REACH OR SOLVE? DEEP DIVING INTO AGENTIC RL GAINS WITH CHECKPOINT HANDOFFS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning (RL) is widely used to improve language-model agents, and its gains are usually measured by final task success. However, an agent's earlier actions shape the states in which its later decisions are made, so final task success conflates the ability to reach useful states with the ability to complete the task once there. Comparing agents only on the states each one reaches does not separate the two, since each agent is then scored on states selected by its own actions. To address this conflation, we introduce checkpoint handoff, an evaluation protocol that decouples reaching from completing without retraining. One checkpoint acts as a *reacher* up to a handoff point, and another continues as the *solver* from the same replayed history. In detail, (1) REACH measures how often a reacher arrives at states that a replayable environment verifies to be a fixed number of actions from success, and (2) SOLVE measures how often a solver completes the task from identical cloned copies of those states. Crossing supervised fine-tuning (SFT) and RL checkpoints in both roles across two benchmarks and two independently released training pipelines, we find that the gain from switching the solver from SFT to RL is consistently larger when RL is the reacher, at all three model scales on TravelPlanner and on both ALFWorld splits. Further analyses on ALFWorld show that RL improves both REACH and SOLVE. The solver gain is larger under an RL reacher because RL reaches solvable states more often, and separately measured REACH and SOLVE gaps recover most of this difference. Because handoff only requires replaying one checkpoint's history under another, agentic RL evaluations can report arrival and completion alongside final success.

1 INTRODUCTION

Language-model agents have rapidly become a central application of foundation models. They search, call tools, and interact with environments over multiple turns (Xie et al., 2024; Yao et al., 2025; Shridhar et al., 2021). As these agents take on longer and more interactive tasks, the methods used to train them have evolved accordingly (Chu et al., 2025; Zhang et al., 2026). Specifically, supervised fine-tuning (SFT) first develops these abilities by imitating demonstrations. AgentTuning mixes agent interaction trajectories with general instruction data (Zeng et al., 2024), while Agent-FLAN redesigns the training corpus to separate format following from agent reasoning (Chen et al., 2024). More recently, agentic reinforcement learning (RL) improves a policy through interaction and reward, with work studying trajectory-level optimization and stability (RAGEN), transition-level credit assignment (Agent Lightning), progressively longer interaction horizons (AgentGym-RL), and reusable skill libraries (SkillRL) (Wang et al., 2025; Luo et al., 2025; Xi et al., 2026; Xia et al., 2026). Yet these advances leave unclear whether RL's gains over SFT come from reaching more useful states, acting better from a given state, or both. This distinction matters for understanding what RL learns and deciding which part of training or evaluation to improve.

The difficulty arises because an agent's later inputs depend on its earlier actions. In a closed loop, actions change the environment and hence subsequent observations (Sutton & Barto, 2018). SFT and RL checkpoints can thus receive the same task while generating different histories and being scored from different states. In Figure 1, one checkpoint reaches the wrong cabinet with raw beef, while the other has already cooked and placed it. We call this policy dependence of evaluation inputs the *endogenous state problem*. Endpoint comparisons change both the states reached and the actions taken from them, leaving the source of a gain unresolved.

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

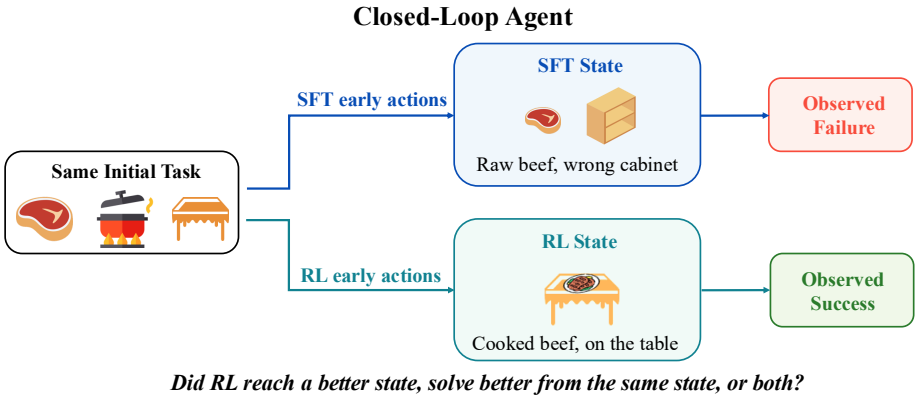

Figure 1: **The endogenous state problem.** Two checkpoints receive one task and act. Their early actions leave them in different states, and each is then scored from the state it produced.

Several evaluation families, reviewed more fully in Appendix A, look inside an endpoint. Progress evaluators score subgoal completion, process evaluators assess intermediate decisions and procedural compliance, and proxy-state methods verify the environment state left by a tool call (Ma et al., 2024; Gritta et al., 2026; Chuang et al., 2026). Exploration analyses measure entry into useful regions, while failure diagnosis separates errors made before and after a capability becomes exercisable (Ye et al., 2026; Ji et al., 2026; Shao et al., 2026). Causal decompositions further attribute outcomes to actions, transitions, or perception (Triantafyllou et al., 2025; Wang et al., 2026a). **These methods provide useful descriptions of a checkpoint's realized rollout, but they do not reveal how another checkpoint would continue from the same state and history.** Comparing policies only on the states they reach does not fix this. Arrival depends on both the policy and task difficulty, so such filtering mixes solver ability with the difficulty of the surviving tasks (Hernán et al., 2004). Restricting to tasks that both policies reach narrows the target to a selected subset, and even then the two policies need not arrive with the same history or budget. Attributing the overall gain therefore requires comparing solvers from identical states while retaining non-arrivals in the population-level comparison.

To address this attribution gap, we propose *checkpoint handoff*, which intervenes on who continues from a state. Specifically, we introduce two roles. A *reacher* acts first and determines which states the agent arrives at, and a *solver* then continues from a state held fixed and determines whether the task is finished. We cross these roles over $\{\mathrm{SFT}, \mathrm{RL}\}$ without retraining. At an environment-verified frontier, we clone the environment, history, and remaining budget and let both solvers continue from identical inputs. REACH measures arrival frequency, while SOLVE measures completion conditional on arrival. Non-arrivals remain failures in the full-population endpoint comparison. This intervention separates policy-induced state visitation from state-conditional value (Kakade & Langford, 2002) on released checkpoints. Profiles measured on disjoint data then test whether the two components account for how much larger the solver gain is under an RL reacher.

Experimentally, we systematically study SFT/RL checkpoint pairs released by two independent training pipelines on TravelPlanner and ALFWorld. We organize the comparison around three questions. **RQ1** asks whether the gain from switching the solver from SFT to RL depends on whether the reacher is SFT or RL. **RQ2** asks whether RL's advantage over SFT comes from reaching the solvable frontier more often, solving better from identical states, or both. **RQ3** asks whether independently measured SFT/RL differences in REACH and SOLVE account for the size of the RQ1 interaction.

This paper makes three contributions.

- We characterize the endogenous state problem of closed-loop agents by decomposing endpoint success into state visitation and state-conditional value, and show that neither endpoint comparison nor filtering to reached states can attribute an RL gain.
- We propose checkpoint handoff, which separates REACH from SOLVE on released checkpoints without retraining, using an environment-verified frontier, cloned-state continuation, and an out-of-sample test that recombines independent REACH and SOLVE gaps.

- Experiments on two benchmarks and two pipelines show a positive reacher–solver interaction in every condition, and ALFWorld replay finds RL gains in both REACH and SOLVE.

2 THE ENDOGENOUS STATE PROBLEM

2.1 SETTING AND ESTIMAND

Let a task T be drawn from a probability distribution $p_{\mathcal{T}}$. At step t, let x_t be the environment configuration, o_t the observation, a_t the action, $h_t = (o_0, a_0, \ldots, a_{t-1}, o_t)$ the visible history, and b_t the remaining action budget. We call

$$\Sigma_t = (T, x_t, h_t, b_t) \in \mathcal{S} \tag{1}$$

the evaluation state: the input from which the rest of the episode is scored. The checkpoint sees only h_t; a replay-capable evaluator also retains x_t. The episode ends at a step t_{term} by submission, termination, or budget exhaustion, and a task verifier $g_T : \mathcal{S} \to \{0, 1\}$ scores the endpoint $Y = g_T(\Sigma_{\text{term}})$, where $\Sigma_{\text{term}} = \Sigma_{t_{\text{term}}}$. Let $\mathcal{C} = \{\text{SFT}, \text{RL}\}$ index released checkpoints and $\pi_c(a \mid h)$, $c \in \mathcal{C}$, their induced policies on any visible history h. Decoding is fixed, so we intervene on c and read the consequences off π_c. For a fixed state $\sigma \in \mathcal{S}$ and a random continuation seed, let $Y(R; \sigma) \in \{0, 1\}$ be the potential endpoint when checkpoint $R \in \mathcal{C}$ continues from σ to termination, in the sense of potential outcomes (Rubin, 1974); its mean $V_R(\sigma) = \Pr(Y(R; \sigma) = 1)$ is the continuation value, a property of the state and the solver alone. For a fixed handoff distribution, solver performance is its mean continuation value.

We define a handoff rule as a prespecified map from the reacher's trajectory to a cut τ, selected independently of solver assignment and continuation outcomes. Running $W \in \mathcal{C}$ alone up to τ induces a distribution d_W^τ over Σ_τ, and we let a second checkpoint R continue. The reacher W sets the measure, the solver R sets the integrand,

$$J_\tau(W, R) = \mathbb{E}_{\Sigma \sim d_W^\tau}[V_R(\Sigma)]\,. \tag{2}$$

Assignments with $W = R$ provide same-checkpoint controls; those with $W \neq R$ switch checkpoints at handoff.

The attribution question. The diagonal controls run one checkpoint in both roles and report $J_\tau(c, c)$. Writing J for J_τ, the RL gain then splits into a reach term and a solve term along either of two paths, by the same telescoping that underlies the performance difference lemma (Kakade & Langford, 2002):

$$\begin{aligned} J(\text{RL}, \text{RL}) - J(\text{SFT}, \text{SFT}) &= \underbrace{J(\text{RL}, \text{RL}) - J(\text{SFT}, \text{RL})}_{\text{reach, under the RL solver}} + \underbrace{J(\text{SFT}, \text{RL}) - J(\text{SFT}, \text{SFT})}_{\text{solve, on SFT states}} \\ &= \underbrace{J(\text{RL}, \text{SFT}) - J(\text{SFT}, \text{SFT})}_{\text{reach, under the SFT solver}} + \underbrace{J(\text{RL}, \text{RL}) - J(\text{RL}, \text{SFT})}_{\text{solve, on RL states}}\,. \end{aligned} \tag{3}$$

Every right-hand term is a cross assignment in which one checkpoint continues from states produced by the other. Endpoint evaluation observes only $J_\tau(c, c)$, so it leaves d_W^τ and V_R confounded; the two paths can therefore disagree. The three research questions follow directly: RQ1 varies the reacher while measuring the solver gain, RQ2 attributes RL's advantage to reach and solve, and RQ3 tests the interaction prediction below.

2.2 ENDPOINT COMPARISONS CHANGE TWO FACTORS AT ONCE

The same-checkpoint contrast on the left of Equation 3 changes the measure and integrand together, so it cannot attribute the difference to either. Evaluating all four assignments instead yields the risk-difference interaction of a two-by-two factorial design (VanderWeele, 2015),

$$\begin{aligned} I_\tau = &\big[J_\tau(\text{RL}, \text{RL}) - J_\tau(\text{RL}, \text{SFT})\big] \\ &- \big[J_\tau(\text{SFT}, \text{RL}) - J_\tau(\text{SFT}, \text{SFT})\big]. \end{aligned} \tag{4}$$

I_τ is the difference between the two solve terms of Equation 3, and equally between the two reach terms. A positive I_τ means that switching the solver to RL gains more when RL is also the reacher, so the two attribution paths disagree. It does not say whether this comes from how often each reacher delivers a solvable state or from how the solver gap varies across states (Section 3.2).

2.3 Arrival Diverges Long Before Success Does

Figure 2 shows that the two checkpoints already diverge by the third reacher action; by the end of the budget, RL reaches the solvable frontier about six times as often as SFT. The curves record environment membership in $\mathcal{F}_2$, the set of states exactly two valid actions from success with enough budget to complete them (the general $\mathcal{F}_D$ is defined in Section 3.1). This verified frontier gives the common boundary for solver comparison in Section 3. Endpoint success still combines arrival and completion, whereas reached-only conditioning changes the target population; Section 2.4 and Appendix D quantify the resulting selection and budget effects.

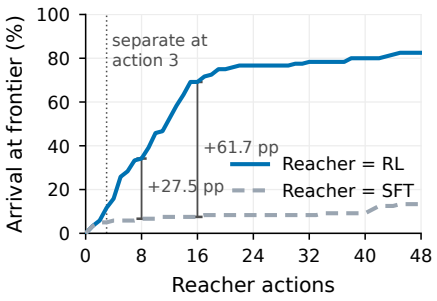

Figure 2: **Arrival at $\mathcal{F}_2$ by reacher action.** ALFWorld unseen split, 120 paired trajectories per checkpoint.

2.4 Keeping Only Reached States Changes the Estimand

Figure 3 evaluates the same ALFWorld episodes under two different estimands. The full-population estimand keeps every initial episode. An episode whose reacher never arrives has no continuation state and therefore counts as a failure. This quantity combines how often a reacher arrives with what a solver does after arrival. The reached-only estimand conditions on the reacher having arrived and averages only over the states that this reacher actually produced. It therefore measures conditional completion, not the overall endpoint gain, and it does not compare a common set of states across the two reachers.

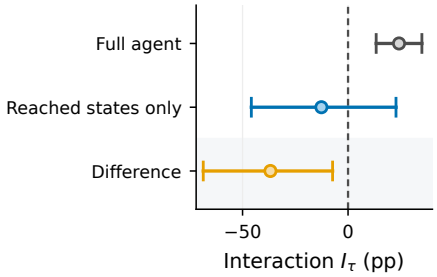

Figure 3: I_τ **under two estimands.** Same ALFWorld unseen-split episodes, scored two ways; 95% paired bootstrap.

The two estimands answer different questions. A reacher that arrives mainly on easy tasks can have a different conditional solver gain from one that also reaches harder tasks: the full-population estimand retains arrival frequency, whereas reached-only normalizes it away. In our data the full-population interaction is positive while the reached-only estimate has the opposite direction and a wide interval; Appendix D quantifies the support loss behind that interval.

Conditioning on arrival targets conditional completion, but using it to explain the population-level RL gain induces selection because arrival depends on the policy and task difficulty (Hernán et al., 2004). Restricting further to episodes that both policies reach changes the target again. The protocol therefore retains non-arrivals for the population endpoint and uses cloned states for the conditional solver effect; the retention diagnostic appears in Appendix D.

3 Methodology: Checkpoint Handoff

Checkpoint handoff crosses released reachers and solvers without retraining (Equation 2). Solvers receive no label identifying the reacher. Each solver continues live from the restored handoff state, so its actions change the environment as they would in deployment. Figure 4 summarizes the design and the ALFWorld prediction test; Appendix B, Table 5 maps the controls to the quantities they identify.

3.1 Handoff at Equal Remaining Distance

For replayable environments such as ALFWorld, we align handoffs by remaining distance: after the same number of actions, one reacher may be one action from success and another ten. Distance to success is used as a goal-conditioned value (Kaelbling, 1993), a proximity measure in embodied

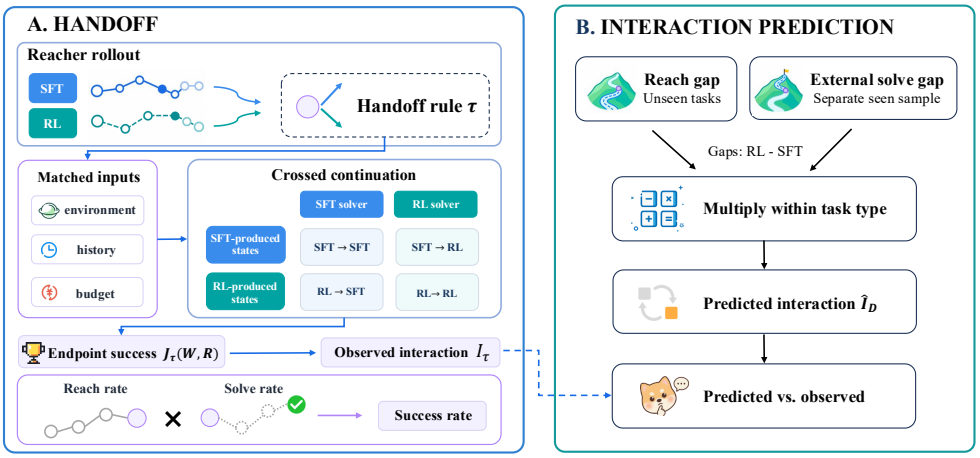

Figure 4: **Checkpoint handoff and interaction prediction. (A)** A benchmark-specific rule selects a handoff from each reacher's trajectory. Both solvers continue with matched inputs, yielding four endpoint success rates and their interaction. **(B)** On ALFWorld, target arrival gaps and independent SOLVE gaps give a predicted interaction, compared with the observed value (Equation 10).

navigation evaluation (Anderson et al., 2018), and a reward-shaping potential (Ng et al., 1999). We use it to index states, not to reward them. For any valid action sequence $\mathbf{a}$ from state σ, let $\Phi_T(\sigma, \mathbf{a})$ be the state obtained by exact replay and $|\mathbf{a}|$ its length. The remaining distance, computed by replay in the environment, is

$$\ell_T(\sigma) = \min_{\mathbf{a} \text{ valid from } \sigma} \{|\mathbf{a}| : \; g_T(\Phi_T(\sigma, \mathbf{a})) = 1\}, \tag{5}$$

with $\ell_T(\sigma) = \infty$ if no successful sequence exists. For $D \geq 1$ the solvable frontier is

$$\mathcal{F}_D(T) = \{\sigma \in \mathcal{S} : \; \ell_T(\sigma) = D, \; b(\sigma) \geq D\}, \tag{6}$$

where $b(\sigma)$ is the budget component of σ. Membership depends only on the environment, never on which checkpoint produced the state: every retained state is exactly D valid actions from success and has budget to get there. With early budget B_{early}, the handoff is the first hitting time $\tau_D = \inf\{t \leq B_{\text{early}} : \Sigma_t \in \mathcal{F}_D(T)\}$, and an empty set yields the non-arrival $\perp$ with $V_R(\perp) = 0$. τ_D depends on the reacher alone, as Section 2.1 requires; we write $J_D = J_{\tau_D}$ and $I_D = I_{\tau_D}$. We use $D = B_{\text{cont}} = 2$, the shortest horizon that lets a solver act, observe feedback, and act again. Enumerating one- and two-action continuations certifies this distance against the environment's success flag (Appendix B). Every retained state requires two actions to succeed, and both solvers receive exactly that budget.

3.2 REACH AND SOLVE FROM CLONED STATES

Let $Z_W = \mathbf{1}\{\Sigma_{\tau_D} \neq \perp\}$ indicate that reacher W arrived. REACH is the arrival rate $A_W = \Pr(Z_W = 1)$, the share of episodes in which W reaches the frontier at all. Non-arrivals are neither discarded nor imputed. Conditional on arrival, and defined when $A_W > 0$, let q_W^D be the natural frontier-state distribution, namely $d_W^{\tau_D}$ conditioned on $\Sigma_{\tau_D} \neq \perp$. We restore each reached state twice and assign the SFT and RL solvers under identical environment, history, admissible actions, budget, and seed schedule. We pair seeds because seed choice alone moves reported agent performance by a wide margin (Henderson et al., 2018). SOLVE is the conversion rate $C_{W,R}^D = \mathbb{E}_{\Sigma \sim q_W^D}[V_R(\Sigma)]$, the share of those arrivals that solver R turns into success. With $V_R(\perp) = 0$ the endpoint is the product of the two rates,

$$J_D(W, R) = A_W \, C_{W,R}^D, \tag{7}$$

so the reacher moves the first factor and the solver the second. Matching the environment state and visible history fixes task progress and available information; switching only the continuation checkpoint identifies the same-state SOLVE gap $\delta(\sigma) = V_{\text{RL}}(\sigma) - V_{\text{SFT}}(\sigma)$, and

$$I_D = A_{\text{RL}} \, \mathbb{E}_{\Sigma \sim q_{\text{RL}}^D}[\delta(\Sigma)] - A_{\text{SFT}} \, \mathbb{E}_{\Sigma \sim q_{\text{SFT}}^D}[\delta(\Sigma)]. \tag{8}$$

With $\bar{\delta}_W = \mathbb{E}_{\Sigma \sim q_W^D}[\delta(\Sigma)]$ the mean SOLVE gap on states produced by reacher W,

$$I_D = \underbrace{(A_{\mathrm{RL}} - A_{\mathrm{SFT}})\,\bar{\delta}_{\mathrm{RL}}}_{\text{arrival}} + \underbrace{A_{\mathrm{SFT}}\,(\bar{\delta}_{\mathrm{RL}} - \bar{\delta}_{\mathrm{SFT}})}_{\text{state}}. \tag{9}$$

The arrival term is positive whenever RL both reaches and solves better. The state term is positive only if the RL solver's advantage grows on RL-produced states. For a prespecified set of states μ_D, chosen independently of solver assignment and outcomes, we also report $\Delta_{\mathrm{solve}}(\mu_D) = \mathbb{E}_{\Sigma \sim \mu_D}[\delta(\Sigma)]$. We report δ separately on SFT-produced and RL-produced states, so a SOLVE gain cannot be an artefact of RL states being easier.

3.3 PREDICTING INTERACTION FROM REACH AND SOLVE GAPS

Equation 7 is an identity on the same natural states. A stronger test forecasts the interaction from components measured elsewhere, before the target continuations are run. Let $k \in \mathcal{K}$ index task strata with prevalence p_k. Natural rollouts on the target split give the arrival rate $A_W(k) = \Pr(Z_W = 1 \mid k)$. An independently selected set of $\mathcal{F}_D$ states gives each solver's SOLVE rate $C_R^{\mathrm{ext}}(k)$ on states the target checkpoints never visited; its selection and scoring never touch target continuation outcomes. Under this component model, the interaction is the prevalence-weighted product of the two gaps,

$$\widehat{I}_D^{\mathrm{comp}} = \sum_k p_k \left[A_{\mathrm{RL}}(k) - A_{\mathrm{SFT}}(k)\right]\left[C_{\mathrm{RL}}^{\mathrm{ext}}(k) - C_{\mathrm{SFT}}^{\mathrm{ext}}(k)\right]. \tag{10}$$

The residual $R_{\mathrm{comp}} = I_D - \widehat{I}_D^{\mathrm{comp}}$ is a forecast error: nothing in Equation 10 is estimated from the observed I_D. Its transport condition, in the sense of Pearl & Bareinboim (2011), is that within k the external rate approximates continuation success on naturally reached target states regardless of their reacher. The per-type residuals quantify how well this condition holds across the target task distribution.

4 EXPERIMENTS

4.1 SETUP

TravelPlanner (Xie et al., 2024) tests RQ1 across model scales using recorded-history handoffs. ALFWorld (Shridhar et al., 2021) supports verified-frontier handoffs for RQ2 and RQ3. Agent-STAR (Wu et al., 2026) and SkillRL (Xia et al., 2026) are independent Qwen-family training pipelines. Decoding settings were identical within each SFT/RL pair and fixed before any handoff ran (Appendix B).

TravelPlanner and Agent-STAR. We cross released 1.5B, 3B, and 7B Agent-STAR SFT/RL checkpoints on the same 180 TravelPlanner validation tasks; their immutable revisions are listed in Appendix B, Table 4. TravelPlanner's two-stage setting separates information collection from plan generation over a static database (Xie et al., 2024). Agent-STAR records tool interactions in a message history and terminates with a submitted plan (Wu et al., 2026). We therefore use the recorded trajectory as the handoff input: task prompt, model messages, tool calls, and tool responses. Each reacher runs to termination under the native 60-decision limit; we cut immediately before its final model call. This retains the reacher's accumulated information and intermediate reasoning while leaving plan completion to the solver. The cut depends only on the recorded reacher run. We reset the task, replay the prefix, and verify identical reconstructed histories. Both solvers receive the same record and at most $B = 4$ model decisions, allowing further tool use before submission under a common budget. A fixed Formatter converts the result for the official evaluator. Crossing reachers and solvers gives four assignments, each evaluated on the same 180 tasks, or 720 rows per scale (Appendix B).

ALFWorld and SkillRL. ALFWorld's TextWorld engine represents household states and action effects symbolically (Shridhar et al., 2021). We use its admissible actions and success flag to verify remaining distance by enumeration. The unseen split places tasks in rooms held out from training, and the seen split reuses training rooms. We run released SkillRL 7B SFT/RL checkpoints on 30 prespecified unseen-split tasks, five from each official task type, with four paired seeds. REACH

comes from natural rollouts: a reacher acts for at most 48 steps, and after each action the environment is asked whether the current state lies on $\mathcal{F}_2$ (Section 3.1). SOLVE comes from replay. We reset the task under the same seed and replay the recorded reacher actions to restore the arrival state. Observation, admissible actions, and rendered prompt must match the recording. Each solver then takes two actions from that clone, in a separate replay of the same state. This yields 240 natural trajectories, 115 verified frontier states, and 480 endpoint rows, scored by the environment's native `won` flag. A separately selected seen-split sample of the same design supplies the external SOLVE profile (Appendix C) and a replication. All replay, solvability, pairing, prompt, and completeness checks pass (Appendix B).

Inference. Arrival and endpoint rates are empirical means over prespecified tasks and paired seeds, with zero endpoint success for every non-arrival. All uncertainty uses 100,000 task-level paired bootstrap resamples (Efron & Tibshirani, 1994) that keep seeds, the four checkpoint assignments, and task-type counts together. Tasks are the unit that varies, and few-run evaluations call for interval estimates rather than point scores (Agarwal et al., 2021). For the component-based prediction, each draw independently samples tasks within each task type from the target and external samples, retains all paired seeds for every sampled task, and recomputes the observed interaction, predicted interaction, and residual from that draw. On ALFWorld, same-state contrasts identify the solver effect because the environment decides τ_D before any solver runs, both solvers start from exact clones, and only the solver changes.

4.2 RQ1: THE SOLVER GAIN FROM SFT TO RL DEPENDS ON THE REACHER

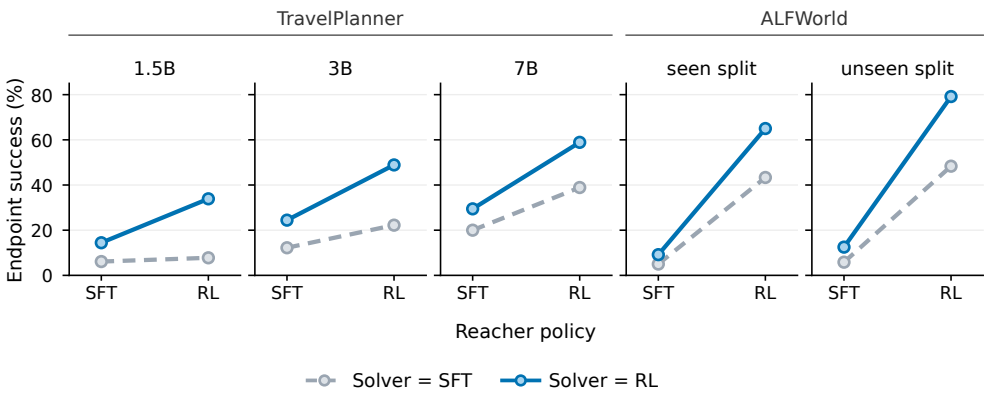

Figure 5: **Endpoint success for the four reacher by solver assignments.** Non-parallel lines indicate a reacher–solver interaction.

Table 1: **Endpoint success for the four reacher (W) by solver (R) assignments.** Subscripts on the R=RL columns give the solver effect under that reacher, and the last column is their difference, I_τ of Equation 4. Brackets are 95% paired-bootstrap intervals.

Setting	n	Endpoint success (%) W=SFT R=SFT	W=SFT R=RL	W=RL R=SFT	W=RL R=RL	Interaction (pp)
TravelPlanner / Agent-STAR						
1.5B	180	6.1	$14.4_{+8.3}$	7.8	$\mathbf{33.9}_{+26.1}$	+17.8 [9.4, 26.1]
3B	180	12.2	$24.4_{+12.2}$	22.2	$\mathbf{48.9}_{+26.7}$	+14.4 [6.1, 22.8]
7B	180	20.0	$29.4_{+9.4}$	38.9	$\mathbf{58.9}_{+20.0}$	+10.6 [0.6, 20.6]
ALFWorld / SkillRL						
Seen split	120	5.0	$9.2_{+4.2}$	43.3	$\mathbf{65.0}_{+21.7}$	+17.5 [6.7, 27.5]
Unseen split	120	5.8	$12.5_{+6.7}$	48.3	$\mathbf{79.2}_{+30.8}$	+24.2 [13.3, 35.0]

The Solver Effect Is Two to Five Times Larger Under the RL Reacher Table 1 compares the SFT and RL solvers separately under each reacher. In Figure 5 the vertical gap between the two lines

is the solver effect, so a gap that changes across reachers indicates an interaction. The interaction is positive in all five conditions, and every lower bound in Table 1 lies above zero. The size of this gap is the informative part. On TravelPlanner, switching to the RL solver gains between two and three times as much on RL-produced histories as on SFT-produced ones, at every scale. On ALFWorld the ratio rises to about five. The SFT reacher rarely hands the solver a finishable state, and no SFT-reacher entry exceeds 12.5% whichever solver continues. A better solver pays off only on episodes that reach a solvable state, and on ALFWorld the larger gain comes entirely from arrival (Sections 4.3–4.5).

With Scale, the Gain Moves From the Solver to the Reacher At 1.5B the SFT reacher with the RL solver beats the RL reacher with the SFT solver, 14.4% against 7.8%; an RL history is worth little to an SFT solver. At 7B the order flips, 29.4% against 38.9%, so the RL history alone now carries most of the gain. The interaction shrinks from 1.5B to 7B, but the three intervals overlap and the 7B lower bound sits close to zero. We read the decline as an observation, not a trend. The intervals support a positive interaction at every scale and under two independently trained pipelines.

4.3 RQ2A: RL Reaches the Frontier More Often than SFT

RL Reaches an Equally Solvable State Six Times as Often On ALFWorld each entry of Table 1 equals the arrival rate in Table 2(a) times the completion rate in Table 2(b), to the reported precision. Equation 2 is an identity here, and the question is how much of the gain sits in each term. On the unseen split the SFT reacher reaches $\mathcal{F}_2$ in 13.3% of trajectories and the RL reacher in 82.5%. Every frontier state is exactly two actions from success, so the contrast counts arrivals at a common boundary. Mean actions to arrival or stopping, 44.0 against 17.8, separate two regimes: the SFT reacher usually runs to the 48-action limit without arriving, the RL reacher usually arrives within the first third of the budget. The gap replicates on the seen split, 15.8% against 81.7%, and REACH is the component on which the two splits agree most closely (Figure 6). Appendix D reports the split replication, budget contrast (Figure 10 and Table 7), and non-arrival audit: RL with eight actions already out-reaches SFT with forty-eight, and almost every SFT non-arrival is a loop that repeats actions without changing the state. More budget does not close the deficit.

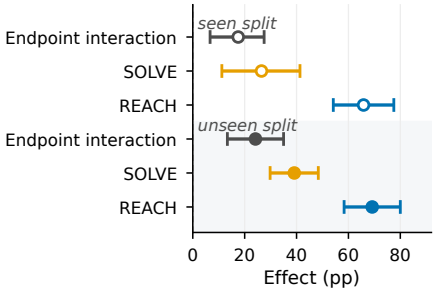

Figure 6: **REACH, SOLVE, and the endpoint interaction.** Filled markers are the unseen split, open the seen split.

Table 2: **REACH and SOLVE on ALFWorld.** Each row is a reacher. (a) Arrival at $\mathcal{F}_2$ over 120 trajectories, with mean actions to arrival or stopping. (b) Completion by each solver from identical replayed states that the reacher produced. Brackets are task-stratified paired-bootstrap 95% intervals; a dash marks a stratum too small for one.

		(a) REACH		(b) SOLVE from identical replayed states			
Split	Reacher	Arrival %	Steps	States n	R=SFT %	R=RL %	RL − SFT (pp)
Seen split	SFT	15.8	41.6	19	31.6	57.9	26.3 —
	RL	81.7	15.7	98	53.1	79.6	26.5 —
	Pooled			117	49.6	76.1	26.5 [11.2, 41.3]
Unseen split	SFT	13.3	44.0	16	43.8	93.8	50.0 [15.4, 83.3]
	RL	82.5	17.8	99	58.6	96.0	37.4 [27.7, 47.0]
	Pooled			115	56.5	95.7	39.1 [29.8, 48.4]

4.4 RQ2B: RL Outperforms SFT from Identical States

The RL Solver Is Near Ceiling Whoever Produced the State From two exact clones of each of the 115 reached unseen-split states, the SFT solver completes 56.5% and the RL solver 95.7% (Table 2b). The clones share task, history, admissible actions, two-action budget, and seed schedule, so the gap is a solver effect, and its lower bound lies well above zero on both splits. The origin of

Table 3: **SOLVE split into choosing a bridge action and finishing from it.** Shares of reached states with 95% bootstrap intervals; the last two columns count states completed by only one solver ($n = 115$ on the unseen split, 117 on the seen split).

Split	Solver	Bridge %	Finish | bridge %	Complete %	RL only	SFT only
Seen split	SFT	72.6 [64.1, 80.3]	68.2 [57.6, 77.6]	49.6 [40.2, 59.0]	41	10
	RL	92.3 [87.2, 96.6]	82.4 [75.0, 88.9]	76.1 [68.4, 83.8]		
Unseen split	SFT	72.2 [63.5, 80.0]	78.3 [68.7, 86.7]	56.5 [47.8, 65.2]	45	0
	RL	97.4 [93.9, 100.0]	98.2 [95.5, 100.0]	95.7 [91.3, 99.1]		

the state matters only to the SFT solver: it completes 43.8% of the states its own reacher produced and 58.6% of the RL-produced ones, while the RL solver completes 93.8% and 96.0%. The same-state effect is no smaller on SFT-produced states (50.0 against 37.4 points), so the SOLVE gain is not an artefact of RL states being easier. Of the 115 states, 45 are won by the RL solver only and none by the SFT solver only (Table 3). The same table splits SOLVE into choosing a bridge action and finishing from it. SFT loses at both stages, and the RL advantage at the second action is the less stable across splits. Appendix D gives the corresponding action-stage breakdown. Appendix E (Table 8) walks through one state: holding the pencil, the SFT solver spends its first action looking around while the RL solver walks to a shelf and places it.

4.5 RQ3: PREDICTING REACHER–SOLVER INTERACTION FROM REACH AND SOLVE GAPS

Independent Reach and Solve Gaps Recover the Aggregate Interaction We test whether the interaction can be recovered from its two component gaps. For each task type, we multiply the RL–SFT difference in frontier arrival by the RL–SFT difference in continuation success, then average across the six balanced types. Arrival rates come from the target split, while continuation success is measured on an independent seen-split sample with no state or continuation outcome shared with the 115 replayed target states. The target `valid_unseen` states are reserved for evaluating the observed interaction; the independent `valid_seen` sample supplies the external SOLVE profile, making this an out-of-sample transport test across splits conditional on task type. Equation 10 predicts an interaction of 20.7 percentage points, compared with an observed value of 24.2 points (Figure 7, Appendix C). The black-diamond overall point is the unweighted mean of the six task-type predictions and observations, $(20.7, 24.2)$. The prediction recovers the positive interaction and its aggregate scale, with a residual of 3.5 points (95% CI $[-7.2, 14.2]$). Equation 9 agrees in sample, with arrival terms of 25.9 and 17.5 points and state terms of -1.7 and 0.0 on the unseen and seen splits, so the interaction on ALFWorld is carried by arrival. Appendix C gives the per-type comparison, including Figures 8 and 9, along with their source rates in Table 6.

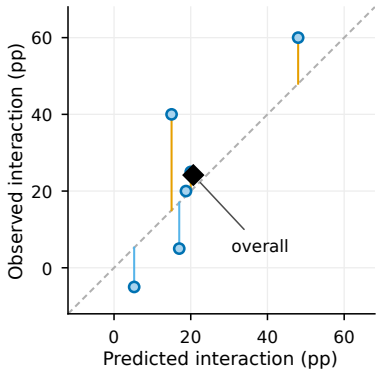

Figure 7: **Predicted against observed interaction, per task type.** The dashed line is $y = x$; the black diamond is the unweighted mean across the six task types.

5 CONCLUSION

An agent's earlier actions shape the environment states it later acts from, so its final success rate mixes two abilities: reaching useful states and completing the task once there. In this paper, we propose checkpoint handoff, an evaluation protocol that separates the two by letting one checkpoint reach a state and another continue from an exact copy of it. Experiments with SFT and RL checkpoints show that the RL solver's advantage is larger when RL is also the reacher. On ALFWorld, RL improves both abilities, and this larger advantage comes from reaching solvable states more often. Looking beneath final success in this way gives a clearer picture of what agentic RL changes.

AI USE STATEMENT

This study uses AI tools to write parts of the code, refine the idea, and adjust the LaTeX formatting.

REPRODUCIBILITY STATEMENT

Appendix B lists the checkpoints and revisions, task samples, generation settings, and replay checks. Appendix C describes the independent SOLVE sample, and Appendix D gives the seen-split replication and additional analyses. Code is available at `https://anonymous.4open.science/r/AgenticRL_Handoff-0F6A`. The supplementary material includes the configurations, continuation records, Formatter replies, rewards, and summary statistics used in the paper.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, 2021.

Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir R. Zamir. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018. doi: 10.48550/arXiv.1807.06757. URL `https://arxiv.org/abs/1807.06757`.

Zehui Chen, Kuikun Liu, Qiuchen Wang, Wenwei Zhang, Jiangning Liu, Dahua Lin, Kai Chen, and Feng Zhao. Agent-FLAN: Designing data and methods of effective agent tuning for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 9354–9366. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.557.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. SFT memorizes, RL generalizes: A comparative study of foundation model post-training. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 10818–10838. PMLR, 2025. URL `https://proceedings.mlr.press/v267/chu25c.html`.

Yun-Shiuan Chuang, Chaitanya Kulkarni, Alec M. Chiu, Avinash Thangali, Zijie Pan, Shivani Shekhar, Yirou Ge, Yixi Li, Uma Kona, Linsey Pang, and Prakhar Mehrotra. Toward scalable verifiable reward: Proxy state-based evaluation for multi-turn tool-calling LLM agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics: Industry Track*, pp. 1251–1264. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-industry.87.

Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall/CRC, 1994.

Ashritha Gonuguntla. The replay gap: Static evaluation of model switching in LLM agents scores the wrong world. In *Efficient Reasoning Workshop at COLM*, 2026. URL `https://openreview.net/forum?id=8gqqiNrzyA`.

Milan Gritta, Debjit Paul, Xiaoguang Li, Lifeng Shang, Jun Wang, and Gerasimos Lampouras. Process evaluation for agentic systems. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 2678–2692. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-eacl.140.

Langzhou He, Junyou Zhu, Yue Zhou, Zhengyao Gu, Junhua Liu, Wei-Chieh Huang, Henry Peng Zou, David Wipf, Philip S. Yu, and Qitian Wu. Resolving action bottleneck: Agentic reinforcement learning informed by token-level energy. *arXiv preprint arXiv:2605.14558*, 2026.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *AAAI Conference on Artificial Intelligence*, 2018.

Miguel A. Hernán, Sonia Hernández-Díaz, and James M. Robins. A structural approach to selection bias. *Epidemiology*, 15(5):615–625, 2004.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020.

Yifu Huo, Shunjie Xing, Chenglong Wang, Peinan Feng, Qiaozhi He, Yan Ding, Anxiang Ma, Yuxin Gao, Tongran Liu, Tong Xiao, and Jingbo Zhu. Learning from environmental feedback: Credit assignment across multiple timescales for agentic reinforcement learning. *arXiv preprint arXiv:2608.08255*, 2026.

Yuxiang Ji, Zengbin Wang, Yong Wang, Shidong Yang, Ziyu Ma, Guanhua Chen, Zonghua Sun, Liaoni Wu, and Xiangxiang Chu. Learning agentic policy from action guidance. *arXiv preprint arXiv:2605.12004*, 2026.

Hangzhan Jin, Sitao Luan, Tianwei Ni, Sicheng Lyu, Guillaume Rabusseau, Reihaneh Rabbany, Doina Precup, and Mohammad Hamdaqa. RL fine-tuning heals OOD forgetting in SFT. *arXiv preprint arXiv:2509.12235*, 2025. Presented at the NeurIPS 2025 Workshop on Foundations of Reasoning in Language Models.

Leslie Pack Kaelbling. Learning to achieve goals. In *International Joint Conference on Artificial Intelligence*, pp. 1094–1099, 1993.

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the 19th International Conference on Machine Learning*, pp. 267–274, 2002.

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.

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. AgentBoard: An analytical evaluation board of multi-turn LLM agents. In *Advances in Neural Information Processing Systems*, volume 37, pp. 74325–74362, 2024. doi: 10.52202/079017-2365.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *International Conference on Machine Learning*, pp. 278–287, 1999.

Judea Pearl and Elias Bareinboim. Transportability of causal and statistical relations: A formal approach. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence*, pp. 247–254, 2011.

Qihan Ren, Peng Wang, Ruikun Cai, Shuai Shao, Dadi Guo, Yuejin Xie, Yafu Li, Quanshi Zhang, Xia Hu, Jing Shao, and Dongrui Liu. Rethinking generalization in reasoning SFT: A conditional analysis on optimization, data, and model capability. *arXiv preprint arXiv:2604.06628*, 2026. Presented at the FoGen Workshop at ICML 2026 (non-archival).

Donald B. Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology*, 66(5):688–701, 1974.

Wei Shao, Chongzhou Fang, Zuxiong Tan, Zequan Liang, Setareh Rafatirad, Avesta Sasan, and Houman Homayoun. Beyond end-to-end success: Diagnosing failures in long-horizon security LLM agents. *arXiv preprint arXiv:2608.20563*, 2026.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Cote, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018.

Stelios Triantafyllou, Aleksa Sukovic, Yasaman Zolfimoselo, and Goran Radanovic. Counterfactual effect decomposition in multi-agent sequential decision making. In *Proceedings of the 42nd International Conference on Machine Learning*, pp. 60072–60098, 2025.

Tyler J. VanderWeele. *Explanation in Causal Inference: Methods for Mediation and Interaction*. Oxford University Press, 2015.

Haozhe Wang, Qixin Xu, Changpeng Wang, Taofeng Xue, Chong Peng, Wenhu Chen, and Fangzhen Lin. Bad seeing or bad thinking? rewarding perception for multimodal reasoning. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026a. URL https://icml.cc/virtual/2026/poster/62726. To appear.

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. doi: 10.48550/arXiv.2504.20073. URL https://arxiv.org/abs/2504.20073.

Zihan Wang, Chi Gui, Xing Jin, Qineng Wang, Licheng Liu, Kangrui Wang, Shiqi Chen, Linjie Li, Zhengyuan Yang, Pingyue Zhang, Yiping Lu, Jiajun Wu, Li Fei-Fei, Lijuan Wang, Yejin Choi, and Manling Li. RAGEN-2: Reasoning collapse in agentic RL. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026b. URL https://ragen-ai.github.io/. Oral presentation; to appear.

Xixi Wu, Qianguo Sun, Ruiyang Zhang, Chao Song, Junlong Wu, Yiyan Qi, and Hong Cheng. Demystifying reinforcement learning for long-horizon tool-using agents: A comprehensive recipe. In *Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing*, 2026. URL https://github.com/WxxShirley/Agent-STAR. To appear.

Zhiheng Xi, Jixuan Huang, Chenyang Liao, Baodai Huang, Jiaqi Liu, Honglin Guo, Yajie Yang, Rui Zheng, Junjie Ye, Jiazheng Zhang, Wenxiang Chen, Wei He, Yiwen Ding, Guanyu Li, Zehui Chen, Zhengyin Du, Xuesong Yao, Yufei Xu, Jiecao Chen, Tao Gui, Zuxuan Wu, Qi Zhang, Xuanjing Huang, and Yu-Gang Jiang. AgentGym-RL: An open-source framework to train LLM agents for long-horizon decision making via multi-turn RL. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=ZgCCDwcGwn.

Peng Xia, Jianwen Chen, Hanyang Wang, Jiaqi Liu, Kaide Zeng, Yu Wang, Siwei Han, Yiyang Zhou, Xujiang Zhao, Haifeng Chen, Zeyu Zheng, Cihang Xie, and Huaxiu Yao. SkillRL: Evolving agents via recursive skill-augmented reinforcement learning. In *ICLR 2026 Workshop on Memory for LLM-Based Agentic Systems*, 2026. URL https://openreview.net/forum?id=FYc2IygegR. Oral presentation.

Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and Yu Su. TravelPlanner: A benchmark for real-world planning with language agents. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54590–54613. PMLR, 2024. URL https://proceedings.mlr.press/v235/xie24j.html.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/1b126cc38b8638e07bef37e7b2bb72bf-Abstract-Conference.html.

Ziang Ye, Wentao Shi, Yuxin Liu, Yu Wang, Zhengzhou Cai, Yaorui Shi, Qi Gu, Xunliang Cai, and Fuli Feng. Look before you leap: Autonomous exploration for LLM agents. *arXiv preprint arXiv:2605.16143*, 2026.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. AgentTuning: Enabling generalized agent abilities for LLMs. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3053–3077. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.181.

Guibin Zhang, Hejia Geng, Xiaohang Yu, Zhenfei Yin, Zaibin Zhang, Zelin Tan, Heng Zhou, Zhong-Zhi Li, Xiangyuan Xue, Yijiang Li, Yifan Zhou, Yang Chen, Chen Zhang, Yutao Fan, Zihu Wang, Songtao Huang, Francisco Piedrahita Velez, Yue Liao, Hongru Wang, Mengyue Yang, Heng Ji, Jun Wang, Shuicheng Yan, Philip Torr, and Lei Bai. The landscape of agentic reinforcement learning for LLMs: A survey. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=RY19y2RI1O. Survey Certification.

A Related Work

Training and comparing SFT and RL agents. Multi-turn RL for agents optimizes whole trajectories. RAGEN treats the rollout as the unit of optimization and studies when training collapses, Agent Lightning decouples agent execution from training and assigns credit at the transition level, and AgentGym-RL lengthens the interaction horizon during training (Wang et al., 2025; Luo et al., 2025; Xi et al., 2026). SkillRL grows a recursive skill library, and Agent-STAR distills a training recipe for long-horizon tool use (Xia et al., 2026; Wu et al., 2026). Diagnostics of such runs localize reasoning collapse and token-level action bottlenecks (Wang et al., 2026b; He et al., 2026). Studies that compare the two regimes report that RL generalizes where SFT memorizes, that RL fine-tuning restores out-of-distribution ability lost during SFT, and that SFT generalization depends on data and model capability (Chu et al., 2025; Jin et al., 2025; Ren et al., 2026). These works measure whether and how much RL helps, each scoring a checkpoint on the states it reaches by itself. We take released checkpoints from two of these pipelines and ask which ability the gain comes from.

Agent evaluation beyond endpoint success. AgentBoard scores progress toward subgoals, process evaluation grades each step of an agentic trajectory, and proxy-state rewards verify the environment state a tool call leaves behind (Ma et al., 2024; Gritta et al., 2026; Chuang et al., 2026). Exploration analyses measure how often a policy enters useful regions, guide it there with action-level supervision, or assign credit across several timescales of environment feedback (Ye et al., 2026; Ji et al., 2026; Huo et al., 2026). Failure diagnosis separates errors made before a capability becomes exercisable from errors made once it is (Shao et al., 2026). Each of these describes the rollout of a single policy, so they either restrict attention to the states that policy reached (Section 2.4) or change the state distribution and the continuation value together (Section 2.2). Checkpoint handoff adds the comparison they lack: a second policy acting from the same state.

Counterfactual attribution of agent outcomes. Counterfactual effect decomposition attributes multi-agent outcomes to individual actions and environment transitions (Triantafyllou et al., 2025), and perception-aware rewards separate vision-language errors of perception from errors of reasoning (Wang et al., 2026a). Both decompose the outcome of a single model into parts of the process that produced it, whereas checkpoint handoff compares two policies acting from the same state. We are also aware of concurrent work on per-step model routing, which forks live agent trajectories and shows that predicting the outcome of a mid-episode model switch from each model's logged runs misjudges it (Gonuguntla, 2026). That work evaluates routers for efficiency, while ours focuses on what agentic RL training changes in a policy. Our claims concern behavior, not internal mechanisms.

B Reproducibility Details

Released checkpoints. We evaluate existing checkpoints without retraining them. Table 4 lists the immutable revisions used throughout the paper. Agent-STAR (Wu et al., 2026) provides three SFT/RL pairs for TravelPlanner, and SkillRL (Xia et al., 2026) provides one SFT/RL pair for ALFWorld. The two releases constitute independent training pipelines, but both use architectures from the broad Qwen family; the experiment therefore does not establish transfer across unrelated base-model architectures.

Table 4: Released checkpoints and fixed revisions.

Benchmark	Checkpoint	Revision
TravelPlanner	`xxwu/Agent-STAR-SFT-1.5B`	`b3d55c10e763231d13f591bfa2ad2cb99d19b981`
TravelPlanner	`xxwu/Agent-STAR-RL-1.5B`	`279b7c75586b2a6ada3eac5a5f406a147522c787`
TravelPlanner	`xxwu/Agent-STAR-SFT-3B`	`dfb0d943ad8c7a1d6a5e38064a75a6f4e8b4e503`
TravelPlanner	`xxwu/Agent-STAR-RL-3B`	`705723c62e82ed15844783f221572ea1c78f4185`
TravelPlanner	`xxwu/Agent-STAR-SFT-7B`	`945532c2ca454caaf1f62d1c42916cbec0a2973a`
TravelPlanner	`xxwu/Agent-STAR-RL-7B`	`7013d9fd0fff692d8f6f352ae5db4351d174eab9`
ALFWorld	`Jianwen/Alfworld-7B-SFT`	`ba9c962eef80a49fc63a94c9728209a36057c671`
ALFWorld	`Jianwen/Alfworld-7B-RL`	`2ce16cb90e6357892dde201928279d4513d35c59`

TravelPlanner protocol. For each scale, Val180 contains 60 easy, 60 medium, and 60 hard tasks. The primary analysis crosses two reachers and two solvers at budget $B = 4$, yielding 720 rows

per scale. Inference uses Python 3.10.20, PyTorch 2.6.0 with CUDA 11.8, vLLM 0.8.1 with CUDA 11.8, FP16, and a 32K-token context. We use nucleus sampling (Holtzman et al., 2020) with temperature 0.6, top-p 0.95, and seed 2027. Confidence intervals use 100,000 task-level paired bootstrap resamples. The archived Formatter alias is `deepseek-chat`; its raw replies are retained. All requests completed, and recomputing the official Agent-STAR evaluator produced zero per-example reward differences. The three formal terminal summaries pass their protocol acceptance checks.

TravelPlanner handoff construction. Each reacher first runs its released checkpoint under the native 60-decision and 60-tool limits. The handoff is the point immediately before that trajectory's final model call, rather than a common elapsed step. The rule is a function of the recorded reacher trajectory alone, so it satisfies Section 2.1, but its states are late rather than early. We reset the task and replay every preceding assistant action and tool response, then require the reconstructed message list to equal the recorded list. For each history, the SFT and RL solvers receive that entire list and the recorded final-turn seed; no message is summarized, reordered, or semantically repaired. The recorded first solver decision is the $B = 1$ snapshot. If it is a valid tool call, the evaluator executes it, appends its native tool response, and permits at most three further decisions, using consecutive paired seeds, to form $B = 4$. Answers, invalid actions, and context limits are absorbing. Thus $B = 4$ counts four post-handoff model decisions, not the four reacher–solver assignments. The Formatter is applied only after the solver terminates. At every scale, the $B = 4$ audit contains 720 unique rows, 180 per assignment, with equal paired input hashes, no Formatter request failures, and no official-score recomputation differences.

Formatter measurement audit. The fixed-handoff Val180 run initially contained two unique empty Formatter envelopes, corresponding to four endpoint rows. The separately registered repair protocol retried each input exactly once with the same model, prompt, JSON contract, and disabled-thinking setting, using no new trajectories and changing no checkpoint assignment; it writes to an isolated artifact. Both requests returned valid envelopes. The repaired directory contains 292 valid unique Formatter samples, 720 formatted rows, and 720 officially scored rows; the evaluator recomputation reported zero mismatches. On the 716 unaffected rows, all identity fields, formatted-plan hashes, official metrics, and r_{sum} are identical to the source artifact. Eight diagnostic `reward_details` messages differ only in the order of an unordered list, which is recorded separately and does not change any score. The fixed-handoff interaction remains 7.22 points with 95% CI $[1.67, 12.78]$, so the positive interaction is not explained by the two transient measurement failures.

ALFWorld protocol. The unseen-split sample (`valid_unseen` in the ALFWorld release; the seen-split sample is `valid_seen`) contains 30 tasks stratified across the six ALFWorld task types. Each released checkpoint receives 48 early environment steps. Natural arrival is measured at the environment-verified distance-two frontier $\mathcal{F}_2$. Every arriving state is replayed, verified, cloned, and continued by both solvers with two remaining environment steps and four paired seed slots. Inference uses vLLM 0.8.1, BF16, a 4,096-token prompt limit, a 512-token response limit. We use temperature-scaled multinomial sampling with temperature 0.4 and top-p 1 (no probability truncation), with selection seed 2029. The outcome is the environment's native `won` flag. The main protocol produces 115 verified frontier states and passes all replay, pairing, and completion checks.

ALFWorld solvability check and clone audit. For a candidate action prefix, the check first rejects terminal states, then replays every currently admissible action. It rejects the candidate if any first action wins; otherwise it replays every admissible second action from each nonterminal successor and accepts only if at least one two-action path wins. This exhaustive local search includes alternative shortest paths and states from which some actions are irreversible. Across the 115 accepted handoffs, it tested 3,227 first actions and 89,511 second actions, and never failed.

The environment is restored by deterministic reset with the same task seed and exact replay of the recorded reacher actions; we do not serialize a Python environment object. After reset and every reacher action, equality is checked on a JSON-safe record containing the observation, ordered admissible-action list, score, terminal flag, and `won` flag. At handoff, the visible history, rendered prompt, prompt-token count, and token-ID digest must also agree. The two solvers are run in separate replays from this operational clone. All 230 arrival-solver replays passed, all 115 solver pairs

Table 5: **Design questions and the protocol control that answers each.**

Design question	Protocol control	Reported quantity
Are states equally close to success?	Exhaustive $\mathcal{F}_2$ replay check that never consults a checkpoint, and a matched two-action solver budget	Arrival at a common solvable frontier
Do solvers receive the same input?	Exact replay with equal environment observations, history, admissible actions, budget, prompt, and paired seeds	Same-state SOLVE effect
Are difficult non-arrivals removed?	Every natural non-arrival remains a zero endpoint outcome; reached-state results are reported separately	Deployed-agent endpoint interaction
Does scoring introduce the interaction?	Environment-native `won` in ALFWorld; archived Formatter replies and exact official-score recomputation in TravelPlanner	Stable benchmark outcomes
Is the interaction prediction fitted to the target split?	Conversion is measured on an independently selected sample before it is combined with target arrival	Out-of-sample interaction prediction

Table 6: **Per-task-type interaction prediction (Equation 10).** Arrival is from the target `valid_unseen` split and the SOLVE profile from the independent sample. The overall row is the unweighted mean over types, the stratified bootstrap unit; dashes mark cells it leaves undefined.

	Target-split arrival %		External SOLVE %		Interaction (pp)		
Task type	SFT	RL	SFT	RL	Pred.	Obs.	Resid.
look at obj in light	20.0	80.0	10.0	35.0	15.0	40.0	25.0
pick & place	35.0	70.0	85.0	100.0	5.2	−5.0	−10.2
clean then place	10.0	70.0	20.0	100.0	48.0	60.0	12.0
cool then place	0.0	100.0	80.0	100.0	20.0	25.0	5.0
heat then place	15.0	100.0	60.0	80.0	17.0	5.0	−12.0
pick two & place	0.0	75.0	60.0	85.0	18.8	20.0	1.2
Overall	13.3	82.5	—	—	20.7	24.2	3.5

had equal inputs and seed schedules, and no prompt exceeded 4,096 tokens. Table 5 maps each common attribution concern to the protocol control that defines the reported quantity.

C INDEPENDENT SOLVE PROFILE SAMPLE

The external SOLVE profile is selected before inspecting continuation outcomes on the target `valid_unseen` sample. It comes from a separate `valid_seen` experiment containing five tasks per task type. Four paired seeds yield 120 distance-two states per solver. We combine these six type-specific conversion rates with target-split type-specific arrival rates. This estimator requires transportability of continuation success within task type from the independent sample to the target split. It deliberately does not assume that task type is sufficient for every aspect of state difficulty.

The resulting interaction prediction is 20.7 percentage points, compared with an observed interaction of 24.2 points. The residual is 3.5 points with 95% CI $[-7.2, 14.2]$, quantifying the remaining difference between prediction and observation and the uncertainty from the finite task samples. Figure 8 plots both solver levels by task type rather than their difference: *pick & place* gains only 5 points because the SFT solver already completes 95.2% of those states, whereas *clean then place* gains 100 points from a floor of zero. Figure 9 shows the per-type residuals behind the 3.5-point aggregate: type-level errors are larger and have both signs. Table 6 lists the same residuals with the arrival and conversion rates they are computed from. Type-level residuals run from $+25.0$ on look at object in light to -12.0 on heat then place and cancel in the unweighted mean. Comparing the external conversion gap with the same-state SOLVE gap on the target's own replayed states (Figure 8) locates the error. For look at object in light the external profile gives a 25-point gap where the target states show 70, and the prediction falls 25 points short. For pick and place it gives 15 where the target shows 4.8, and it overshoots by 10.2. Five of the six residuals share the sign of this difference; heat then place is the exception, with gaps of 20 and 21.7 that agree. The error sits in the transport of the within-type SOLVE profile, which is the assumption the external test was designed to expose.

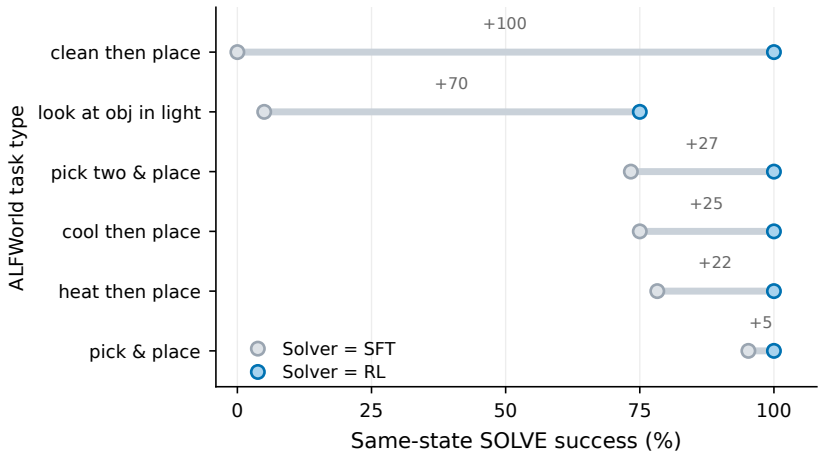

Figure 8: **Same-state SOLVE by task type on `valid_unseen`.** Rows are sorted by the RL-minus-SFT gap. The source data report no per-type intervals.

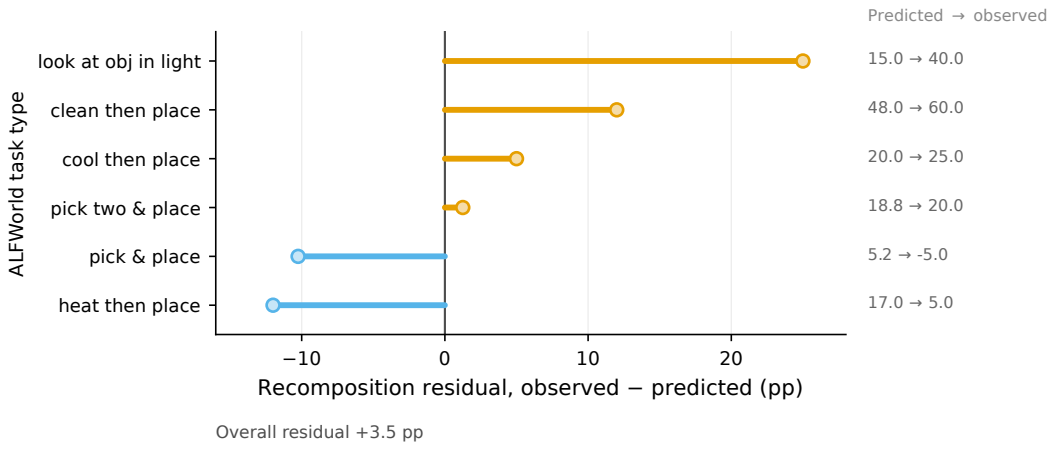

Figure 9: **Per-task-type interaction-prediction residual, observed minus predicted.** Positive means the external profile underpredicts. The source data report no per-type intervals.

D ROBUSTNESS CHECKS

Seen-split replication. On a separately selected, type-balanced 30-task `valid_seen` sample, RL improves frontier arrival by 65.8 points with 95% CI $[54.2, 77.5]$, improves same-state completion by 26.5 points with CI $[11.2, 41.3]$, and yields a positive endpoint interaction of 17.5 points with CI $[6.7, 27.5]$. All three effects replicate on a different split with positive lower confidence bounds.

SOLVE by solver action. Every frontier state admits bridge actions, first actions from which one further action wins, so SOLVE factors into choosing a bridge action and finishing from it (Table 3). On `valid_unseen` the SFT solver chooses a bridge action on 72.2% of states and finishes from one on 78.3% of those; the RL solver does so on 97.4% and 98.2%. The two stages contribute comparably to the SFT deficit, 25.2 and 19.9 points, and their products reproduce the completion rates of 56.5% and 95.7%. Of the 115 states, 45 are won by the RL solver only and none by the SFT solver only. On `valid_seen` the RL solver alone wins 41 states and the SFT solver alone wins 10. The SFT solver behaves almost identically on the two splits, 72.6% against 72.2% at the bridge stage and 49.6% against 56.5% overall. The change is on the RL side: its finishing rate given a bridge action falls from 98.2% to 82.4%, while its bridge rate falls only from 97.4% to 92.3%. The replication preserves the direction and the two-stage structure of the SOLVE effect; the RL advantage at the second action is the less stable of the two.

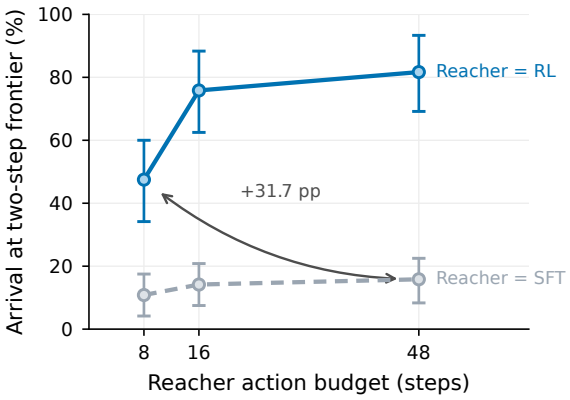

Figure 10: **Frontier arrival against reacher action budget on `valid_seen`.** Arrival at three budgets on the separately selected `valid_seen` sample, with task-stratified paired-bootstrap 95% intervals. The `valid_unseen` curve is Figure 2.

Table 7: **Frontier arrival by reacher budget on `valid_seen`.** The bolded row contrasts RL at 8 actions with SFT at 48. Brackets are task-stratified paired-bootstrap 95% intervals from 100,000 resamples.

(a) Arrival by reacher action budget, seen split

Budget (steps)	Reacher	Arrival %	95% CI
8	SFT	10.8	[4.2, 17.5]
8	RL	47.5	[34.2, 60.0]
16	SFT	14.2	[7.5, 20.8]
16	RL	75.8	[62.5, 88.3]
48	SFT	15.8	[8.3, 22.5]
48	RL	81.7	[69.2, 93.3]

(b) Contrasts

Comparison	Effect (pp)	95% CI
RL@8 − SFT@8	36.7	[23.3, 50.0]
RL@16 − SFT@16	61.7	[48.3, 74.2]
RL@48 − SFT@48	65.8	[54.2, 77.5]
RL@8 − SFT@48	**31.7**	[19.2, 44.2]

Early-action budget diagnostic. We reuse archived `valid_seen` trajectories at common early-action budgets 8, 16, and 48. Table 7 lists the arrival levels and their contrasts. This post-hoc diagnostic shows a positive RL arrival advantage at eight steps that remains large as both checkpoints receive more steps. In the direct cross-budget comparison, RL at eight actions exceeds SFT at 48 actions by 31.7 points with CI $[19.2, 44.2]$. Budget data exist for `valid_seen` only; no accepted `valid_unseen` budget artefact exists, so these intervals are not attached to the `valid_unseen` curve of Figure 2. The claim is bounded by the protocol: cumulative arrival within 48 reacher actions under frozen policies, with no extrapolation to additional sampling, search, or replanning.

Non-arrival failure modes. Every `valid_unseen` trajectory that never reaches $\mathcal{F}_2$ (104 SFT, 21 RL, all running the full 48 actions) is assigned one of seven mutually exclusive labels by an automatic audit of its action sequence. For SFT, 102 trajectories are loops that repeat actions without changing the state and 2 miss a subgoal; for RL, 18 are loops, 2 wander, and 1 never picks up the target object. No trajectory shows format collapse. At the step level, SFT repeats its immediately preceding action on 66.7% of steps (RL 43.4%), revisits an earlier state on 86.3% (75.2%), and leaves the state unchanged on 76.2% (47.2%); the median number of distinct actions per trajectory is 4 for SFT and 8 for RL. Trajectory-level bootstrap intervals from 100,000 resamples put the RL-minus-SFT difference in loop share at -12.4 points with CI $[-28.6, 1.0]$. The audit is descriptive and makes no mechanism claim.

Non-arrival retention diagnostic. We compare two estimands on the accepted 480-row `valid_unseen` matrix. The endpoint estimand retains non-arrivals as zero outcomes. The survivor-only estimand computes the paired RL-minus-SFT solver effect separately on the states reached by each reacher, then differences those two effects. Both use 100,000 task-stratified paired-bootstrap draws with all four seeds and checkpoint assignments kept together. Fifty-five draws have no SFT arrival and are excluded under the fixed empty-denominator rule, a rate of 0.055% against the 1% stopping threshold.

The endpoint interaction is 24.2 points with 95% CI $[13.3, 35.0]$. The survivor-only estimate is -12.6 points with CI $[-45.8, 22.7]$, and its paired difference from the endpoint interaction is -36.8 points with CI $[-68.6, -7.3]$. SFT reaches $\mathcal{F}_2$ in only 16 task and seed trajectories from six tasks and has no arrivals in two of the six task types; RL reaches it in 99 trajectories from 28 tasks. This support loss explains the wide survivor-only interval. The diagnostic was specified after its point estimates were observed, so it establishes neither a confirmatory effect nor a negative conditional interaction. It shows that deleting non-arrivals does not preserve the endpoint estimand in this setting.

E A Handoff State in Detail

Table 8 shows one of the 45 `valid_unseen` states that the RL solver completes and the SFT solver does not, selected by a prespecified rule (SFT-produced state, simplest task type, SFT first action not a bridge action). The task is "put a pencil in shelf". The SFT reacher produced the state by taking the pencil from the desk; the cloned observation reads "You pick up the pencil 1 from the desk 1", and the solvability check lists six bridge actions, walking to any of shelves 1 to 6, each followed by the single winning action of moving the pencil there. Both solvers receive the identical clone; the two handoff prompts hash to the same value. The SFT solver's reasoning names the right plan, "head straight to the nearest shelf", and then spends its first action on `look`; its second action walks to a shelf, one action too late. The RL solver walks to a shelf and places the pencil.

Table 8: **Two solvers on one cloned state.** Bridge actions in blue. The SFT solver's first-step reasoning is quoted from its archived reply.

Step	SFT solver	RL solver
1	`look` *"I should head straight to the nearest shelf. I need to locate a shelf first … I'll scan the room."*	`go to shelf 1`
2	`go to shelf 1`	`move pencil 1 to shelf 1`
Outcome	budget exhausted, not won	won