow \min(n_a, n_b)$; $c_p \leftarrow |\widehat{u}_p|\, n_p/(n_p+\kappa)\, q_\ell$
    end for
    $\mathcal{V}_c \leftarrow \{(\textsc{Orient}(p, \operatorname{sign}\widehat{u}_p), c_p) : p \in \mathcal{C}_c,\ |\widehat{u}_p| \geq \epsilon\}$
    $\mathcal{P} \leftarrow \mathcal{P} \cup \textsc{Disjoint}_2(\textsc{EarliestQualified}(\mathcal{V}_c))$
  end for
  for each pair-aware actor minibatch $\mathcal{M}$ **do**
    Compute d_θ and $\mathcal{L}_{\text{pair}}$ on its microbatches using Eqs. 5–6 ▷ Learn
    Update θ with $\mathcal{L}_{\text{task}} + \lambda_{\text{pair}}\mathcal{L}_{\text{pair}}$ using A^{task} and $\mathcal{P}$
  end for
end for

Table 4 compares the task and preference interfaces with the related designs discussed in the main text.

Table 4: Comparison conditions, evidence, and policy interfaces. Rows summarize method designs; the matched performance evaluation in this paper uses GraphGPO. MCGPO's frontier is a rule-derived annotation of progress.

Method	Comparison conditions	Evidence and aggregation	Policy interface
GiGPO (Feng et al., 2025)	Repeated anchor states	Returns within trajectory and state groups	Trajectory and step advantages
HGPO (He et al., 2026)	Current state and historical-context hierarchy	Weighted advantages across context depths	Step advantage
BiPACE (Wang et al., 2026a)	Behavioral clusters from policy features	Action-conditioned peer returns	Step advantage
GRAPHGPO (Cheng et al., 2026)	Observed transition graph	Changes in distance to observed goals	Graph advantage
G^{2}PO (Wang et al., 2026c)	Identical observations in a global transition graph	Graph state values and globally standardized edge TD errors	Edge and episode advantages
AT-GRPO (Peng et al., 2026)	Dialogue trees with adaptive expansion depth	Returns over stage-dependent observation ranges	Group-relative policy loss
STAPO (Qi et al., 2026)	Anchor states and normalized-entropy outliers	Trajectory-aware rewards and trajectory-independent penalties	Selective step advantages
GEAR (Lv et al., 2026)	Dependencies among response criteria	Dependency-adjusted criterion probabilities	Scalar rubric reward
MCGPO	State and observed progress with rule annotation	Two-action support selects resolution; opposite-fold outcomes label pairs	Auxiliary weighted pairwise objective

A.1 COMPARISON KEYS AND FIRST VISITS

Each decision has a pre-action state summary h, active rubric criteria C, and rule annotation F. Boolean true and finite positive channel values activate membership in C; extractor diagnostics are

excluded. Appendix D.1 gives the environment-specific channels. For vocabulary V, prerequisites $P(j)$, blocking sources $B(j)$, and the set V_{block} of blocking-source nodes,

$$F(C; V, G_x) = \{j \in V \setminus (C \cup V_{\text{block}}) : P(j) \subseteq C,\ B(j) \cap C = \emptyset\}. \tag{8}$$

F annotates inactive criteria whose prerequisites are satisfied and which are not blocked. The opening status is empty; later records use the extractor schema. In ALFWorld, this vocabulary change can affect the first evidence row retained at the opening boundary.

Table 5: State fields for the ordered matching rules. Both rules retain the rubric signature (C, F).

Environment	Exact state	Coarse state
ALFWorld	Task type, location, held object, acquired/processed/placed object sets, light status	Task type, location, holding/processing flags, placement count, light status, remaining-budget bin
WebShop	Page, search query, product identity, selected options	Page, searched/product-open flags, binned type/attribute/option match, price satisfaction

Rollout slots define $\mathcal{I}_0 = \{0, 2, 4, 6\}$ and $\mathcal{I}_1 = \{1, 3, 5, 7\}$. For key g, each trajectory contributes its first eligible visit:

$$\begin{aligned} t_i^{(\ell)}(g; b) &= \min\{t \geq b : g_{i,t}^{(\ell)} = g\}, \\ \mathcal{G}_c^{(\ell)}(g; b) &= \{(i, t_i^{(\ell)}(g; b)) : i \in \mathcal{I}_c,\ t_i^{(\ell)}(g; b) < \infty\}. \end{aligned} \tag{9}$$

Candidate groups use $b = 1$, while evidence groups use $b = 0$. Candidates have different canonical actions. Opposite-fold first visits provide the support counts and empirical action-success rates used by the support gate in Eq. 2. The constructor retains the earliest qualified comparison per trajectory pair, then allocates up to two disjoint decision-row pairs per task and direction.

Table 6: Fixed preference-construction and learning settings.

Setting	Value
Sibling trajectories / fold sizes	8 / $4 + 4$
Minimum support for each action m	1
Terminal-score threshold for evidence	0.999
Outcome-contrast threshold ϵ	0.10
Support shrinkage κ	2
Resolution weights $(q_{\text{exact}}, q_{\text{coarse}})$	$(1, 0.70)$
Accepted pairs per task and direction	at most 2
Auxiliary-loss coefficient λ_{pair}	0.05

The four-trajectory evidence fold permits balanced action support $n_\star \in \{1, 2\}$. Its smallest nonzero binary-outcome contrast is $1/3$, so the configured threshold retains every non-tied supported contrast. Shrinkage contributes $1/3$ or $1/2$ at the two support levels. Invalid groups keep their task update with zero auxiliary weight.

A.2 Outcome provenance and pairwise update

Let D fix the admitted batch, rollout slots, keys, actions, turn indices, and hyperparameters. In candidate direction c,

$$\mathcal{P}_c(D; y_c, y_{1-c}) = \Phi_c(D, y_{1-c}). \tag{10}$$

Thus changing y_c leaves that direction's labels and weights unchanged under these fixed inputs. The reverse direction uses y_c as evidence. Batch admission uses whole-group outcomes and lies outside this conditional property. Finite-support contrasts remain observational estimates; the evidence weight controls their optimization contribution.

Local pairwise gradient. During an actor update, pair membership, orientation, the rollout policy, and weight c_p are fixed. With $\delta_\theta = d_\theta(+) - d_\theta(-)$ and $g_\theta^\pm = \nabla_\theta d_\theta(\pm)$, one pair contributes

$$\nabla_\theta \ell_p = -c_p\, \sigma(-\delta_\theta)(g_\theta^+ - g_\theta^-). \tag{11}$$

The actor colocates both rows in a microbatch, applies the normalization in Eq. 6, and adds the scaled auxiliary term to the task loss. RQ2 measures the policy effect of this addition.

A.3 SHARED TASK OBJECTIVE

The GraphGPO task branch uses $R_{i,t}^{\text{graph}} = 10\gamma^{D(s_{i,t+1})}$, where D is directed distance to an observed successful goal; unreachable states receive one more than the largest finite distance. Let $Z_{i,t} = \sum_k r_{i,t,k}^{\text{tok}}$ be the token-level task score, including invalid-action penalties. The advantage sums the normalized graph return within the task's anchor state group and the normalized task score within its response group:

$$\widehat{A}_{i,t}^{\text{task}} = \text{Norm}_{\mathcal{U}_{x,s_{i,t}}}(R_{i,t}^{\text{graph}}) + \text{Norm}_{\mathcal{U}_x}(Z_{i,t}). \tag{12}$$

Normalization uses the group mean and sample SD with 10^{-6} in the denominator. Singleton step groups have centered score zero; singleton task groups use mean zero and unit scale. Both task-credit coefficients are one, and each row advantage is broadcast to its valid response tokens.

For token ratio $\rho = \pi_\theta/\pi_{\text{old}}$, define $\ell_{\text{clip}} = \max\{-\rho A, -\text{clip}(\rho, 0.8, 1.2)A\}$ and $\ell_{\text{dc}} = \min\{\ell_{\text{clip}}, -3A\}$ when $A < 0$, with $\ell_{\text{dc}} = \ell_{\text{clip}}$ otherwise. The masked token mean $\langle\cdot\rangle_{\mathcal{B},m}$ gives

$$\mathcal{L}_{\text{task}} = \left\langle \ell_{\text{dc}}(\rho, \widehat{A}^{\text{task}}) + 0.01\,\widehat{K} - 0.001\,\mathcal{H}(\pi_\theta) \right\rangle_{\mathcal{B},m}, \tag{13}$$

where $\widehat{K} = \text{clip}(e^v - v - 1, -10, 10)$ and $v = \log \pi_{\text{ref}} - \log \pi_\theta$ on sampled tokens. The mask follows the response format in Appendix F. Gradient accumulation and distributed reduction use the shared actor configuration.

B EXPERIMENTAL SETUP

This section records the shared training recipe used by GraphGPO and MCGPO. The environment prompts appear in Appendix F. Comparisons within a backbone use the same task data, response protocol, optimizer, and evaluation grid.

B.1 TRAINING AND EVALUATION CONFIGURATION

Table 7 summarizes the environment and backbone profiles. Each update uses $P \times G$ admitted trajectories; shared dynamic group filtering can require additional generated groups when enabled.

Table 7: Training and rollout configuration for matched GraphGPO and MCGPO runs. Micro-batch sizes are reported per GPU. Mean–std denotes the shared GraphGPO within-group advantage normalization mode. TP denotes rollout tensor parallelism.

	ALFWorld		WebShop	
Setting	Small	Large	Small	Large
Qwen2.5 backbone	1.5B	7B	1.5B	7B
Qwen3 backbone	—	—	4B	8B
Task groups / update (P)	16	16	16	16
Trajectories / group (G)	8	8	8	8
Admitted trajectories / update	128	128	128	128
Training updates	250	250	250	250
Max. environment steps	50	50	15	15
Observation history (pairs)	2	2	2	2
Max. prompt length (tokens)	2048	2048	5120	5120
Max. response length (tokens)	512	512	512	512
PPO mini-batch	256	64	64	64
Actor micro-batch	32	4	8	4
Log-prob. micro-batch	32	16	16	16
GPUs	2	4	2	4
TP	2	4	2	4
GraphGPO normalization	Mean–std	Mean–std	Mean–std	Mean–std
Discount γ	0.10	0.10	0.20	0.20

Every setting trains for 250 policy updates with eight sibling rollouts per task, response length at most 512 tokens, rollout temperature 1.0, learning rate 10^{-6} with a constant schedule, and weight decay

0.01. Gradient checkpointing is enabled. The task objective's clipping and regularization coefficients are specified in Appendix A.3. Validation samples one rollout on each of the same 128 tasks every five updates at temperature 0.4.

Continuation and evaluation lineage. Main runs retain their archived $0 \to 150 \to 250$ protocol. Mechanism controls target 250 updates from initialization and use their corresponding Full reference seed (WebShop seed 0; ALFWorld seed 1). The ALFWorld controls use the corrected epoch-loop budget. Recovery restores model, optimizer, scheduler, random-number state, dataloader state, and the update counter. Reported scheduled evaluations follow the retained checkpoint lineage; reload evaluations and rolled-back branches are excluded. Controls are separate executions, so their recorded recovery histories can differ.

C Supplementary policy results and ablations

These results complement RQ1, RQ2, and RQ4 with seed-level readouts, uncertainty, and the control definitions. Success is reported in percent; paired gains use percentage points.

C.1 Readouts and seed-level results

Our primary metric summarizes late-training success over the final five scheduled evaluations of the shared 250-update budget. Averaging these evaluations reduces sensitivity to an individual checkpoint and its sampled rollouts. For each training seed, we compute

$$L_5 = \frac{1}{5} \sum_{t \in \{230,235,240,245,250\}} S(t). \tag{14}$$

Here $S(t)$ is validation success at update t. The same budget and end-of-budget window apply to all matched methods and seeds. We average within each seed before computing the mean and sample SD across seeds. Table 8 retains all designated Qwen2.5 seeds; Table 9 provides the three-seed WebShop extension underlying Table 3.

Table 8: Per-seed Qwen2.5 policy success (%). L_5 averages the final five scheduled evaluations of the 250-update budget; E_{250} reports the final evaluation.

Setting	Seed	MCGPO L_5	GraphGPO L_5	MCGPO E_{250}	GraphGPO E_{250}
ALFWorld / 1.5B	0	92.03	90.47	97.66	92.97
	1	94.53	93.13	94.53	96.88
	2	94.38	94.22	94.53	96.88
ALFWorld / 7B	0	95.31	96.88	96.88	95.31
	1	96.88	95.00	96.88	96.88
	2	96.72	92.66	94.53	92.97
WebShop / 1.5B	0	78.13	76.25	79.69	75.00
	1	81.09	78.44	78.91	77.34
	2	75.16	76.41	67.19	75.78
WebShop / 7B	0	84.84	79.06	86.72	83.59
	1	83.28	81.09	81.25	78.13
	2	75.47	82.03	72.66	78.91

Table 9: All paired Qwen3 WebShop seeds for the primary L_5 readout. Each entry averages updates 230, 235, 240, 245, and 250. Gains are computed before final display rounding; three-seed means appear in Table 3.

Backbone	Seed	GraphGPO (%)	MCGPO (%)	Gain (pp)
Qwen3-4B	0	79.84	85.63	+5.78
	1	75.16	78.59	+3.43
	2	79.53	82.19	+2.66
Qwen3-8B	0	81.25	85.00	+3.75
	1	77.50	86.72	+9.22
	2	80.31	82.81	+2.50

Learning progress and uncertainty. Per-seed endpoint values $E_{250} = S(250)$ are retained in Table 8 for completeness. Normalized integrated success summarizes learning progress over the complete raw evaluation grid $t_j = 5j$:

$$I_s = \frac{1}{250} \sum_{j=1}^{50} \frac{S_s(t_{j-1}) + S_s(t_j)}{2} (t_j - t_{j-1}). \tag{15}$$

Table 10 reports success and integrated success with paired intervals $\overline{\Delta} \pm 4.30265\, s_\Delta/\sqrt{3}$. The three training seeds determine uncertainty; evaluations along one training trajectory are dependent. The L_5 and integrated-success intervals span zero in all four Qwen2.5 settings. These intervals quantify the uncertainty accompanying the observed mean gains.

Table 10: Success and learning progress for GraphGPO and MCGPO across three training seeds. Entries give mean ± sample SD; both scores use a 0–100 scale. Δ is MCGPO minus GraphGPO in percentage points. Success follows Eq. 14; normalized raw-curve AUC summarizes learning progress. Intervals are 95% paired Student-t intervals ($\mathrm{df} = 2$); definitions appear in Appendix C.

Setting / readout	Success (%)	Normalized AUC (%)
ALFWorld / 1.5B		
GraphGPO	92.60 ± 1.93	77.42 ± 1.73
MCGPO	93.65 ± 1.40	78.68 ± 1.34
Paired Δ (pp)	+1.04	+1.25
95% paired interval	[−0.87, +2.96]	[−0.67, +3.18]
ALFWorld / 7B		
GraphGPO	94.84 ± 2.11	89.02 ± 1.42
MCGPO	96.30 ± 0.86	90.48 ± 0.87
Paired Δ (pp)	+1.46	+1.46
95% paired interval	[−5.59, +8.50]	[−0.21, +3.13]
WebShop / 1.5B		
GraphGPO	77.03 ± 1.22	66.00 ± 0.85
MCGPO	78.13 ± 2.97	68.98 ± 1.62
Paired Δ (pp)	+1.09	+2.98
95% paired interval	[−4.04, +6.23]	[−3.04, +8.99]
WebShop / 7B		
GraphGPO	80.73 ± 1.52	71.21 ± 1.15
MCGPO	81.20 ± 5.02	71.18 ± 1.12
Paired Δ (pp)	+0.47	−0.02
95% paired interval	[−15.30, +16.24]	[−4.68, +4.64]

C.2 Matched training-transition budgets at 1.5B

This retrospective diagnostic covers the twelve Qwen2.5-1.5B runs: MCGPO and GraphGPO on ALFWorld and WebShop, each with training seeds 0, 1, and 2. The effective configurations disable dynamic rollout filtering and collect $16 \times 8 = 128$ trajectories per update. The logged mean episode length $\overline{L}_{m,s,u}$ counts active environment steps per unique trajectory. Actor padding duplicates rows, while this metric deduplicates them by trajectory identity. We reconstruct cumulative training transitions as

$$B_{m,s}(t) = \sum_{u=1}^{t} \mathrm{round}\big(128\, \overline{L}_{m,s,u}\big)\,. \tag{16}$$

With 128 trajectories, the three-decimal logged mean uniquely determines the integer total; every reconstructed update passes the rounding check. We follow the retained checkpoint lineage across resumes. Validation, reload-only evaluation, rolled-back updates, and unfinished failed attempts are outside this training budget.

For each environment and seed, we enumerate the scheduled checkpoint pairs $t_M, t_G \in \{5, 10, \ldots, 250\}$ satisfying

$$\frac{|B_{M,s}(t_M) - B_{G,s}(t_G)|}{\min\{B_{M,s}(t_M), B_{G,s}(t_G)\}} < 0.01, \tag{17}$$

where M and G denote MCGPO and GraphGPO. We maximize the smaller of the two budgets, breaking ties by smaller relative mismatch and then larger combined update count. Selection uses the interaction counts alone. The plotted readout is raw validation success at the selected checkpoint, aggregated equally over the three training seeds. The budget labels average the paired costs across seeds. These checkpoint readouts are distinct from the primary L_5 metric.

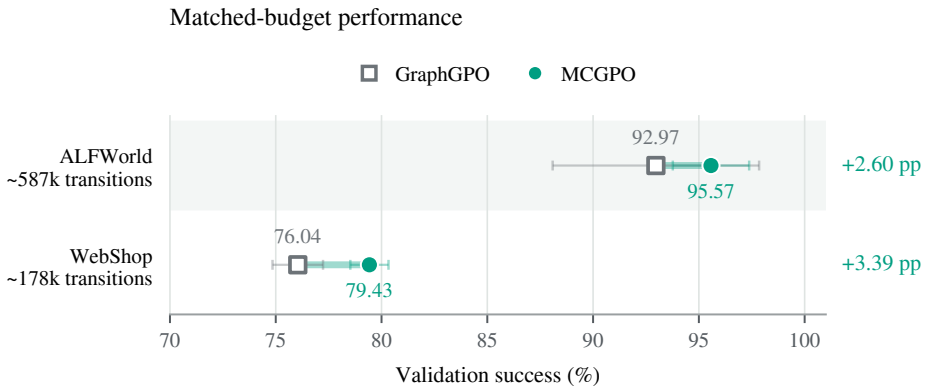

Figure 5: Qwen2.5-1.5B validation success at nearly matched training-transition budgets. Markers show three-seed means and whiskers sample SD. Each seed's checkpoint pair differs in budget by less than 1%; displayed budgets average the paired costs. Counts cover retained training interactions; Appendix C.2 specifies matching and restart accounting.

Table 11: Selected budget-matched checkpoints for Qwen2.5-1.5B. Slash-separated entries give GraphGPO/MCGPO. Gap is the relative budget difference in Eq. 17; Δ is MCGPO minus GraphGPO success in percentage points.

Env.	Seed	Updates	Transitions	Gap (%)	Success (%)	Δ (pp)
ALFWorld	0	230/250	594,465/592,472	0.336	87.50/97.66	+10.16
	1	250/250	575,983/581,151	0.897	96.88/94.53	-2.34
	2	245/250	590,739/585,967	0.814	94.53/94.53	0.00
WebShop	0	250/215	174,190/173,006	0.684	75.00/80.47	+5.47
	1	250/230	188,293/187,096	0.640	77.34/78.91	+1.56
	2	250/200	172,791/171,773	0.593	75.78/78.91	+3.13

Mean gains are 2.60 points on ALFWorld and 3.39 on WebShop. WebShop has three positive paired differences; ALFWorld has one positive, one zero, and one negative difference. Descriptive paired 95% Student-t intervals ($\mathrm{df} = 2$) are $[-13.90, 19.11]$ and $[-1.50, 8.27]$ points for ALFWorld and WebShop, respectively.

C.3 Mechanism controls

Table 2(a) reports one completed Qwen2.5-1.5B run per variant and environment. The candidate-progress control uses candidate trajectories' progress in place of opposite-fold terminal outcomes, jointly changing evidence source, labels, support, and weights. The loss-removal control sets $\lambda_{\mathrm{pair}} = 0$ while retaining construction diagnostics. These single-run comparisons evaluate the specified interventions; replication would estimate their variability.

The rubric-shaping control keeps GraphGPO credit and adds

$$\Delta r_t = 0.05\, N_t^{\mathrm{increase}} - 0.05\, N_t^{\mathrm{decrease}} \tag{18}$$

to the executed response's last valid token before advantage computation, with pairwise learning disabled. Counts include changes in graded and count-valued channels; a channel can increase more than once. This Qwen2.5 comparison evaluates the event-shaping rule at a fixed coefficient.

Factorial controls of evidence source and weighting. Table 2(b) crosses two factors on Qwen3-4B/WebShop: opposite-fold versus same-fold terminal evidence, and evidence-based versus uniform

loss weights. All four variants share initialization, training data, action-only prompts with thinking disabled, and 250 updates with 16 task groups of eight trajectories per update. We use seeds 0, 1, and 2, learning rate 10^{-6}, and fixed folds $\{0, 2, 4, 6\}$ and $\{1, 3, 5, 7\}$. The construction parameters in Table 6 and $\lambda_{\text{pair}} = 0.05$ are shared, without variant-specific tuning.

A (Full) and C use the four opposite-fold trajectories as terminal evidence; B and D use the four same-fold trajectories, including the candidates themselves. A and B retain the original evidence-weight formula. C and D assign equal loss weights to the selected preferences while preserving their original total weight. Matching rules, support thresholds, and pair-selection criteria otherwise remain fixed.

Table 12 reports individual seeds and paired contrasts. Opposite-fold evidence improves success under both evidence weighting ($A - B$: +2.34 pp) and uniform weighting ($C - D$: +2.40 pp). Evidence weighting improves success with both opposite-fold ($A - C$: +1.61 pp) and same-fold evidence ($B - D$: +1.67 pp). All four contrasts are positive in every seed, and their paired 95% intervals lie above zero. These results support each design choice under both settings of the other factor.

Table 12: Qwen3-4B WebShop factorial controls from Table 2(b). (a) Per-seed success (%), averaging the final five scheduled evaluations. (b) Paired gains in percentage points, with 95% Student-t intervals over three seed differences ($\mathrm{df} = 2$). Statistics precede display rounding.

(a) Per-seed task success (%)

Configuration	Seed 0	Seed 1	Seed 2
A: MCGPO (Full)	85.63	78.59	82.19
B: Same-fold terminal	83.28	76.56	79.53
C: Uniform weighting	84.22	76.72	80.63
D: Same-fold + uniform	81.56	74.84	77.97

(b) Paired component gains

Contrast	Mean gain (pp)	95% paired CI
$A - B$	+2.34	$[+1.57, +3.12]$
$A - C$	+1.61	$[+1.02, +2.21]$
$C - D$	+2.40	$[+1.28, +3.52]$
$B - D$	+1.67	$[+1.44, +1.89]$

Scorer-matched Qwen3 comparison. This experiment compares two ways of using the same scorer-derived supervision, testing whether the complete MCGPO preference-learning scheme improves on reward shaping. For Qwen3-4B/8B, GraphGPO + shaping uses all scorer information available to MCGPO (Appendix A.1) through training rewards. Table 13 reports the three-method comparison and paired gains across seeds 0, 1, and 2. MCGPO improves mean success over shaping by 2.29 points at 4B and 2.66 at 8B, with positive gains in all six paired runs. These results support the complete MCGPO scheme as a more effective use of the scorer signals than the compared reward-shaping alternative.

For differences d_s, the paired 95% Student-t interval is $\bar{d} \pm t_{0.975,2}\, s_d/\sqrt{3}$, where s_d is their sample SD. It summarizes training-seed variability under the paired-t approximation; both intervals lie above zero.

C.4 Representation controls

State-only, Rubric-only, and Full use the state coordinate, rubric coordinates, and their joint context, respectively, with the same preference construction and actor objective. Table 14 reports seeds 0, 1, and 2 in all four settings. Full has the highest mean under both readouts throughout, exceeding State-only by 3.96–9.90 and Rubric-only by 5.15–9.01 percentage points on the primary readout. Full and State-only produce different selected preference sets during training. Joint keys retain revisits after progress changes and condition candidate actions and opposite-fold evidence on both state and progress. State-only can merge different progress stages; Rubric-only can merge different environmental situations. The gains support joint context for selecting the auxiliary learning signal. WebShop State-only already includes task-match score bins, so this control compares how structured training information is used. Appendix D illustrates visit retention and support in a separate diagnostic sample.

Table 13: WebShop comparison with a scorer-matched shaping control. (a) Task success (%), reported as mean ± sample SD over three seeds. (b) MCGPO gains over GraphGPO + shaping, in percentage points. Paired 95% Student-t intervals ($\mathrm{df} = 2$) use the per-seed differences shown.

(a) Task success (%)

Method	Qwen3-4B	Qwen3-8B
GraphGPO	78.18 ± 2.62	79.69 ± 1.95
GraphGPO + shaping	79.85 ± 3.21	82.19 ± 1.44
MCGPO	$\mathbf{82.14} \pm 3.52$	$\mathbf{84.84} \pm 1.96$

(b) Paired gains over GraphGPO + shaping (pp)

Backbone	Seed 0	Seed 1	Seed 2	Mean gain	95% paired CI
Qwen3-4B	+2.50	+1.87	+2.50	+2.29	$[+1.39, +3.19]$
Qwen3-8B	+2.50	+3.28	+2.19	+2.66	$[+1.26, +4.05]$

Table 14: Supplementary representation controls at 250 updates. Full, State-only, and Rubric-only use seeds 0, 1, and 2 in every setting. Entries give mean success (%) and sample SD across the three seeds. L_5 averages the final five scheduled evaluations, and E_{250} is the terminal evaluation. State-only retains the environment-specific fields of its state abstraction, including WebShop's match-score bins.

Setting	Comparison key	n	L_5	E_{250}
ALFWorld / 1.5B	MCGPO	3	93.65 ± 1.40	95.57 ± 1.80
	State-only	3	85.57 ± 2.32	83.85 ± 1.97
	Rubric-only	3	87.81 ± 1.59	87.76 ± 1.80
ALFWorld / 7B	MCGPO	3	96.30 ± 0.86	96.09 ± 1.35
	State-only	3	92.34 ± 1.25	91.93 ± 1.97
	Rubric-only	3	91.15 ± 1.83	90.36 ± 1.97
WebShop / 1.5B	MCGPO	3	78.13 ± 2.97	75.26 ± 7.00
	State-only	3	72.50 ± 1.25	73.18 ± 1.19
	Rubric-only	3	70.42 ± 0.94	72.40 ± 2.96
WebShop / 7B	MCGPO	3	81.20 ± 5.02	80.21 ± 7.09
	State-only	3	71.30 ± 4.48	69.27 ± 7.26
	Rubric-only	3	72.19 ± 2.99	70.57 ± 2.26

C.5 Learning curves and supervision dynamics

This subsection supplies the full curves referenced by RQ1 and the aggregation definitions for RQ3. Figures 6 and 7 describe policy learning and interaction length over the same retained training runs.

Training behavior. The ALFWorld analysis includes all 750 updates from three completed Full runs. Windows 26–50, 101–125, and 226–250 contain 25 updates per seed. Mean training success rises from 60.3% to 89.9% and 95.3%, while mean rollout length falls from 29.34 to 13.91 and 10.17 steps. Length counts active interactions per distinct trajectory, including successful, failed, and truncated runs under the 50-step cap. Mean anchor-state group size rises from 4.10 to 5.54 and 6.62; it counts decision rows sharing task and observation, including repeated visits. These are joint training trends; component effects are assessed in RQ2.

Preference supply and agreement. Figure 4 uses all 750 WebShop 1.5B updates in nonoverlapping 25-update windows. Each seed contributes equally to the plotted means and sample SD. Counts include zero-pair updates; outcome ratios pool W, T, L within each seed/window before aggregation. The pooled 74.35% agreement instead weights all 1,990 accepted pairs. Coarse matching supplies 1,805 pairs (90.70%).

Weights and entry into learning. Evidence contrast and weight means first weight updates by accepted-pair count within each seed/window. Their pooled means are 0.781 and 0.208.

Among all pairs, 1,547 have balanced evidence support one and 443 have support two. The weight combines outcome contrast, support shrinkage, and matching resolution. TensorBoard values matched to the retained process and update confirm nonzero pairwise loss in all 511 updates with accepted

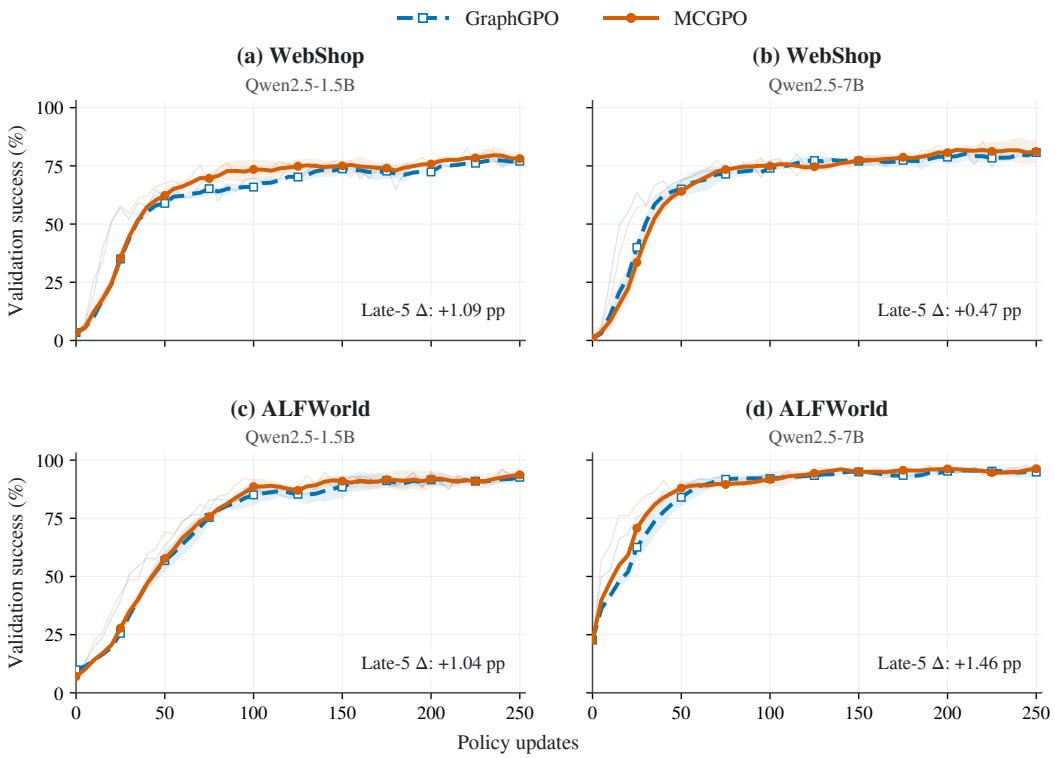

Figure 6: Validation success in all four Qwen2.5 settings. Each seed uses up to five trailing scheduled evaluations, including all available points at the beginning. Lines and bands give the three-seed mean and sample SD; faint lines retain raw means. The fixed L_5 window is defined separately in Eq. 14.

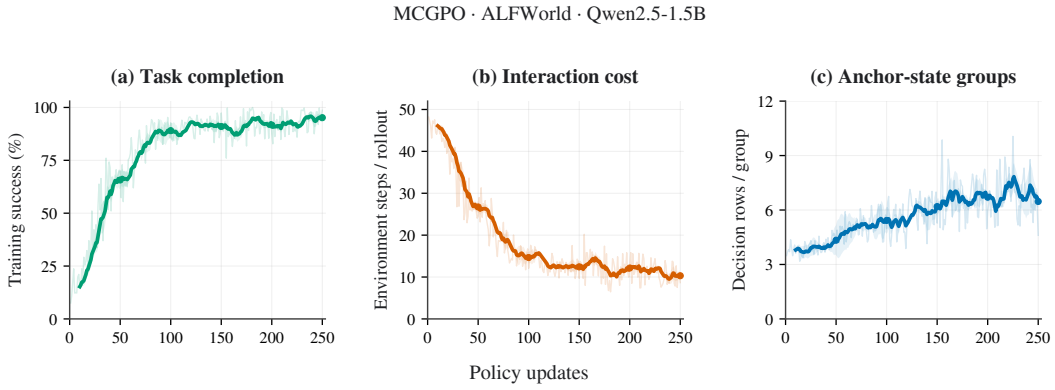

Figure 7: ALFWorld Qwen2.5-1.5B training behavior: (a) rollout success, (b) environment steps per rollout, and (c) anchor-state group size in the task-credit branch. Dark lines average ten trailing updates within each seed, then report the three-seed mean; bands show sample SD and faint lines show seed means.

pairs; all 239 empty-pair updates have zero loss. This establishes objective activation, complementing the loss-removal control in RQ2. All these counts describe the constructor's selected output. Source logs retain the exact count identities and checkpoint lineage.

D RUBRIC, MATCHING, AND PREFERENCE DIAGNOSTICS

This section specifies the comparison keys used in RQ2 and supplies diagnostics for RQ3. The rubric specification defines recorded task progress; archived examples illustrate visit retention and matching support, and WebShop outcome diagnostics characterize selected preferences.

D.1 Environment rubric rules

Extractors run after each transition. The next decision uses that transition's status as its pre-action C and F, following the opening convention in Appendix A.1. ALFWorld reads task metadata and tracks target-object sets $\mathcal{A}, \mathcal{P}, \mathcal{L}$ for acquisition, processing, and placement. The required count is $r = 2$ for Pick2 and $r = 1$ otherwise. Successful take, process, and placement feedback updates these sets; entity matching preserves object-instance numbers.

Table 15: ALFWorld channel definitions. Symbols identify the nodes in Table 17; $1[\cdot]$ denotes an indicator. The sets and required object count r are defined in the text.

Channel (symbol)	Status after the observed transition
`target_acquired` (A)	$\min(r, \lvert\mathcal{A}\rvert)$; acquisition history persists after placement.
`target_in_hand` (H)	Whether the worker's held object matches the target type.
`target_processed` (P)	$1[\mathcal{P} \neq \emptyset]$; processing history persists after release.
`target_placed` (L)	$\min(r, \lvert\mathcal{L}\rvert)$; taking or relocating an object updates this set.
`light_on` (B)	Tracked target-light state: successful off commands/feedback set false; on, use, or toggle set true unless off is indicated.
`goal_progress` (G)	Look: $H + B$. Clean/Heat/Cool: $1[\mathcal{P} \neq \emptyset] + 1[\mathcal{L} \neq \emptyset] + 1[\mathcal{P} \cap \mathcal{L} \neq \emptyset]$. Pick/Pick2: $\min(r, \lvert\mathcal{L}\rvert)$.
`solved` (S)	The environment's `won` flag.

WebShop reads browser-session fields and its existing goal scorer. Product-match channels therefore use additional structured training supervision, including scorer evaluation before purchase. Reusing rollouts saves trajectory collection while retaining this information requirement. The extractor's task contract omits the target product identifier; the environment scorer retains its goal metadata. The adapters use deterministic rules and the benchmark scorer. With matched/requested attribute and option counts M_a, M_o, N_a, N_o, type score u, and price predicate p, purchase readiness is

$$q_{\text{preview}} = u \, \frac{M_a + M_o + p}{N_a + N_o + 1}. \tag{19}$$

Absent components and caught scoring errors retain initialized zeros. This includes an omitted option component for a no-option goal. Terminal evidence uses $y = 1[\text{score} \geq 0.999]$.

Table 16: WebShop channel definitions. Symbols refer to WebShop nodes in Table 17. Match components come from the environment goal scorer; default handling is specified in the text.

Channel (symbol)	Status after the observed transition
`search_issued` (Q)	Whether the session has search keywords or its searched flag is true.
`product_opened` (O)	Whether a product is selected or the product-open flag is true.
`type_match` (T)	The scorer's type-match component `r_type`.
`attribute_match` (A)	Requested-attribute match fraction; scorer `r_att` is stored as `r_attr`.
`option_match` (U)	Requested-option match fraction `r_option`; the standard worker retains zero when the scorer omits this component.
`price_ok` (P)	The scorer's Boolean price predicate `r_price`.
`purchase_readiness` (R)	Preview score of the selected product/options, computed before purchase using Eq. 19.
`purchased` (B)	The environment's `done` flag.
`task_score` (K)	Raw environment score clipped to $[0, 1]$, before binary reward remapping.
`solved` (S)	True exactly when the environment terminates with raw score 1.

Table 17 lists the task-rule graphs. All non-blocking edges supply the prerequisite tests in Eq. 8; the text-environment graphs have no blocking edges. ALFWorld's coarse budget bin partitions the remaining horizon at $1/4, 1/2, 3/4$. WebShop's score bins use $\text{round}_{\text{even}}(4 \ \text{clip}(v, 0, 1))$. Action keys preserve canonical ALFWorld commands; WebShop uses navigation/purchase markers and goal-aware search, product, and option keys from the pre-action page, with goal-token overlap thresholds $0.15, 0.35, 0.60$. Missing required trace fields disable auxiliary supervision while retaining the task update. Progress-measurement errors can change groups and action support; extractor precision and recall against independent annotations remain unmeasured.

Table 17: Complete dependency templates. Symbols use the respective environment's channel table. E denotes `requires_ever` and C denotes `requires_current`. Each ALFWorld family includes the common edge; set notation expands to one edge per element.

Environment/task	Relation	Directed edges
ALFWorld: all	E	$A \to H$
Pick / Pick2	E	$A \to L$
	C	$L \to G, G \to S$
Clean / Heat / Cool	E	$A \to P, A \to L$
	C	$P \to L, P \to G, L \to G, G \to S$
Look	C	$H \to G, B \to G, H \to S, B \to S, G \to S$
WebShop: all	E	$Q \to O$
	C	$O \to \{T, A, U, P\}$
	C	$\{T, A, U, P\} \to R$
	C	$R \to B, B \to \{K, S\}$

D.2 Matching support and a captured example

The census uses ten archived update-60 captures: 160 task instances and 1,280 trajectories across the four text settings. These captures are a separate diagnostic sample from the training runs in Table 14. Each candidate fold contains four trajectories; groups count first eligible visits per trajectory and key. Coverage divides records in groups of size at least two by all retained records at that resolution. Figure 8 reports the distribution over groups; Table 18 reports coverage over records and counts supported pairs. Both precede outcome filtering.

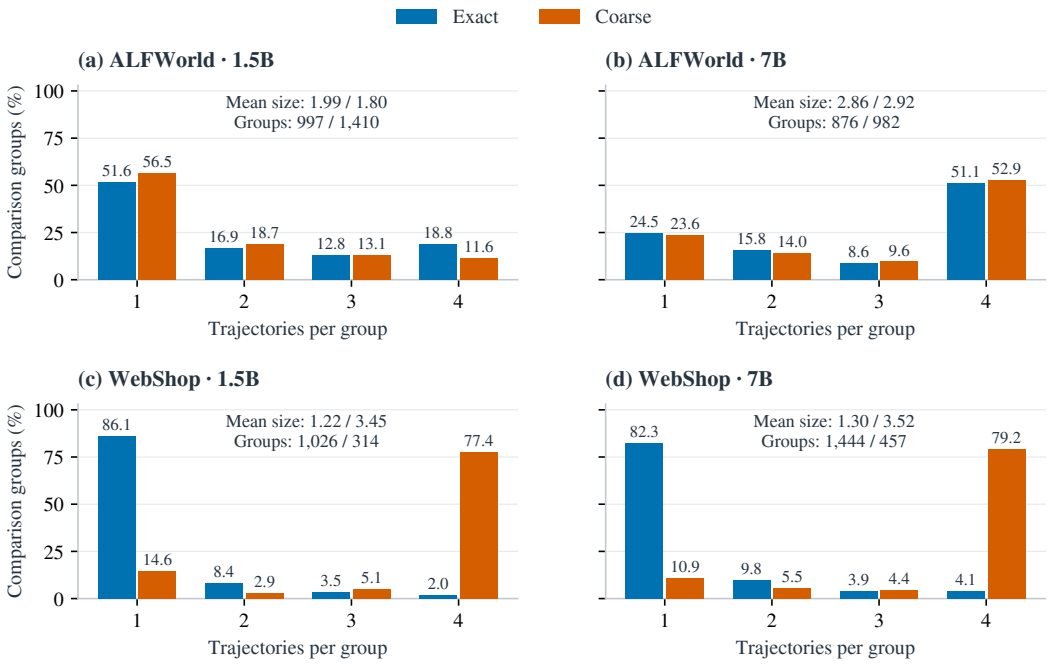

Figure 8: First-visit group sizes across ten update-60 captures (1.5B: seeds 1/2; 7B: seeds 0/1/2, in both environments). Exact and coarse rules retain the same rubric coordinates; groups count distinct contributors within a four-trajectory candidate fold. Bars include singletons, and annotations report mean sizes and group counts.

Exact and coarse are evaluated separately, so supported-pair counts can overlap. On these archived snapshots, Full replay reproduces the stored pairs and directions; State-only selects the same pairs, with no accepted pairs in the WebShop subset after outcome filtering. These statistics characterize local matching availability. Figure 9 illustrates how progress preserves a revisit as a distinct comparison context; Table 14 assesses the benefit of applying joint context throughout training.

Table 18: Single-resolution comparison availability in all ten captured update-60 batches. Non-singleton coverage is the fraction of first-visit records in groups with at least two contributors. Supported pairs have distinct actions and at least one opposite-fold observation of each action. These counts precede outcome-contrast filtering and the final quota; exact and coarse counts overlap.

	Non-singleton records (%)		Supported row pairs	
Setting	Exact	Coarse	Exact	Coarse
ALFWorld / 1.5B	74.07	68.57	143	133
ALFWorld / 7B	91.43	91.90	86	252
WebShop / 1.5B	29.19	95.76	2	19
WebShop / 7B	36.61	96.89	14	79

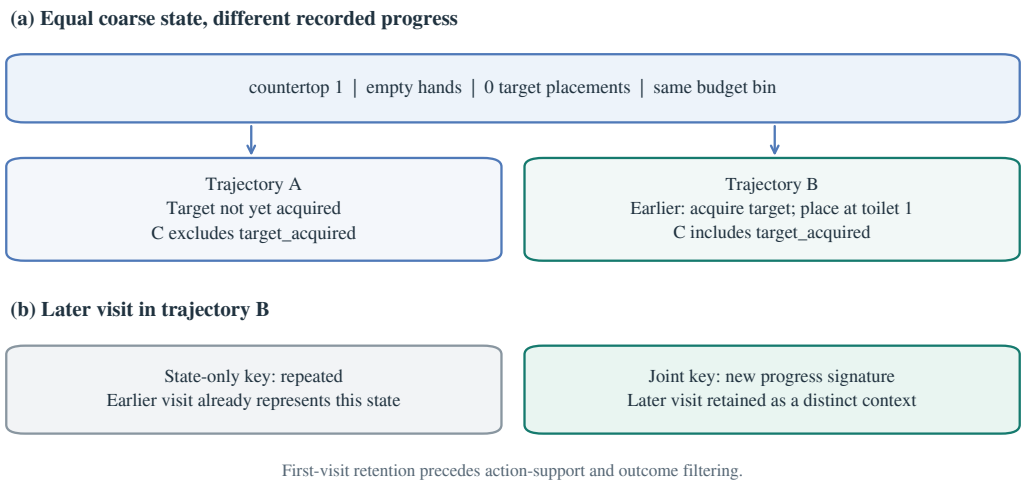

Figure 9: Captured ALFWorld example: *put two toiletpaper in drawer*. Two trajectories reach countertop 1 with empty hands, zero target placements, and the same budget bin. One previously acquired toiletpaper 2 and placed it at toilet 1. Its changed progress signature retains the later visit as a distinct context under the joint first-visit rule, before preference selection.

D.3 WebShop agreement with candidate outcomes

Table 19: Archived WebShop training-trace diagnostics for Full, pooled over three seeds per Qwen2.5 backbone. $W/T/L$ denotes counts of preferred-candidate wins, ties, and reversed outcomes; agreement is $(W + T/2)/(W + T + L)$. Active gives accepted-pair updates / audited updates; Exact is the fraction of accepted pairs using exact matching. All counts refer to the retained trace sample.

Setting	Pairs	$W/T/L$	Agreement (%)	Active	Exact (%)
WebShop / 1.5B	1428	771/568/89	73.88	330/450	8.3
WebShop / 7B	1425	633/693/99	68.74	319/450	20.1

Table 19 pools WebShop $W/T/L$ counts of 1404/1261/188. Agreement measures association with candidate terminal outcomes after preference construction; RQ2 supplies the policy controls. These archived samples contain repeated training observations; the complete 250-update WebShop 1.5B diagnostics and objective-activation check appear in Appendix C.5.

E Limitations

MCGPO is evaluated on ALFWorld and WebShop with Qwen backbones. Transfer to other domains requires suitable state summaries, rubric channels, and task rules; WebShop also uses structured goal-scorer information during training. Preferences are inferred from finite groups of observational rollouts: coarse matching can retain outcome-relevant differences, and terminal success depends on subsequent decisions as well as the matched action. Evidence weighting moderates the influence of sparse comparisons, while direct validation of local action preferences remains an important direction.

Our main comparisons use three training seeds and repeated evaluation on a fixed validation set, so small gains retain statistical uncertainty. Evaluation with broader model families, more seeds, and independent test tasks would further assess generalization and statistical precision.

F Environment Prompts

GraphGPO and MCGPO use the same environment templates within each backbone setting. Each filled template forms a user message in the backbone's chat template; rubric coordinates are computed by the training adapters. The boxes below give the interaction prompts, including their response instructions. Only line wrapping, trailing whitespace, and the typographic hyphen in *e-commerce* are normalized.

Placeholders in braces are filled from the current environment state. The first decision omits the history block; later decisions include up to the two most recent observation–action pairs, as specified in Table 7. Each history entry has the form `[Observation n: 'observation', Action n: 'action']`. `step_count` counts completed actions and `current_step=step_count+1`. A recorded ALFWorld example appears in Appendix G.

F.1 ALFWorld

ALFWorld provides the current text observation and admissible-action set. The task appears in the reset observation and is inserted explicitly as `{task_description}` at subsequent decisions. The admissible-action list excludes the help command. Both Qwen2.5 backbones use the following reasoning-and-action prompts.

ALFWorld Interaction Prompt: first decision

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.

ALFWorld Interaction Prompt: subsequent decisions

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.

F.2 WebShop

The prompt includes the shopping goal, rendered page, and available search/click actions. A search bar contributes `search[<your query>]`; clickable elements contribute `click[element]`. If a history-containing prompt exceeds 13,000 characters, the implementation falls back to the no-history context while retaining the current goal and observation. The following complete prompts apply to Qwen2.5-1.5B/7B.

WebShop Interaction Prompt: first decision (Qwen2.5)

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.

WebShop Interaction Prompt: subsequent decisions (Qwen2.5)

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.

Qwen3-4B/8B uses the same task, observation, admissible-action, and history fields with an action-only response instruction. The complete first-decision and subsequent-decision templates follow. Both use the backbone's chat template with `enable_thinking=False`.

WebShop Interaction Prompt: first decision (Qwen3)

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.
Choose exactly one action from the admissible−action list. Respond with exactly one complete <action>selected action</action> block and no other text.

WebShop Interaction Prompt: subsequent decisions (Qwen3)

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}

WebShop Interaction Prompt: subsequent decisions (Qwen3) (continued)

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.
Choose exactly one action from the admissible–action list. Respond with exactly one complete <action>selected action</action> block and no other text.

G Illustrative Task and Rollout Trace

This complete ALFWorld rollout uses MCGPO with Qwen2.5-1.5B-Instruct (seed 2, update 60, sibling slot 4, zero-based). The task requires placing two distinct objects into a drawer. We select the shortest successful sibling of this previously audited task for compactness. It illustrates the interaction protocol, not a representative performance estimate or a matched preference pair.

ALFWorld task: recorded reset observation

–= Welcome to TextWorld, ALFRED! =–

You are in the middle of a room. Looking quickly around you, you see a countertop 1, a drawer 8, a drawer 7, a drawer 6, a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1, a garbagecan 1, a handtowelholder 1, a sinkbasin 2, a sinkbasin 1, a toilet 1, a toiletpaperhanger 1, and a towelholder 1.

Your task is to: put two toiletpaper in drawer.

This reset observation fills the initial template in Appendix F; later decisions use the latest observation, admissible actions, and up to two preceding observation–action pairs. The box preserves all 12 parsed environment commands, including the repeated pickup and replacement at steps 6–7 and the sink visit at step 8. Feedback is verbatim, omitting appended location, inventory, move-history, and action-list annotations. Nonterminal feedback enters the next observation; success is the literal terminal feedback.

Complete rollout: put two toiletpaper in drawer

Step 1. Action: go to countertop 1
Feedback: You arrive at countertop 1. On the countertop 1, you see a cloth 1, a soapbottle 1, a spraybottle 1, a toiletpaper 3, and a toiletpaper 2.

Step 2. Action: take toiletpaper 2 from countertop 1
Feedback: You pick up the toiletpaper 2 from the countertop 1.

Step 3. Action: go to drawer 1
Feedback: You arrive at drawer 1. The drawer 1 is closed.

Step 4. Action: open drawer 1
Feedback: You open the drawer 1. The drawer 1 is open. In it, you see nothing.

Step 5. Action: move toiletpaper 2 to drawer 1
Feedback: You move the toiletpaper 2 to the drawer 1.

Step 6. Action: take toiletpaper 2 from drawer 1
Feedback: You pick up the toiletpaper 2 from the drawer 1.

Step 7. Action: move toiletpaper 2 to drawer 1
Feedback: You move the toiletpaper 2 to the drawer 1.

Step 8. Action: go to sinkbasin 1
Feedback: You arrive at sinkbasin 1. On the sinkbasin 1, you see a sink 1, and a soapbar 2.

Step 9. Action: go to toilet 1
Feedback: You arrive at toilet 1. On the toilet 1, you see a candle 1, a soapbar 3, and a toiletpaper 4.

Complete rollout: put two toiletpaper in drawer (continued)

Step 10. Action: take toiletpaper 4 from toilet 1
Feedback: You pick up the toiletpaper 4 from the toilet 1.

Step 11. Action: go to drawer 1
Feedback: You arrive at drawer 1. The drawer 1 is open. In it, you see a toiletpaper 2.

Step 12. Action: move toiletpaper 4 to drawer 1
Feedback: success

Feedback and outcome. Steps 1–11 receive reward 0; step 12 receives reward 10 and terminal success label $Y = 1$. Post-action placement counts are 1, 0, 1, and 2 at steps 5, 6, 7, and 12, respectively. The acquisition count stays at one while toiletpaper 2 is handled repeatedly, then reaches two when toiletpaper 4 is acquired at step 10. Matching uses progress available before the current action.