Grounding Agent Reasoning with Structured Process Supervision for Multi-turn Reinforcement Learning

Renting Rui
Shanghai Jiao Tong University

Yulei Qin
Tencent Youtu Lab

Weiwen Liu
Shanghai Jiao Tong University

Yunjia Xi
Shanghai Jiao Tong University

Ke Li
Tencent Youtu Lab

Xing Sun
Tencent Youtu Lab

Weinan Zhang*
Shanghai Jiao Tong University

Yong Yu*
Shanghai Jiao Tong University

Abstract

Long-horizon LLM agents trained with outcome-based RL exhibit a systematic failure mode: reasoning diverges from environmental observations at a certain step, and the error accumulates across subsequent turns. We find that 74% of failed trajectories contain reasoning–observation inconsistency, and among failed episodes with an inconsistency, 60% of turns in the 15-turn window starting at its first occurrence are inconsistent, indicating that agents rarely self-correct once they lose track of the environment. Process supervision is a natural remedy, but its effectiveness depends on having a well-defined target to evaluate. Under free-form reasoning, a judge has no structured anchor: the agent can produce tokens broadly compatible with prior observations without ever committing to a specific claim about task progress or the expected outcome of the next action. We propose **ARC** (**A**nchored **R**easoning with **C**ommitment), which requires the agent to explicitly reason before each action, *what has been established so far* and *what the environment should look like after the next action*—two commitments that are directly verifiable against interaction history and the subsequent observation respectively. This structure anchors the agent's reasoning to the observed environment state across turns and provides well-defined targets for process supervision. To prevent interference between sparse outcome and dense process signals, we decouple their advantages via independent group normalization. On ALFWorld, WebShop, and WebArena, ARC improves over the strongest baseline by 6.4, 3.5, and 9.6 points respectively.

1 Introduction

Large language model-based agents are increasingly deployed in interactive environments requiring long-horizon decision-making, such as web navigation[Zhou et al., 2023], software engineering[Jimenez et al., 2023], computer use[Xie et al., 2024], and embodied control[Driess et al., 2023]. These settings share three characteristics: long horizons, partial observability and sparse supervision in the form of terminal-only rewards[Sumers et al., 2023]. Under such conditions, agents typically follow the ReAct paradigm which alternates between reasoning and actions conditioned on environmental observations[Yao et al., 2022b]. The agentic reasoning in multi-turn settings maintains awareness of past interactions and serves as the immediate context for the next action[Wei et al., 2026].

*Corresponding authors: Weinan Zhang (wnzhang@sjtu.edu.cn) and Yong Yu (yyu@apex.sjtu.edu.cn).

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

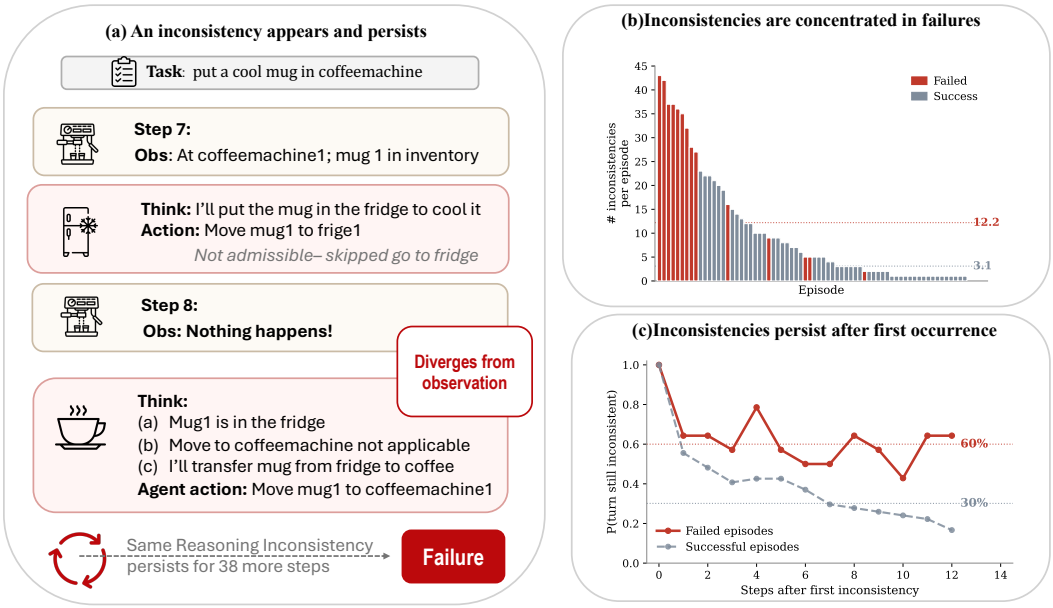


Figure 1: **Reasoning inconsistency is a key failure mode in long-horizon agents.** (a) A GRPO-trained agent treats the "Nothing happens" feedback at Step 8 as if the failed action had succeeded, forming the false belief that "Mug1 is in the fridge." The inconsistency persists for 38 more steps and leads to failure. (b) Failed episodes contain $4\times$ more inconsistencies than successful ones. (c) Once an inconsistency appears, it persists much more often in failed episodes than in successful ones.

Current approaches train agents with outcome-based RL such as GRPO[Shao et al., 2024], assigning rewards only at the terminal step. This paradigm has proven effective for mathematical reasoning, in which the model generates a self-contained chain of thought without external feedback. Multi-turn agent tasks differ in a critical respect: at each step the environment returns observations that the agent must integrate into its ongoing reasoning. In practice, agents exhibit a failure mode specific to this setting: their reasoning diverges from environmental observations, and once diverged, the error compounds across subsequent steps. As illustrated in Figure 1(a), when tasked with `put a cool mug in coffeemachine`, an agent at step 7 issues `move mug 1 to fridge 1` *without first walking to the fridge*, an illegal action that the environment rejects with "`Nothing happens.`" At step 8, instead of recognising the rejection, the agent reasons that *"mug 1 is in the fridge"*—a state inconsistent with the observation it just received—and issues an action that returns the mug to the coffee machine. The same belief is then repeated at every subsequent step: the agent never walks to the fridge, never queries its inventory, and accumulates 38 more illegal actions before the episode terminates in failure. In our evaluation of outcome-based RL on ALFWorld[Shridhar et al., 2020], this pattern is systematic: failed trajectories accumulate on average **4$\times$ more** reasoning–observation inconsistencies than successful ones (12.2 vs. 3.1 per episode; Figure 1(b)), and once an inconsistency appears, **60% of the subsequent 15 turns remain inconsistent in failed episodes, compared to 30% in successful ones** (Figure 1(c)), indicating that once the agent's reasoning diverges from observations, it tends to stay diverged for the rest of the failed episode. GRPO does not resolve this because it computes a single sequence-level advantage and applies it uniformly to every token in the sequence: the step where reasoning first diverged receives the same advantage as any correct step, leaving the model with no training signal to penalise reasoning–observation divergence where it occurs.

A natural remedy is to introduce a process reward at each turn[Zheng et al., 2025b]. Process supervision has proven effective in settings where intermediate steps are directly verifiable: mathematical reasoning steps can be checked for correctness[Lightman et al., 2024], search queries and retrieved documents can be scored for relevance[Xiong et al., 2025], and generated code can be executed against test cases[Li et al., 2025]. In these settings, the process reward targets something concrete, namely an action that either passes a check or does not. Reasoning inconsistency in agent tasks presents a different challenge. The divergence occurs inside the thinking process, not in the action

itself, and free-form reasoning provides no well-defined target for a judge to evaluate. An agent can produce reasoning tokens that are broadly compatible with prior observations, avoiding explicit contradictions, without ever committing to a specific claim about what has been accomplished or what outcome the next action should produce. Surface-level consistency checks can be satisfied without the agent actually tracking task state. What is missing is not denser supervision, but a structure that forces the agent to make explicit, verifiable commitments before each action: claims that can be checked against interaction history and against the subsequent observation.

We address this by requiring the agent to make two explicit commitments at each step, an approach we call **ARC**(**A**nchored **R**easoning with **C**ommitment). A global progress statement requires the agent to state what has been accomplished and what remains, grounding its current understanding in the actual observation history. A local outcome prediction requires the agent to commit to an expected environment state after the current action, before the action is executed. Together, these two commitments enforce a reasoning discipline that anchors the agent to the actual environment state across turns. With explicit structure in place, process supervision becomes tractable: the LLM judge assesses whether the global progress statement accurately reflects the observation history, and whether the local outcome prediction is borne out by the subsequent observation. To prevent sparse outcome rewards and dense process rewards from interfering with each other during training, we decouple their advantage estimates via independent group normalization before combining them.

Our main contributions are as follows.

• We identify and quantify a systematic failure mode in outcome-based RL for LLM agents: reasoning-observation inconsistency, in which agent reasoning diverges from environmental reality and compound across steps.

• We propose ARC, a structured reasoning schema with two explicit commitments per step: a global progress statement and a local outcome prediction that enforce a reasoning discipline keeping the agent anchored to the actual environment state and enable well-defined process supervision.

• Experiments on ALFWorld, WebShop, and WebArena demonstrate that ARC reduces inconsistent trajectories and improves task completion over both outcome-only RL and process-supervised baselines.

2 Related Works

2.1 Reinforcement Learning for LLM Agents

RL has been increasingly adopted to train LLM-based agents. Early efforts use RLHF[Ouyang et al., 2022] to align outputs with human preferences via learned reward models. More recently, GRPO[Shao et al., 2024] replaces an explicit critic by sampling multiple trajectories per query and using their mean reward as a baseline, significantly enhancing reasoning capabilities. Several variants improve upon GRPO: DAPO[Yu et al., 2025] introduces clip-higher and dynamic sampling, GSPO[Zheng et al., 2025a] reformulates clipping at the sequence level, ARPO[Dong et al., 2025] promotes exploration via high-entropy tokens, and GiGPO[Feng et al., 2025] groups state–action pairs sharing the same state for finer advantage estimation. These methods have demonstrated effectiveness across diverse agentic settings including web search, code generation, and GUI interaction[Wu et al., 2025, Qin et al., 2025, Cai et al., 2025].

2.2 Process Supervision for LLM Agents

Process reward models (PRMs) provide step-level reward rather than a single terminal signal[Zheng et al., 2025b], first validated in math reasoning[Lightman et al., 2024, Wang et al., 2024]. Recent work extends PRMs to agentic settings along two lines. The first infers step-level reward from trajectory outcomes: AgentPRM[Xi et al., 2026] trains step-level value functions via MC rollouts, iStar[Liu et al., 2025] derives implicit step rewards from trajectory-level DPO, and ProgRM[Zhang et al., 2025a] assigns progress reward by LCS alignment against reference trajectories. The second evaluates each action against independent step-level criteria: Web-Shepherd[Chae et al., 2025b] checks actions against subgoal checklists, STEVE[Lu et al., 2025] verifies GUI actions from pre/post screenshots, RAG-Gym[Xiong et al., 2025] trains a step-level critic for retrieval, and TARL[Tan et al., 2025] uses an LLM judge to score dialogue turns. Both lines provide denser supervision than

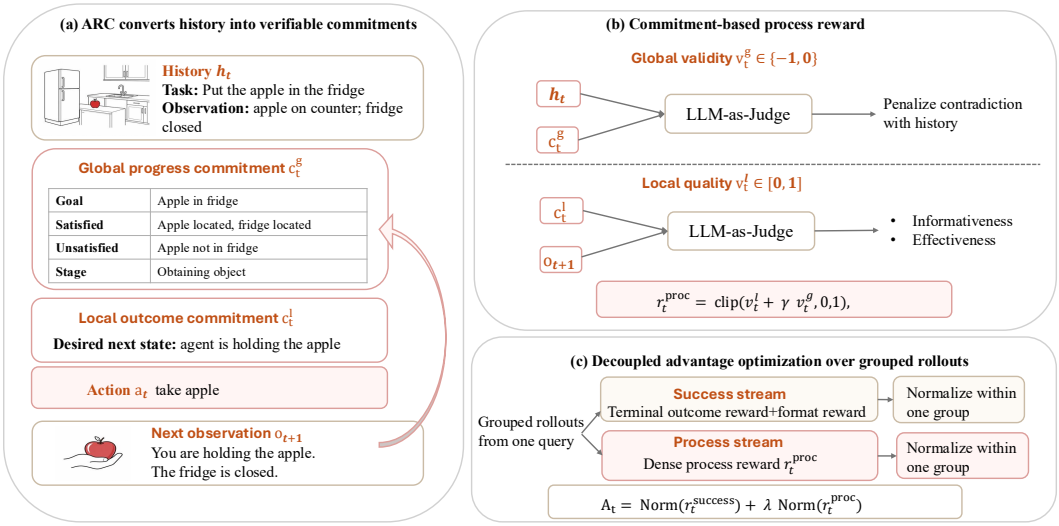


Figure 2: **Overview of ARC.** (a) The policy produces a global progress commitment c_t^g (task-level milestones) and a local outcome commitment c_t^l (desired next environment state, DNES), then generates action a_t conditioned on both. (b) A frozen LLM judge scores each turn: global validity $v_t^g \in \{-1, 0\}$ penalizes claims contradicting history; local quality $v_t^l \in [0, 1]$ evaluates DNES against the subsequent observation o_{t+1}. (c) Outcome and process rewards are normalized independently within each rollout group and combined via weight λ.

terminal rewards, but all evaluate the quality of *actions* rather than the quality of *reasoning*. ARC addresses this gap by making reasoning commitments explicit and directly supervisable.

3 Method

3.1 Problem Formulation

We model the multi-turn interaction as a Partially Observable Markov Decision Process (POMDP), where the environment evolves according to a latent state s_t that is not fully observable to the agent. At each turn, the agent receives an observation o_t and outputs an action a_t. Since a single observation is generally insufficient to infer the underlying state, the agent conditions its decisions on the interaction history $h_t = (o_{\leq t}, a_{<t})$. A POMDP is specified by $(\mathcal{S}, \mathcal{A}, \mathcal{O}, T, \Omega, R)$, where T denotes the transition dynamics and Ω denotes the observation model.

In our setting, actions are natural-language instructions executed by the environment. The environment provides sparse feedback, issuing only a terminal success signal $R_T \in \{0, 1\}$ at the end of each episode. The learning objective is to maximize the expected terminal success.

To support long-horizon decision making under partial observability, the agent produces an explicit commitment pair $c_t = (c_t^g, c_t^l)$ before generating the executable action. The model outputs <think> <global>$\cdots$</global><local>$\cdots$</local></think><action>$\cdots$</action>, where only the <action> segment is executed by the environment to produce the next observation. This generation order induces a joint policy over commitments and actions:

$$\pi_\theta(c_t, a_t \mid h_t) = \pi_\theta(c_t \mid h_t)\, \pi_\theta(a_t \mid h_t, c_t).$$

3.2 Anchored Reasoning with Commitment

We introduce **ARC** (**A**nchored **R**easoning with **C**ommitment), a structured reasoning schema that requires the agent to make two explicit, verifiable claims before each action: one about what has been established in the task so far, and one about what the environment should look like after the next successful action. These commitments serve as anchors that keep the agent's reasoning grounded in the actual environment state across turns and provide well-defined targets for process supervision.

Global progress commitment c_t^g. c_t^g captures slowly-varying task progress and follows a fixed four-field schema: **(i) Goal**, a one-sentence restatement of the objective and constraints from the task prompt; **(ii) Progress (satisfied)**, a list of milestones or conditions that have been confirmed true based on interaction history; **(iii) Progress (unsatisfied)**, a list of milestones or conditions that remain unconfirmed or are known to be false; and **(iv) Stage**, a one-phrase characterization of the current task stage (e.g., *locating target*, *verifying constraints*, *completing purchase*). This global commitment evolves slowly across turns and must remain consistent with the accumulated interaction history. It is directly verifiable: a judge can check whether the satisfied/unsatisfied milestones accurately reflect what has been observed and what remains unknown.

Local outcome commitment c_t^l. c_t^l is a single-sentence specification of the **Desired Next Environment State (DNES)**: the observable environment change that should hold in o_{t+1} if the next action succeeds (e.g., *the cart page now shows item X*, *the desklamp is illuminating pillow 1*). Crucially, the DNES specifies an *outcome predicate* rather than an action: it does not prescribe how to achieve the change (e.g., "click X"), but what environmental condition should be satisfied after a correct interaction. For a DNES to be useful, it must be specific enough to discriminate between candidate actions; a vacuous predicate (e.g., "something will change") provides no constraint on action selection and is treated as uninformative. This design yields two properties that are important for learning: (1) a specific DNES constrains action generation toward task-consistent environment changes, and (2) the DNES is directly verifiable against the subsequent observation o_{t+1}, providing a low-variance, step-level learning signal.

3.3 Process Reward Design

With explicit commitments in place, we assign dense, turn-level process rewards by evaluating commitment quality at each step using a frozen LLM-as-Judge.

(i) Global commitment validity $v_t^g \in \{-1, 0\}$. The judge checks whether the global commitment accurately reflects the accumulated interaction history. A score of -1 is assigned if the commitment contains a claim that contradicts prior observations (e.g., marking a milestone as satisfied when the interaction history shows it has not been achieved); otherwise the score is 0. This directly penalizes the reasoning–observation inconsistency identified in the introduction.

(ii) Local commitment quality $v_t^l \in [0, 1]$. The judge assesses two aspects of the DNES: *informativeness* (how specifically the DNES constrains candidate actions) and *effectiveness* (whether the predicted environment state is borne out by the subsequent observation o_{t+1}). Effectiveness is gated by informativeness: a vacuous prediction that imposes no constraint on action selection receives no credit regardless of o_{t+1}, preventing the agent from gaming the reward with trivially satisfied predicates. The implemented local quality uses the weighted normalized informativeness and effectiveness scores, as specified in Eq. 3.

Composite process reward. The composite process reward at turn t is

$$r_t^{\text{proc}} = v_t^l + \gamma\, v_t^g, \tag{1}$$

where $v_t^l \in [0, 1]$ rewards local commitment quality, $v_t^g \in \{-1, 0\}$ penalizes global inconsistency, and γ is a scalar weight. We clip r_t^{proc} to $[0, 1]$. In the implementation, $v_t^l = 0.7(L_t - 1)/4 + 0.3E_t$ with $L_t \in \{1, \ldots, 5\}$, $E_t \in \{0, 1\}$ set to zero when $L_t \leq 2$, and $\gamma = 0.5$. The global validity term is $v_t^g \in \{-1, 0\}$. This matches the expanded scoring rule in Eq. 3 and the frozen judge rubric in Appendix D.1.

3.4 Decoupled Advantage Optimization

The agent is trained with two reward signals: a sparse outcome reward $r_t^{\text{succ}} \in \{0, 1\}$ broadcast from the terminal success signal to all turns, and a dense process reward r_t^{proc} defined above. We additionally apply a deterministic format reward $r_t^{\text{fmt}} \in \{0, 1\}$ that checks for well-formed commitment tags, incorporated as $r_t^{\text{succ}} \mathrel{+}= \beta_f\, r_t^{\text{fmt}}$.

A standard approach to combining multiple reward signals is to sum them before computing advantages. This is problematic when the constituent signals differ in scale and density: when summed before group normalization, the group statistics are dominated by the high-variance dense signal, which compresses the effective contribution of the outcome reward to the advantage estimate.

Table 1: Main results on ALFWorld and WebShop. ALFWorld reports task success rates averaged over Pick, Clean, Heat, Cool, Look, and Pick2 categories, with Overall denoting aggregate performance. WebShop reports both Score and Success. ARC is compatible with multiple base RL optimizers. Best results are in bold and second best are underlined.

Method	ALFWorld							WebShop	
	Pick	Clean	Heat	Cool	Look	Pick2	Overall	Score	Success
Zero-shot	33.4	21.6	19.3	6.9	2.8	3.2	14.8	26.4	7.8
ReAct	48.5	35.4	34.3	13.2	18.2	17.6	31.2	46.2	19.5
Reflexion	62.0	41.6	44.9	30.9	36.3	23.8	42.7	58.1	28.8
PPO (critic)	92.3	64.0	92.5	89.5	80.3	68.8	80.4	81.4	68.7
RLOO	87.6	78.2	87.3	81.3	71.9	48.9	75.5	80.3	65.7
GRPO	90.8	66.1	89.3	74.7	72.5	64.7	77.6	79.3	66.1
GiGPO	<u>97.7</u>	82.7	**98.8**	83.7	89.3	79.2	<u>90.8</u>	84.4	72.8
VAGEN (GRPO)	97.1	<u>92.6</u>	68.8	<u>92.0</u>	<u>92.3</u>	**87.5**	90.0	<u>85.3</u>	<u>78.1</u>
ARC (GRPO)	**100**	**100**	<u>93.8</u>	**100**	**100**	<u>83.3</u>	**96.4**	**88.1**	**81.6**

We therefore adopt the decoupled normalization principle of GDPO [Liu et al., 2026], normalizing each reward stream independently within the rollout group:

$$A_t = \frac{r_t^{\text{succ}} - \mu(r^{\text{succ}})}{\sigma(r^{\text{succ}}) + \epsilon} + \lambda \cdot \frac{r_t^{\text{proc}} - \mu(r^{\text{proc}})}{\sigma(r^{\text{proc}}) + \epsilon}, \tag{2}$$

where $\mu(\cdot)$ and $\sigma(\cdot)$ are computed over all turns in the group and λ is a scalar weight. This ensures that the outcome advantage is normalized against the average task success within the group, and the process advantage against the average reasoning quality, so that neither signal dominates the other by virtue of scale alone.

4 Experiments

4.1 Experimental setting

Benchmarks. ALFWorld [Shridhar et al., 2020] is a text-based embodied environment requiring multi-step household manipulation under partial observability, comprising 3,827 task instances across six categories. We follow the standard evaluation protocol of prior work[Feng et al., 2025]. WebShop [Yao et al., 2022a] is a simulated e-commerce environment where agents navigate HTML-based pages to satisfy shopping constraints. WebArena [Zhou et al., 2023] evaluates scalability across five real-world websites. We follow the WebRL [Qi et al., 2025] protocol using WebArena-Lite (165 evaluation tasks, 647 training tasks).

Baselines. We compare against prompting-based agents (Zero-shot, ReAct [Yao et al., 2022b], Reflexion [Shinn et al., 2023]), outcome-only RL methods (PPO[Schulman et al., 2017], GRPO [Shao et al., 2024], GiGPO [Feng et al., 2025]), and VAGEN [Wang et al., 2025], which augments outcome-based RL with dense per-turn process supervision.

Models and Implementation. We use Qwen2.5-7B-Instruct [Yang et al., 2024] as the base model, trained from scratch without task-specific supervised data. Qwen3-235B-A22B-FP8 [Yang et al., 2025] serves as a frozen LLM-as-Judge providing turn-level process supervision during training only; it is not used at inference. Full hyperparameters are provided in Appendix A.

4.2 Main Results on ALFWorld and WebShop

Table 1 summarizes the main results on ALFWorld and WebShop. ARC achieves state-of-the-art performance on both benchmarks, reaching 96.4% overall success on ALFWorld and 81.6% success on WebShop.

ARC vs. outcome-only RL. Compared to GRPO, ARC improves ALFWorld overall by +18.8 points and WebShop success by +15.5 points. The gap persists against GiGPO (+5.6 on ALFWorld,

+8.8 on WebShop), which uses fine-grained step-level credit assignment on top of GRPO. Since GiGPO already addresses credit assignment across turns, this remaining gap suggests that the gains from ARC stem specifically from structured process supervision on the reasoning process, rather than from improved optimization dynamics alone.

ARC vs. VAGEN. VAGEN augments GRPO with dense per-turn supervision on environment state prediction, making it the most directly comparable baseline. ARC outperforms VAGEN by +6.4 points on ALFWorld and +3.5 points on WebShop. Both methods add process-level supervision to outcome-based RL, but VAGEN targets the accuracy of environment state transitions while ARC supervises the consistency of the agent's reasoning commitments with actual observations. This gap suggests that grounding the reasoning process is a more effective supervision target than grounding state predictions.

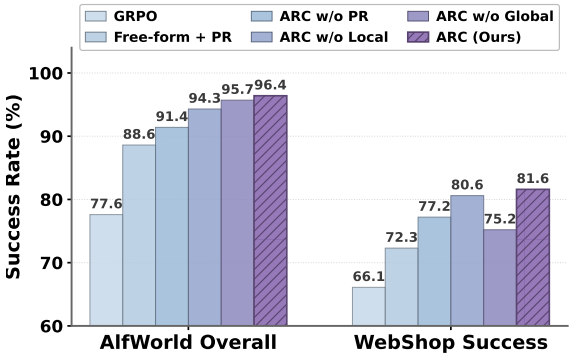


Figure 3: Ablation results on ALFWorld and WebShop. We cross two axes: reasoning format (structured schema vs. free-form) and supervision signal (outcome reward only vs. process reward), with component-level ablations removing global or local commitments.

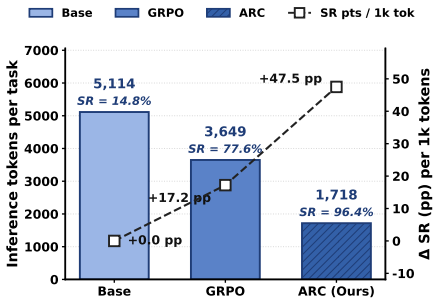


Figure 4: Inference token cost vs. success rate on ALFWorld. ARC uses 53% fewer tokens than GRPO while achieving an 18.8-point higher success rate (about 2.76× better token efficiency).

4.3 Ablation Analysis

We conduct ablation studies to isolate the contributions of (i) the structured commitment schema and (ii) the process reward. Results are shown in Figure 3.

Structure matters more than reward signal. We design a 2×2 comparison crossing reasoning format (free-form vs. structured schema) with supervision signal (outcome-only vs. process reward). As shown in Figure 3, adding structure to free-form reasoning (GRPO → ARC w/o PR) yields +13.8% on ALFWorld and +11.1% on WebShop. Adding process reward to free-form reasoning (GRPO → Free-form + PR) yields +11.0% and +6.2%. Crucially, **structured commitments without process reward (91.4%) outperform free-form reasoning with process reward (88.6%) on ALFWorld**, demonstrating that the schema itself—by forcing explicit grounding and intent tracking—provides stronger learning signal than dense reward on unstructured text. Combining both achieves the best performance (96.4%, 81.6%), confirming that structure and process supervision are complementary.

Both commitment components are necessary. Removing either commitment component degrades performance. On ALFWorld, removing the local commitment causes a larger drop (96.4 → 94.3), suggesting that per-turn grounding in observable evidence is critical for embodied tasks. On WebShop, removing the global commitment is more damaging (81.6 → 75.2), indicating that long-horizon intent tracking matters more in multi-step web navigation. Together, effective agent reasoning requires both trajectory-level anchoring and step-level grounding.

Effect of advantage coefficient λ. Table 2 reports the effect of λ in Eq. 2, the scalar weight that scales the process advantage relative to the outcome advantage. A moderate value ($\lambda = 0.1$) achieves the best performance on both ALFWorld (96.5%) and WebShop (81.3%). Larger values degrade

Table 2: Effect of λ in Eq. 2. λ=0 corresponds to ARC without process reward (commitment schema only). Results are validation success rate (%).

λ	0.0	0.1	0.2	0.3	0.5
ALFWorld	91.4	**96.5**	96.4	82.3	60.9
WebShop	77.2	**81.3**	72.3	63.3	58.7

Table 3: Effect of model size. Both methods use the same training recipe on Qwen2.5-3B-Instruct. We report ALFWorld overall success rate and WebShop success rate (%).

Method	ALFWorld	WebShop
GRPO	73.4	59.5
ARC (Ours)	**92.3**	**73.3**
Δ	+18.9 pp	+13.8 pp

performance substantially (e.g., $\lambda = 0.5$ drops to 60.9% on ALFWorld and 58.7% on WebShop), confirming that process supervision should act as a regularizer rather than the dominant training signal.

ARC generalizes to smaller model scales. Table 3 shows results with Qwen2.5-3B-Instruct under the same training recipe. ARC consistently improves over GRPO at this scale, yielding absolute gains of **+18.9 pp** on ALFWorld and **+13.8 pp** on WebShop. The results are comparable to the gains observed at 7B scale (+18.8 pp and +15.5 pp respectively). Notably, 3B+ARC (92.3%) outperforms 7B+GRPO (77.6%) on ALFWorld, suggesting that structured process supervision can compensate for reduced model capacity.

4.4 Efficiency Analysis

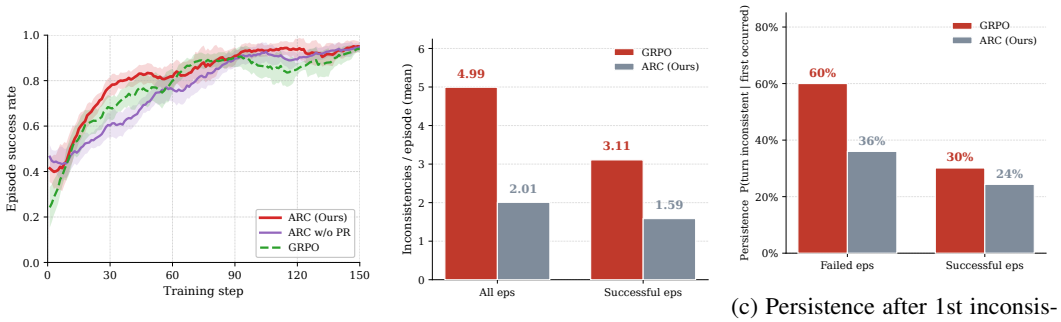
Inference efficiency. As shown in Figure 4, ARC requires only 1,718 inference tokens per task on ALFWorld, compared to 3,649 for GRPO—a 53% reduction. This is a direct consequence of the ARC schema: structured commitments reduce hallucinated or irrelevant actions, shortening episodes by 19.4% and per-turn response length by 29.5%. Despite the richer structured output, the agent reaches the goal in fewer steps. In terms of token efficiency—success-rate gain per 1k tokens relative to the base model—ARC achieves 47.5 success-rate percentage points per 1k inference tokens, compared to 17.2 for GRPO, an approximately $\mathbf{2.76}\times$ improvement.

Training overhead. ARC adds a single LLM judge call per rollout turn during training, incurring $+14\%$ wall-clock overhead over GRPO (28.8 vs. 25.3 hours on ALFWorld). The judge is not invoked at inference, so ARC's deployment cost is identical to GRPO. Together, the training and inference results show that ARC pays a modest training-time cost in exchange for substantially more efficient inference. Full efficiency statistics are in Appendix B. Additional analyses of judge-size sensitivity, prompt sensitivity, and token cost are provided in Appendix E.1.

4.5 Training Dynamics and Mechanism Analysis

Training dynamics. Figure 5a reports learning curves on ALFWorld. The schema-only variant initially trails GRPO because the agent must bootstrap meaningful commitments from sparse outcome rewards alone; once commitments become informative, conditioning actions on structured reasoning yields better decisions and the variant overtakes GRPO. Adding process reward removes this bootstrapping cost: it grounds the commitments early in training, so full ARC improves substantially faster from the first checkpoint and converges to the highest final success rate.

Does ARC reduce reasoning–observation inconsistency? Figures 5b and 5c directly verify the failure mode that motivated our method. ARC cuts the mean per-episode inconsistency count by $\mathbf{2.5}\times$ ($4.99 \to 2.01$); the reduction holds on successful episodes alone ($3.11 \to 1.59$), ruling out an explanation in which ARC merely avoids inconsistent episodes by completing tasks faster. Pooled persistence after the first inconsistency drops from 60% to 36% on failed episodes and from 30% to 24% on successful ones, confirming that even when divergences occur, ARC recovers in fewer turns. These results show that ARC's task-success gains stem from jointly anchoring reasoning to observations rather than from longer reasoning or extra compute. Controlled DNES interventions and realization-rate analyses, a commitment-content audit, and the independent inconsistency-labeling protocol are documented in Appendix E.2, E.3, and E.4.



(a) Learning curves on ALFWorld. (b) Inconsistencies per episode. (c) Persistence after 1st inconsistency.

Figure 5: **Training dynamics and mechanism analysis on ALFWorld. (a)** ARC trains faster and converges higher than GRPO and ARC w/o PR. **(b)** ARC reduces the mean per-episode inconsistency count from 4.99 to 2.01 overall, and from 3.11 to 1.59 on successful episodes alone. **(c)** Pooled persistence after the first inconsistency drops from 60% to 36% on failed episodes and from 30% to 24% on successful ones.

4.6 Generalization to WebArena

Table 4: Performance on WebArena-Lite across five domains. Our method is trained from scratch using Qwen2.5-7B and compared against prompting, external world model, internal world model, and RL baselines.

Model	Method	Reddit	GitLab	CMS	Map	Shopping	Avg SR
Qwen2.5-7B	Prompting	9.5	3.1	8.6	0.0	8.7	6.1
Qwen-235B	Prompting	14.3	9.4	**37.1**	12.9	**26.7**	**21.2**
GPT-4o-mini	Prompting	5.3	3.1	8.2	11.6	18.8	9.4
GPT-4o	Prompting	10.5	10.0	20.0	20.0	11.1	13.9
GPT-4o-turbo	Prompting	10.5	**16.7**	14.3	**36.7**	13.3	17.6
GPT-4o-mini	WMA	7.9	8.7	11.5	22.3	19.3	13.5
Qwen2.5-7B	WebSynthesis	9.5	9.4	17.1	–	19.6	14.9
Qwen2.5-7B	IWM	0.0	15.4	7.1	8.6	6.5	7.3
Qwen2.5-7B	AgentGym-RL	28.6	3.1	17.1	6.5	15.2	13.3
Qwen2.5-7B	Ours	**33.3**	6.3	22.9	6.5	19.5	16.9

We evaluate on WebArena-Lite [Zhou et al., 2023] (165 tasks, five real-world websites) following the WebRL [Qi et al., 2025] protocol. This setting is substantially more challenging than ALFWorld and WebShop, featuring multi-page navigation, diverse website structures, and high-entropy observations. As shown in Table 4, ARC achieves 16.9% average success, outperforming IWM [Zhang et al., 2025b] (+9.6), which predicts raw next observations, and AgentGym-RL (+3.6) under the same base model. Despite training a 7B model from scratch, ARC is competitive with GPT-4o-turbo prompting (17.6%) and surpasses external planning methods (WMA [Chae et al., 2025a], WebSynthesis [Gao et al., 2025]) that use stronger LLMs, confirming that structured process supervision generalizes to complex web environments.

5 Conclusion

We identify reasoning-observation inconsistency as a key failure mode in outcome-based RL for LLM agents: without structural constraints, free-form reasoning drifts from environmental reality across turns, producing actions that are locally plausible but globally misaligned. We propose ARC (Anchored Reasoning with Commitment), which addresses this by requiring the agent to produce two explicit commitments at each turn: a global commitment tracking task progress and a local commitment predicting the next observable outcome. These commitments enforce a reasoning discipline that keeps the agent grounded, and provide well-defined anchors for an LLM judge to deliver dense, turn-level process supervision via decoupled advantage optimization. Experiments on

ALFWorld, WebShop, and WebArena demonstrate consistent improvements over prompting-based agents, outcome-only RL methods, and process-supervised baselines. Ablation studies confirm that both the commitment schema and process reward contribute independently, and that ARC's gains transfer across model scales.

Limitations. Our approach has several limitations. First, ARC relies on a frozen LLM judge (Qwen3-235B) during training to provide process supervision; the quality of commitments learned is bounded by the judge's evaluation capability, and the training cost scales with judge inference. Second, we evaluate exclusively on text-based environments; extending to multimodal observations (e.g., GUI screenshots) may require adapting the commitment schema to handle visual grounding. Finally, the structured schema adds a fixed formatting overhead to the policy output; while this is offset by shorter episodes in practice, it may be suboptimal for very short-horizon tasks where free-form reasoning suffices. Furthermore, the experiments primarily cover predictable text-based transitions. In a stochastic environment a reasonable action may not realize its intended next state, so scoring DNES realization against a single observation can confound action quality with transition randomness. Open-ended user responses pose a similar issue; uncertainty-aware commitments and rewards remain unexplored (Appendix E.4).

Acknowledgments and Disclosure of Funding

The SJTU team is supported by National Natural Science Foundation of China (62322603) and Shanghai Municipal Science and Technology Major Project (2025SHZDZX025D08).

References

Shaofei Cai, Yulei Qin, Haojia Lin, Zihan Xu, Gang Li, Yuchen Shi, Zongyi Li, Yong Mao, Siqi Cai, Xiaoyu Tan, et al. Smartsnap: Proactive evidence seeking for self-verifying agents. *arXiv preprint arXiv:2512.22322*, 2025.

Hyungjoo Chae, Namyoung Kim, Kai Tzu iunn Ong, Minju Gwak, Gwanwoo Song, Jihoon Kim, Sunghwan Kim, Dongha Lee, and Jinyoung Yeo. Web agents with world models: Learning and leveraging environment dynamics in web navigation, 2025a. URL https://arxiv.org/abs/2410.13232.

Hyungjoo Chae, Sunghwan Kim, Junhee Cho, Seungone Kim, Seungjun Moon, Gyeom Hwangbo, Dongha Lim, Minjin Kim, Yeonjun Hwang, Minju Gwak, et al. Web-shepherd: Advancing prms for reinforcing web agents. *arXiv preprint arXiv:2505.15277*, 2025b.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. Agentic reinforced policy optimization. *ArXiv*, abs/2507.19849, 2025. URL https://api.semanticscholar.org/CorpusID:280323841.

Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. *ArXiv*, abs/2505.10978, 2025. URL https://api.semanticscholar.org/CorpusID:278715074.

Yifei Gao, Junhong Ye, Jiaqi Wang, and Jitao Sang. Websynthesis: World-model-guided mcts for efficient webui-trajectory synthesis, 2025. URL https://arxiv.org/abs/2507.04370.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? *ArXiv*, abs/2310.06770, 2023. URL https://api.semanticscholar.org/CorpusID:263829697.

Qingyao Li, Xinyi Dai, Xiangyang Li, Weinan Zhang, Yasheng Wang, Ruiming Tang, and Yong Yu. CodePRM: Execution feedback-enhanced process reward model for code generation. In Wanxiang

Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar, editors, *Findings of the Association for Computational Linguistics: ACL 2025*, pages 8169–8182, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.428. URL https://aclanthology.org/2025.findings-acl.428/.

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

Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, et al. Gdpo: Group reward-decoupled normalization policy optimization for multi-reward rl optimization. *arXiv preprint arXiv:2601.05242*, 2026.

Xiaoqian Liu, Ke Wang, Yuchuan Wu, Fei Huang, Yongbin Li, Junge Zhang, and Jianbin Jiao. Agentic reinforcement learning with implicit step rewards. *arXiv preprint arXiv:2509.19199*, 2025.

Fanbin Lu, Zhisheng Zhong, Ziqin Wei, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Steve: A step verification pipeline for computer-use agent training. *arXiv preprint arXiv:2503.12532*, 2025.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke E. Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Francis Christiano, Jan Leike, and Ryan J. Lowe. Training language models to follow instructions with human feedback. *ArXiv*, abs/2203.02155, 2022. URL https://api.semanticscholar.org/CorpusID:246426909.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Wenyi Zhao, Yu Yang, Xinyue Yang, Jiadai Sun, Shuntian Yao, Tianjie Zhang, Wei Xu, Jie Tang, and Yuxiao Dong. Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning, 2025. URL https://arxiv.org/abs/2411.02337.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, Wanjun Zhong, Kuanye Li, Jiale Yang, Yu Miao, Woyu Lin, Longxiang Liu, Xu Jiang, Qianli Ma, Jingyu Li, Xiaojun Xiao, Kai Cai, Chuang Li, Yaowei Zheng, Chaolin Jin, Chen Li, Xiao Zhou, Minchao Wang, Haolin Chen, Zhaojian Li, Haihua Yang, Hai-Yi Liu, Feng Lin, Tao Peng, Xin Liu, and Guang Shi. Ui-tars: Pioneering automated gui interaction with native agents. *ArXiv*, abs/2501.12326, 2025. URL https://api.semanticscholar.org/CorpusID:275788034.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017. URL https://api.semanticscholar.org/CorpusID:28695052.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Jun-Mei Song, Mingchuan Zhang, Y. K. Li, Yu Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *ArXiv*, abs/2402.03300, 2024. URL https://api.semanticscholar.org/CorpusID:267412607.

Noah Shinn, Federico Cassano, Beck Labash, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In *Neural Information Processing Systems*, 2023. URL https://api.semanticscholar.org/CorpusID:258833055.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew J. Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *ArXiv*, abs/2010.03768, 2020. URL https://api.semanticscholar.org/CorpusID:222208810.

Theodore Sumers, Shunyu Yao, Karthik R Narasimhan, and Thomas L Griffiths. Cognitive architectures for language agents. *Transactions on Machine Learning Research*, 2023.

Weiting Tan, Xinghua Qu, Ming Tu, Meng Ge, Andy T Liu, Philipp Koehn, and Lu Lu. Process-supervised reinforcement learning for interactive multimodal tool-use agents. *arXiv preprint arXiv:2509.14480*, 2025.

Kangrui Wang, Pingyue Zhang, Zihan Wang, Yaning Gao, Linjie Li, Qineng Wang, Hanyang Chen, Chi Wan, Yiping Lu, Zhengyuan Yang, Lijuan Wang, Ranjay Krishna, Jiajun Wu, Fei-Fei Li, Yejin Choi, and Manling Li. Vagen: Reinforcing world model reasoning for multi-turn vlm agents. *ArXiv*, abs/2510.16907, 2025. URL https://api.semanticscholar.org/CorpusID:282210682.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439, 2024.

Tianxin Wei, Ting-Wei Li, Zhining Liu, Xuying Ning, Ze Yang, Jiaru Zou, Zhichen Zeng, Ruizhong Qiu, Xiao Lin, Dongqi Fu, et al. Agentic reasoning for large language models. *arXiv preprint arXiv:2601.12538*, 2026.

Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Deyu Zhou, Pengjun Xie, and Fei Huang. Webwalker: Benchmarking llms in web traversal. *ArXiv*, abs/2501.07572, 2025. URL https://api.semanticscholar.org/CorpusID:275471576.

Zhiheng Xi, Chenyang Liao, Guanyu Li, Zhihao Zhang, Wenxiang Chen, Binghai Wang, Senjie Jin, Yuhao Zhou, Jian Guan, Wei Wu, et al. Agentprm: Process reward models for llm agents via step-wise promise and progress. In *Proceedings of the ACM Web Conference 2026*, pages 4184–4195, 2026.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh J Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Advances in Neural Information Processing Systems*, 37:52040–52094, 2024.

Guangzhi Xiong, Qiao Jin, Xiao Wang, Yin Fang, Haolin Liu, Yifan Yang, Fangyuan Chen, Zhixing Song, Dengyu Wang, Minjia Zhang, et al. Rag-gym: Systematic optimization of language agents for retrieval-augmented generation. *arXiv preprint arXiv:2502.13957*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yi-Chao Zhang, Yunyang Wan, Yuqi Liu, Zeyu Cui, Zhenru Zhang, Zihan Qiu, Shanghaoran Quan, and Zekun Wang. Qwen2.5 technical report. *ArXiv*, abs/2412.15115, 2024. URL https://api.semanticscholar.org/CorpusID:274859421.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *ArXiv*, abs/2207.01206, 2022a. URL https://api.semanticscholar.org/CorpusID:250264533.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *ArXiv*, abs/2210.03629, 2022b. URL https://api.semanticscholar.org/CorpusID:252762395.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen,

Chengyi Wang, Honglin Yu, Weinan Dai, Yuxuan Song, Xiang Wei, Haodong Zhou, Jingjing Liu, Wei Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yong-Xu Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale. *ArXiv*, abs/2503.14476, 2025. URL https://api.semanticscholar.org/CorpusID:277104124.

Danyang Zhang, Situo Zhang, Ziyue Yang, Zichen Zhu, Zihan Zhao, Ruisheng Cao, Lu Chen, and Kai Yu. Progrm: Build better gui agents with progress rewards. *arXiv preprint arXiv:2505.18121*, 2025a.

Kai Zhang, Xiangchao Chen, Bo Liu, Tianci Xue, Zeyi Liao, Zhihan Liu, Xiyao Wang, Yuting Ning, Zhaorun Chen, Xiaohan Fu, Jian Xie, Yuxuan Sun, Boyu Gou, Qi Qi, Zihang Meng, Jianwei Yang, Ning Zhang, Xian Li, Ashish Shah, Dat Huynh, Hengduo Li, Zi Xian Yang, Sara Cao, Lawrence Jang, Shuyan Zhou, Jiacheng Zhu, Huan Sun, Jason Weston, Yu Su, and Yifan Wu. Agent learning via early experience. *ArXiv*, abs/2510.08558, 2025b. URL https://api.semanticscholar.org/CorpusID:281951121.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization. *ArXiv*, abs/2507.18071, 2025a. URL https://api.semanticscholar.org/CorpusID:280017753.

Congming Zheng, Jiachen Zhu, Zhuoying Ou, Yuxiang Chen, Kangning Zhang, Rong Shan, Zeyu Zheng, Mengyue Yang, Jianghao Lin, Yong Yu, et al. A survey of process reward models: From outcome signals to process supervisions for large language models. *arXiv preprint arXiv:2510.08049*, 2025b.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. *ArXiv*, abs/2307.13854, 2023. URL https://api.semanticscholar.org/CorpusID:260164780.

A Implementation Details

All agents are initialized from **Qwen2.5-7B-Instruct** and trained with a PPO-style clipped surrogate objective. At each environment step the agent produces output following the ARC schema: `<think><global>...<local>...</think><action>...</action>`. A binary format reward η_{fmt} is granted when the output conforms to this schema with a parseable `<action>` block; an invalid-action penalty η_{inv} is deducted when the chosen action lies outside the admissible set (WebShop only). GDPO decouples the environment-reward and process-reward advantage streams by normalizing each independently within its group before combining them with weight w_{plan}, preventing the two reward channels from interfering with each other's gradient scale. No KL penalty to the reference policy is used across all environments.

For WebArena we follow the WebRL [Qi et al., 2025] protocol, training on the 647 non-evaluation tasks of WebArena-Lite and evaluating on the held-out 165 tasks. The agent operates on the raw HTML observation with a short prompt window (2,048 tokens) and a long response budget (16,000 tokens) to accommodate chain-of-thought plus structured action output. Compared to ALFWorld and WebShop, WebArena uses a smaller group size (G=4) and mini-batch size (64) due to the substantially longer per-step sequences. Detailed hyper-parameters are listed in Table 5.

Table 5: Hyper-parameters for ALFWorld, WebShop, and WebArena used in our experiments.

Config	**ALFWorld**	**WebShop**	**WebArena**
Data & sequence			
train_batch_size	16	64	128
val_data_size	144	128	165
max_prompt_length	16000	4096	2048
max_response_length	1024	1024	16000
Rollout & environment			
G (samples per prompt)	8	8	4
$T_{\max}$ (max env steps)	50	15	16
H (history length)	50	5	–
rollout / val temperature	1.0 / 0.4	1.0 / 0.4	1.0 / 0.4
tensor_model_parallel_size	1	2	2
Optimizer			
actor learning rate		1×10^{-6}	
ppo_mini_batch_size	1024	1024	64
total_epochs	150	350	10
γ (discount)	0.95	1.0	1.0
GDPO advantage			
plan_fuse_mode		`adv` (decoupled)	
w_{plan}	0.10	0.10	0.10
Reward shaping			
η_{fmt} (format reward)	1.0	1.0	1.0
use_invalid_action_penalty	`False`	`True`	`False`
Cluster			
nnodes $\times$ gpus_per_node	2×8	4×8	4×8

B Efficiency Analysis

Table 6 reports a detailed efficiency comparison between ARC and GRPO under identical training configurations on ALFWorld.

Table 6: Efficiency analysis of ARC vs. GRPO on ALFWorld (Qwen2.5-7B, 16×H20, 150 steps, identical hyper-parameters). The `<global>+<local>` schema reduces both per-turn token count and episode length, more than compensating for the $+14\%$ wall-clock overhead introduced by the judge.

	ARC	GRPO	Δ
Training time			
Total runtime (h)	28.80	25.25	+3.55 (+14.0%)
Wall-clock judge time (h)	3.35	—	94% of overhead
Per-turn response tokens			
Mean	98.2	139.4	−41.2 (−29.5%)
Max	141	612	−471
Episode efficiency			
Episode length (turns)	21.4	26.5	−5.1 (−19.4%)
Actor tokens / episode	2,101	3,697	−1,596 (−43.2%)
Judge overhead (training only)			
Judge calls / episode	21.4	—	one call per turn
Total judge calls	410.7K	—	across full training
Inference			
Judge calls at deploy time	**0**	0	—
Per-token inference cost	Identical (same Qwen2.5-7B, no judge invocation)		

Training overhead. ARC adds $+3.55$ hours of wall-clock training time over GRPO ($+14\%$), of which 94% is attributable to judge API calls during rollout. This is the only additional cost: ARC requires no extra GPU memory, no additional model parameters, and no architectural changes to the actor.

Schema as a length regularizer. Counterintuitively, ARC generates *fewer* tokens per turn than GRPO despite producing structured `<global>+<local>` abstractions (98.2 vs. 139.4 tokens/turn, -29.5%). The schema imposes a strict structural constraint that suppresses the verbose free-form reasoning GRPO exhibits under uncertainty—GRPO's maximum response length reaches 612 tokens versus ARC's cap of 141. Combined with shorter episodes (21.4 vs. 26.5 turns/episode, -19.4%), ARC consumes 43% fewer actor tokens per episode overall, more than compensating for the judge overhead in terms of generation cost.

Zero inference overhead. The judge is invoked exclusively during training to compute turn-level advantage estimates; it is not part of the deployed policy. At inference, ARC produces outputs using the same Qwen2.5-7B backbone as GRPO, with no additional model calls, no additional latency, and no additional hardware dependencies.

C Case Study: A Belief Error That Compounds Across 40 Turns

We provide a side-by-side trace of the baseline (GRPO with outcome-only reward) and ARC on the *same* ALFWorld task, `put a cool mug in coffeemachine`, drawn from the evaluation set. The baseline fails after 50 turns with 42 reasoning–observation inconsistencies; ARC succeeds in 16 turns with zero inconsistencies. The contrast illustrates the failure mode that motivates the paper.

ALFWorld preliminaries. ALFWorld is a partially observable, text-based household environment. The agent maintains an *inventory*—the set of objects currently held—and can interact only with objects co-located with it. A `move X to Y` action requires the agent to be standing next to `Y` *and* to currently hold `X`; otherwise the environment returns the null string "`Nothing happens.`" with no state change. The canonical action for cooling an object is `cool X with fridge 1`, which requires the agent to be at the fridge with `X` in its inventory.

C.1 Baseline (GRPO): a False Belief Locks the Agent for 38 Turns

Up to turn 6 the baseline executes correctly: it locates `mug 1` on the coffee machine and successfully takes it. Turn 7 contains the seed of the failure, which crystallises into a false belief at turn 8 and persists for the remaining 41 turns of the episode.

Turn 7 — an illegal action seeds the trap

OBS: `You pick up the mug 1 from the coffeemachine 1.`
THINK: *"Next step is to cool the mug; I will move it to the fridge."*
ACTION: `move mug 1 to fridge 1` *(illegal: agent is not at the fridge)*

Turn 8 — first inconsistency △

OBS: `Nothing happens.` *(move was rejected; mug is still in inventory)*
THINK: *"**Mug 1 is in the fridge.** The action `move mug 1 to coffeemachine 1` is not applicable because mug 1 is already in the fridge."*
⇐ *contradicts observation: nothing was moved*
ACTION: `move mug 1 to coffeemachine 1` *(ironically valid; this UNDOES the take)*

Turn 9 — the error compounds △

OBS: `You move the mug 1 to the coffeemachine 1.` *(mug back on coffee machine; inventory empty)*
THINK: *"I have already taken mug 1 from the coffeemachine. I will move mug 1 to the fridge."*
⇐ *still believes the mug is in inventory; doubles down*
ACTION: `move mug 1 to fridge 1` *(illegal: inventory empty)*

Turns 11–49 — the inconsistency persists for 38 more turns △

The agent's state has not changed since turn 9 (mug is on the coffee machine, inventory is empty, agent is still next to the coffee machine), but every subsequent reasoning step assumes the mug is in the fridge. Every action is consequently illegal and every observation is the same null string "`Nothing happens.`" Selected turns:

Turn 11	*"The mug is in the fridge."*	`open fridge 1`
Turn 18	*"Mug is in the fridge."*	`open fridge 1`
Turn 30	*"move to fridge taken multiple times."*	`close fridge 1`
Turn 45	*"moved to fridge but no cooling."*	`wait for a few minutes`
Turn 49	(same belief)	`move mug 1 to fridge 1`

⇒ *episode ends in failure.*

Mechanism. The agent's reasoning at turn 8 commits to "*mug 1 is in the fridge*" on the basis of an action (`move mug 1 to fridge 1`) that the environment explicitly rejected with "`Nothing happens.`" This single false belief drives all subsequent reasoning: turn 9 *compounds* the error (now also believing "*I have already taken mug 1*"), and turns 11–49 *persist* the belief across 38 identical observations, which provide no leverage to overturn it because the agent never walks to the fridge nor queries its own state with `inventory` or `look`.

C.2 ARC (Ours): Anchored Commitments Avoid the Trap

On the same task, ARC succeeds in 16 turns with zero flagged inconsistencies. The contrast localises to the four turns immediately after the mug is found.

Turn 12 — locate mug, commit to take

OBS: `On countertop 2, you see ...a mug 1 ....`
THINK: *Progress: located mug 1. Outcome: mug 1 will be in inventory.*
ACTION: `take mug 1 from countertop 2`

Turn 13 — commit to walk to fridge before interacting with it

OBS: `You pick up the mug 1 from the countertop 2.`
THINK: *Progress: have mug. Outcome: I will be at fridge 1.*
ACTION: `go to fridge 1` *(prerequisite: agent must be at fridge to use it)*

Turn 14 — fridge closed, open it

OBS: `You arrive at fridge 1. The fridge 1 is closed.`
THINK: *Outcome: fridge will be open.*
ACTION: `open fridge 1`

Turn 15 — use the canonical cooling action

OBS: `You open the fridge 1. The fridge 1 is open.`
THINK: *Outcome: mug 1 will be cooled.*
ACTION: `cool mug 1 with fridge 1` *(canonical AlfWorld API)*

What ARC's commitments rule out. The schema forces ARC to make two explicit commitments at every turn: a PROGRESS statement summarising what has been accomplished from the observation history, and an OUTCOME prediction stating the expected post-action state. At turn 13 the outcome prediction ("*I will be at fridge*") translates into the prerequisite action `go to fridge 1` *before* attempting to interact with the fridge. This eliminates the precondition error that triggered the baseline's failure and removes the opportunity for a false belief to form. The per-turn process reward further discourages reasoning that is inconsistent with the immediately preceding observation, so even occasional precondition slips do not propagate.

D ARC Prompt Template

The Anchored Reasoning Commitment (ARC) schema is injected into the system prompt for all three environments. The agent is required to produce `<think>...</think>` before every `<action>...</action>`. Inside `<think>`, exactly two sub-blocks must appear: a `<global>` block tracking task-level belief state, and a `<local>` block committing to the Desired Next Environment State (DNES). Below is the ALFWorld version of the ARC prompt template; WebShop and WebArena follow the same schema with environment-specific task descriptions.

ARC Prompt Template — ALFWorld (`ALFWORLD_TEMPLATE_PAPER`)

You are an expert autonomous agent operating in the AlfWorld household environment.
Your task is to: {*task_description*}.

[...history of recent observations and actions ...]

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

Your admissible actions for the current situation are:
[{admissible_actions}]

Now it's your turn to take **exactly one action** for the current step.

You MUST first produce a concise abstraction enclosed within `<think>...</think>`.
Inside `<think>`, you MUST use **ONLY** the following schema:

```
<think>
  <global>
```

```
    Goal: (1 short sentence restating the household task)
    Progress:
      - satisfied: (milestones already satisfied, or "none")
      - unsatisfied: (milestones not yet satisfied, or "none")
      - stage: (short phrase for the current task phase)
  </global>

  <local>
    Desired next environment state:
    (1 sentence: the observable condition that SHOULD hold
     in the NEXT observation. Describe WHAT should change,
     not HOW to act.)
  </local>
</think>

<action>ONE admissible action EXACTLY copied from the list</action>
```

D.1 Process Reward Judge Prompt

The process reward modelis implemented as a frozen **Qwen3-235B-A22B-FP8** called once per agent step. The judge receives the current task, interaction history, current observation (with admissible actions), the full agent output, the taken action, and the next observation, and returns three structured scores in XML.

Judge Prompt (ALFWorld)

```
You are a strict judge scoring whether an agent's abstraction meaningfully helps
action selection in ALFWorld.  Judge ONLY from the provided inputs.  No outside
knowledge.  Output ONLY the required XML.

SCORE A) GLOBAL_VALIDITY (0 or -1)
Global is a task-level belief state.  It is a CONSTRAINT, not a shaping signal.
Only penalize clear belief errors.  Do NOT reward good writing.
GLOBAL_VALIDITY = -1 if ANY of the following holds:
- ⟨global⟩ marks a constraint as satisfied without evidence.
- ⟨global⟩ asserts possession/location/state facts that clearly contradict
CURRENT_OBSERVATION.
GLOBAL_VALIDITY = 0 otherwise.

SCORE B) LOCAL_ACTION_INFORMATIVENESS (1--5)
Local describes the Desired Next Environment State (DNES). It must HELP SELECT
ACTIONS, not merely describe a predictable outcome.
Step B1 (most important):  would multiple materially different actions plausibly
lead to the same DNES?
If YES (generic DNES) ⇒ score ≤ 2.
If NO (DNES narrows action choice) ⇒ score ≥ 4.
LOW-INFORMATION examples (cap at 2):
''The room description updates'' / ''More information is shown''
HIGH-INFORMATION examples:
''The fridge is open and its contents are visible''
''The agent is holding the apple''
Step B2:  DNES must be plausible given CURRENT_OBSERVATION.
Step B3 (secondary):  DNES should enable checking an unsatisfied constraint or
necessary recovery.
Likert mapping:  1 = generic & action-invariant; 3 = mildly constraining; 5 =
sharply constrains action choice and is optimally chosen.

SCORE C) EFFECTIVENESS (0 or 1)
Checks ONLY whether DNES actually occurred in NEXT_OBSERVATION.
RULE: if LOCAL_ACTION_INFORMATIVENESS ≤ 2, EFFECTIVENESS must be 0.

OUTPUT XML ONLY:

<scores>
```

```
  <global_validity>0 or -1</global_validity>
  <local_action_informativeness>INTEGER 1-5</local_action_informativeness>
  <effectiveness>0 or 1</effectiveness>
</scores>
```

The final per-step process reward is computed as:

$$r_{\text{plan}} = \underbrace{\frac{w_l \cdot (L-1)}{4}}_{\text{local quality}} + \underbrace{w_e \cdot E}_{\text{effectiveness}} \underbrace{- w_p \cdot \mathbf{1}[G{=}{-1}]}_{\text{global penalty}}, \quad \text{clipped to } [0,1] \tag{3}$$

where $L \in \{1, \ldots, 5\}$ is the local informativeness score, $E \in \{0, 1\}$ is effectiveness, $G \in \{-1, 0\}$ is global validity, and $(w_l, w_e, w_p) = (0.7, 0.3, 0.5)$.

E Additional Analyses

This appendix provides further analyses of judge sensitivity, commitment behavior, and the diagnostic protocol, supplementing the main benchmark comparisons.

E.1 Sensitivity to the Process Judge

Judge capacity. We held the WebShop ARC training and evaluation configuration fixed and changed only the frozen training-time judge. A Qwen2.5-7B-Instruct judge, matching the policy scale, achieved 79.4% success, compared with 81.6% for Qwen3-235B-A22B-FP8 and 77.2% without process reward. Thus, the 7B judge retains 2.2 of the 4.4 percentage-point gain from process supervision obtained with the stronger judge. This sensitivity test covers WebShop only.

Method	Training-time judge	WebShop success (%)
GRPO	None	66.1
ARC without process reward	None	77.2
ARC	Qwen2.5-7B-Instruct	79.4
ARC	Qwen3-235B-A22B-FP8	**81.6**

Judge-prompt sensitivity. We rescored 50 ALFWorld turns at temperature 0 using the original Qwen3-235B-A22B-FP8 training prompt and three rubric-preserving rewrites. Evaluator inputs, scoring criteria, examples, output schema, informativeness gate, and reward formula were kept fixed. Disagreement denotes a changed final scalar process reward relative to the original prompt; the absolute difference is averaged over all 50 turns.

Rubric-equivalent rewrite	Disagreement (%)	Mean absolute difference
Formal paraphrase	12.0 (6/50)	0.0245
Role-structured	10.0 (5/50)	0.0315
Checklist-style	16.0 (8/50)	0.0410

Between 84% and 90% of turns receive exactly the same process reward under these rewrites, although the nonzero differences show sensitivity is not eliminated.

Judge token usage. Measured across 50 training-judge calls per benchmark using the original prompt, mean input/output/total tokens per call were 2,103/45/2,148 for WebShop and 2,258/45/2,303 for ALFWorld. One judge call is made at each rollout turn during training; there are no judge calls at inference time.

E.2 Commitment Interventions and Realization

Does the DNES affect the generated action? We evaluated 200 turns sampled from held-out WebShop validation trajectories. For each turn, we fixed the task, history, observation, available actions, global commitment, policy checkpoint, and deterministic decoding. After replaying the

original DNES, we regenerated the action with only the DNES changed. Meaning-preserving rewrites changed the action in 2.0% of turns; substituting a DNES sampled from another trajectory changed it in 42.0%; and masking it with an uninformative placeholder changed it in 20.5%. This controlled comparison supports a behavioral role for the pre-action DNES, but it does not constitute a separately trained action-first baseline.

Realization after action execution. For final-stage WebShop rollouts we measured the rate at which the next observation satisfied an informative DNES, where the judge's local informativeness score $L_t > 2$ and effectiveness $E_t \in \{0, 1\}$. The rate is $\sum_t \mathbf{1}[L_t > 2]E_t / \sum_t \mathbf{1}[L_t > 2]$. It is 71.3% overall, 72.8% in successful trajectories, and 60.5% in failed trajectories. This measures agreement between intended and realized outcomes, not the success of every action.

E.3 Content Audit of Policy-Generated Commitments

We conducted a post-hoc content audit over all WebShop validation trajectories using a separate content-focused rubric and manually inspected low-scoring cases.

Content property	Rate (%)
Stage specificity	90.6
Local Stage–DNES–action coherence	78.4
Stage transition stability	73.3
Decision-relevant milestone adequacy	67.9

These properties measure whether Stage descriptions identify concrete task phases, whether the immediate plan is coherent, whether changes in Stage are supported by new evidence, and whether Satisfied/Unsatisfied fields capture decision-relevant conditions. The milestone assessment does not assume an exhaustive predefined decomposition. Lower-scoring cases included repeated WebShop option verification when selected options were not persistently visible in the page text, delayed progress-field updates, and Stage changes without new evidence.

E.4 History Construction and Diagnostic Metric

Policy input and initialization. ARC does not use a separate task-specific SFT stage: Qwen2.5-7B-Instruct begins RL with the structured generation prompt and a binary format reward. At each turn it regenerates Goal, Satisfied, Unsatisfied, and Stage from the task instruction, current observation, admissible actions where available, and retained observation–action history. No fixed milestone list or Stage vocabulary is supplied, and previous ARC reasoning or commitments are *not* included in the history. ALFWorld retains up to 50 previous turns (within its 50-step horizon), WebShop retains the most recent five turns, and WebArena truncates oldest history to the 2,048-token prompt limit while retaining the task instruction and current observation. The local outcome commitment specifies the intended next observable condition; it need not complete the current Stage.

Reasoning–observation inconsistency. The reported inconsistency metric is computed post hoc using a frozen LLM evaluator and a *separate binary-labeling prompt*, rather than the training process-reward rubric. It receives the task, history, observation, admissible actions, thought, and action, and flags a turn only when its thought contradicts current or previously observed environment state or task progress. Mean per-episode counts aggregate these binary flags. For persistence, each trajectory with a flagged turn contributes the 15 positions starting at its first flagged turn; positions after termination count as zero. Persistence is the fraction of flagged positions in these fixed windows, pooled within each reported trajectory group. Diagnostic labels do not enter policy training.

Scope in stochastic or interactive settings. ARC has not been evaluated with stochastic transitions or open-ended user responses. In such settings, scoring a single realized next observation can conflate outcome uncertainty with action quality; uncertainty-aware commitments and evaluation remain future work.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state our contributions: (1) identifying reasoning-observation inconsistency as a failure mode, (2) the ARC framework with structured commitments and process supervision, and (3) empirical improvements on three benchmarks. All claims are supported by experimental results in Section 4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss limitations after the conclusion, including reliance on an external LLM judge during training, evaluation on text-only environments, and the formatting overhead of the structured schema.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: This paper is primarily empirical and does not include theoretical results or proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide full training hyperparameters in Appendix A, describe the commitment schema and judge prompts in the method section, and use publicly available benchmarks (ALFWorld, WebShop, WebArena) and base models (Qwen2.5-7B-Instruct).

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The ALFWorld, WebShop, and WebArena benchmark environments are publicly available. A public link to our training code is not provided in this manuscript; implementation details and judge criteria are documented in the appendix.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Training details including learning rate, batch size, rollout parameters, and judge configuration are provided in Appendix A. Evaluation protocols follow prior work and are described in Section 4.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Due to the high computational cost of RL training (28+ GPU-hours per run), we report single-run results following the convention of prior work in this area (GRPO, GiGPO, VAGEN). We evaluate on the full benchmark test sets to reduce variance.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report wall-clock training time and compute overhead in Section 4.4 and Appendix B, including GPU type and training duration for each benchmark.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This research conforms with the NeurIPS Code of Ethics. We use publicly available benchmarks and open-source models, and do not involve human subjects.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [N/A]

Justification: This work improves RL training methodology for LLM agents in simulated environments (household tasks, web shopping). We do not foresee direct negative societal impacts beyond those generally associated with LLM agent development.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: Our work fine-tunes existing open-source models on simulated benchmarks and does not introduce models with high misuse risk beyond that of the base models.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We cite all benchmarks (ALFWorld, WebShop, WebArena) and models (Qwen2.5, Qwen3) used. All are publicly released under permissive licenses (Apache 2.0 for Qwen models, MIT for benchmarks).

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We do not release new datasets or pre-trained models in this submission.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing or human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

Justification: This work does not involve human subjects research.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: We use Qwen3-235B-A22B-FP8 as a frozen LLM judge to provide turn-level process reward signals during training. This is a core component of our method and is described in detail in Section 3.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.