# THINK OR CALL? COUNTERFACTUAL NECESSITY OPTIMIZATION FOR SOFTWARE AGENTS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Software agents must repeatedly decide whether to invoke a tool or continue reasoning with the evidence already available. Terminal task rewards provide limited credit for individual invocations: successful trajectories may contain redundant calls, while failed trajectories may contain essential ones. We introduce Counterfactual Necessity Optimization (CNO), a method that learns tool routing by comparing the downstream value of these two choices. At selected decision states, CNO restores the same agent and repository state into two isolated branches, executes a proposed tool call in one and a reasoning step in the other, and completes both under a shared frozen policy. Their return difference, accounting for task completion, protocol compliance, and optional-call cost, supplies a soft necessity target. A necessity head learns these targets jointly with the policy and guides routing at deployment without counterfactual rollouts. An obligation mask enforces required interactions during training and inference. We establish conditions for unbiased paired estimation and show that uniform error $\epsilon$ in the optimal tool-versus-reason value gap yields routing regret of at most $H\epsilon$ over horizon $H$. On SWE-bench Verified, CNO improves resolution from 38.8% to 45.8% over outcome-only reinforcement learning while reducing optional calls per resolved issue by 39.1%. Further evaluation on SWE-Bench Pro shows consistent resolution gains. Routing diagnostics show fewer unnecessary invocations and skipped necessary calls, with protocol validity close to mask-only training. These results support invocation-level counterfactual credit for improving task completion and tool efficiency under explicit interaction constraints.

## 1 INTRODUCTION

Software agents spend their interaction budgets on reasoning, repository inspection, code edits, and tests. Each tool call changes the evidence available to later decisions. Running a test can expose a defect that reasoning over the current transcript cannot reveal; repeating a search can consume the remaining budget without improving the patch. Effective tool use therefore requires deciding whether a particular call is worth making at the current state.

Demonstrations and terminal outcome rewards teach agents to invoke tools and complete tasks (Yao et al., 2023; Schick et al., 2023; Qin et al., 2023; Feng et al., 2026; Li et al., 2025). However, terminal success gives limited credit to individual calls. A successful trajectory can contain redundant inspections, and an unsuccessful trajectory can contain essential ones. Call penalties supply a more direct efficiency signal, but assign the same cost to actions with different downstream value. They can also discourage required checks (Ma et al., 2026). Recent preference, process-reward, and omission methods address excessive tool use (Wang et al., 2025; Chen et al., 2026; Li et al., 2026a; Ning et al., 2026); the remaining challenge is to identify the value of a route decision inside a mutable environment.

We address this challenge with a within-trajectory intervention: starting from the same repository, transcript, obligations, and remaining budget, compare the agent's proposed tool call with one step of internal reasoning. Both branches then complete the episode under a shared suffix policy. Their return difference measures the value of the call through its consequences for later actions and task completion. This extends query-level search counterfactuals (Kim, 2026) to repeated decisions in stateful software tasks.



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.

Counterfactual Necessity Optimization (CNO) uses these paired returns to train a necessity head and a route policy. The head amortizes the cost of branching, so deployment requires only the policy and its routing score. A deterministic obligation mask restricts admissible actions throughout training and deployment. Required interactions, such as checking a patch after its last source mutation, are enforced by the protocol; optional calls receive learned tool-versus-reason credit.

Across training, CNO improves SWE-bench Verified resolution by 7.0% over outcome-only reinforcement learning (RL) and reduces optional calls per resolved issue by 39.1%. The method also reduces both unnecessary invocations and skipped necessary calls. Its advantage concerns the relationship between tool use and task completion: Tool-Light uses fewer optional calls per trajectory, and ParetoPO uses fewer tokens.

Our contribution is a paired estimator of tool necessity at repeated states of a software-agent trajectory, together with a training objective that combines necessity distillation, policy optimization, and obligation masking. A finite-horizon analysis connects value-gap accuracy to routing regret. The evaluation measures resolution, both directions of routing error, protocol validity, and calibration, with ablations that isolate the method's components.

## 2 Problem formulation

We model repository repair as a finite-horizon, protocol-constrained Markov decision process. The state is

$$s_t = (x_t, e_t, h_t, b_t, o_t), \tag{1}$$

where $x_t$ contains the transcript and agent memory, $e_t$ is the executable environment snapshot, $h_t$ is the remaining decision horizon, $b_t$ records token and tool budgets, and $o_t$ records pending protocol obligations. An action is an internal reasoning segment $\mathsf{reason}$, a structured tool call $u \in \mathcal{U}(s_t)$, or $\mathsf{finalize}$. A tool proposal specifies both its name and arguments. Reasoning updates the transcript without directly changing the executable environment. The transcript, obligations, and budgets retain information from earlier interactions that affects the next decision. Identical repository contents can therefore lead to different route choices when the available evidence, pending checks, or remaining budget differ.

**Admissible actions.** The environment supplies a predicate $m(s, a) \in \{0, 1\}$ for schema validity and protocol obligations. It excludes actions that violate a task rule or make an outstanding obligation impossible to discharge within the remaining horizon. The executed policy is

$$\pi_\theta^m(a \mid s) = \frac{m(s, a)\pi_\theta(a \mid s)}{\sum_{a'} m(s, a')\pi_\theta(a' \mid s)}. \tag{2}$$

If no admissible action remains, the episode terminates with a protocol failure. Section 3.3 specifies the repository-repair obligations.

**Task and interaction value.** For a trajectory $\tau$, the constrained return is

$$G(\tau) = R_{\text{task}}(\tau) + \lambda_p R_{\text{prot}}(\tau) - \lambda_c C_{\text{opt}}(\tau). \tag{3}$$

Here $R_{\text{task}}$ measures issue resolution, $R_{\text{prot}} \leq 0$ penalizes invalid interactions, and $C_{\text{opt}}$ charges optional calls. The coefficients $\lambda_p, \lambda_c \geq 0$ control the corresponding penalties. A call that discharges a recorded obligation is exempt from the optional-call charge. The obligation's triggering event and required tool class are recorded before that call occurs.

**Necessity of a concrete proposal.** Let $u^*(s)$ be the proposal policy's highest-probability admissible tool call. At a state admitting both that call and reasoning, define

$$Q^\pi(s, a) = \mathbb{E}[G(\tau) \mid s_t = s, a_t = a, \pi], \tag{4}$$

$$\Delta^\pi(s) = Q^\pi(s, u^*(s)) - Q^\pi(s, \mathsf{reason}), \tag{5}$$

where $\pi$ is the policy used after the intervention. We call the proposal *necessary* when $\Delta^\pi(s) > 0$ and *unnecessary* when $\Delta^\pi(s) < 0$; a gap near zero indicates indifference. This operational definition measures the value of the proposed call relative to one reasoning step. It depends on the proposal's arguments, the suffix policy, the available evidence, and the remaining budget.

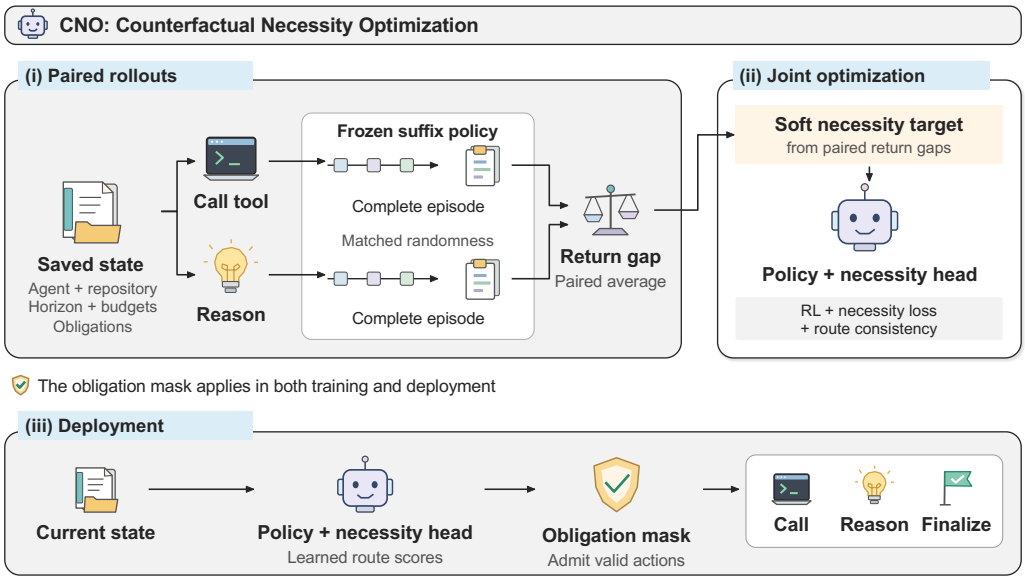



Figure 1: CNO training and deployment. (i) Two isolated branches start from the same saved state, execute different first actions, and complete under one frozen suffix policy. (ii) Their return difference produces a soft necessity target for joint training. (iii) The learned head biases routing at deployment; the obligation mask enforces admissibility. The mask is active in both stages.

## 3 Counterfactual Necessity Optimization

CNO collects paired tool-versus-reason returns during training, distills them into a state-dependent routing score, and optimizes the agent under the same protocol constraints (Figure 1). The intervention evaluates a concrete proposal, while the underlying policy remains responsible for tool selection and arguments.

### 3.1 Paired rollouts from an exact state

An on-policy rollout provides candidate branch states. We stratify them by episode phase and tool-route entropy and sample at most four per trajectory. Sampling includes confident choices as well as uncertain ones, allowing the head to learn from habitual overuse. Only states admitting both intervention actions contribute an optional-route target.

At each selected state, the runtime saves the transcript, memory, repository, process and test state, obligations, random-number state, and remaining budgets. Two isolated restorations produce the same pre-action state. One branch executes $u^*(s)$; the other generates a reasoning segment under the same token cap. Both then complete under a frozen suffix policy, sharing random seeds where the runtime supports them. Each branch starts with the same horizon and budgets; subsequent consumption depends on its actions.

With $K$ suffix samples per action, the target value gap is

$$\widehat{\Delta}(s) = \frac{1}{K} \sum_{k=1}^{K} \left[ G(\tau_k^u) - G(\tau_k^r) \right]. \tag{6}$$

The two trajectories can diverge after the intervention. For example, evidence from a test can change subsequent edits and the final outcome; these downstream effects contribute to the estimated value of the call. The comparison holds the pre-action state, suffix policy, and reward function fixed.

The runtime enforces three conditions. *State fidelity* requires equal pre-action files, process state, transcript, memory, obligations, horizon, and budgets. *Branch isolation* assigns separate writable

environments and episode-local service caches to the two branches. *Suffix consistency* fixes the policy and evaluator versions until both branches finish. Appendix C gives the sampling and runtime details.

**Lemma 1** (Paired estimation)**.** *Conditioned on a saved state $s$, suppose the $K$ return pairs in Equation 6 are independent and identically distributed across $k$, each action marginal follows the target suffix policy, and the returns have finite second moments. Then $\widehat{\Delta}(s)$ is unbiased for $\Delta^{\pi}(s)$ and*

$$\operatorname{Var}[\widehat{\Delta} \mid s] = \frac{1}{K}\left(\sigma_u^2 + \sigma_r^2 - 2\operatorname{Cov}[G^u, G^r \mid s]\right), \tag{7}$$

*where $\sigma_u^2$ and $\sigma_r^2$ are the conditional return variances. If the covariance is nonnegative, the pairing has no greater variance than independent suffix sampling.*

*Proof.* See Appendix A.1. □

Matched randomness can reduce variance when it couples the branch returns positively. Unbiasedness follows from correct action marginals and does not require shared seeds.

### 3.2 Necessity distillation and policy learning

A head $g_\theta(s) \in (0, 1)$ shares the policy representation and predicts the soft target

$$z(s) = \sigma\left(\widehat{\Delta}(s)/T\right), \tag{8}$$

where $\sigma$ is the logistic sigmoid and $T > 0$ controls the transition around indifference. A zero gap gives $z = 1/2$; positive and negative gaps move the target toward one and zero, respectively. Increasing $T$ moves targets closer to indifference for a fixed gap. Monotonicity in $\widehat{\Delta}$ gives the target its interpretation as a routing score.

The necessity loss is weighted binary cross entropy:

$$\mathcal{L}_{\mathrm{nec}} = -\mathbb{E}_s\, w(s)\big[z \log g_\theta + (1 - z)\log(1 - g_\theta)\big]. \tag{9}$$

The weight $w(s)$ combines inverse sampling probabilities for phase and entropy strata with a bounded confidence factor derived from paired-return uncertainty. The inverse-probability component accounts for stratified selection, while the confidence factor reduces the influence of imprecise near-zero gaps. Each paired target is used in one update batch under its originating policy version.

The policy receives a clipped RL objective on $G$ and a routing-consistency loss. Writing its admissible tool mass as $p_\theta^{\mathsf{tool}}(s) = \sum_{u \in \mathcal{U}(s)} \pi_\theta^m(u \mid s)$, the combined objective is

$$\mathcal{L} = \mathcal{L}_{\mathrm{RL}} + \lambda_n \mathcal{L}_{\mathrm{nec}} + \lambda_g \mathbb{E}_s \operatorname{BCE}(z, p_\theta^{\mathsf{tool}}). \tag{10}$$

The RL term optimizes the constrained trajectory return, while necessity supervision learns the local tool-versus-reason comparison. Route consistency connects this comparison to the policy's action probabilities, giving the paired target a direct role in both scoring and route selection. Gradients through $z$ are stopped. States where the protocol forces a tool action are excluded from the optional-route consistency term and train tool selection and argument validity. We summarize the update in Algorithm 1, Appendix B.

At deployment, $g_\theta$ biases the logits of the tool and reasoning action groups, while the policy selects the concrete call and arguments. A routing threshold is selected on development data to maximize task return subject to minimum protocol validity, then fixed for test evaluation. Deployment uses the learned score without counterfactual rollouts.

### 3.3 Protocol-obligation masking

The environment maintains a deterministic obligation automaton. Repository-repair rules require valid call schemas, inspection before a file's first mutation, a designated check after the final source mutation, and completion of pending obligations before finalization. A failed check also requires acknowledgement of the resulting evidence. We specify the transitions in Table 1.

The mask and the optional-call exemption encode the same obligations. This prevents a call-economy objective from rewarding omission of required interactions. A mask-only baseline isolates the contribution of enforcement, and the mask ablation measures its effect within CNO.

Table 1: Obligation transitions in the repository harness. The environment records the event that opens or discharges each obligation.

| Trigger | Obligation | Restriction |
|---|---|---|
| Edit proposal | Inspect the target file | Mutation requires prior inspection |
| Source mutation | Run the designated check | Finalization requires a subsequent check |
| Tool emission | Use a valid name and schema | Malformed calls are inadmissible |
| Check failure | Acknowledge test evidence | Finalization requires acknowledgement |

## 4 FINITE-HORIZON ROUTING ANALYSIS

We isolate the choice between a fixed tool proposal $u(s)$ and reason. Let $Q_h^*(s,a)$ be the optimal constrained action value with $h$ decisions remaining, within this binary action set and under the obligation mask. Define $\Delta_h^*(s) = Q_h^*(s,u) - Q_h^*(s,\text{reason})$ at states where both routes are admissible. States with only one admissible route use that route. Termination and finalization rules are shared by the compared policies.

**Theorem 1** (Routing regret)**.** *Suppose the finite-horizon values are well-defined and, for every reachable state with both routes admissible and at most $H$ decisions remaining, $|\widehat{\Delta}_h(s) - \Delta_h^*(s)| \le \epsilon$. Let $\widehat{\pi}$ choose $u$ exactly when $\widehat{\Delta}_h(s) \ge 0$, among actions admitted by the same mask as the optimal router. Then*

$$V_H^*(s) - V_H^{\widehat{\pi}}(s) \le H\epsilon \quad \textit{for every reachable } s. \tag{11}$$

*Proof.* See Appendix A.2. □

Under the theorem's assumption, states with $|\Delta_h^*(s)| > \epsilon$ retain the correct gap sign and select the optimal route. A mistaken choice is confined to states where the routes differ in optimal value by at most $\epsilon$. Summing these local losses over the remaining decisions yields the linear dependence on horizon. The bound concerns uniform approximation to the optimal gap $\Delta_h^*$; the paired estimator in Section 3.1 and the empirical gap-error measurements concern the current suffix policy's gap $\Delta^\pi$.

## 5 EXPERIMENTS

In this section, We evaluate task completion, tool-use efficiency, and routing quality on 500 SWE-bench Verified issues and 1,865 SWE-Bench Pro issues (Jimenez et al., 2024; Deng et al., 2026).

**Models.** We use Qwen3.5-35B-A3B (Qwen Team, 2026) as the common backbone for CNO and all baselines. The necessity head shares the policy representation, as described in Section 3.2.

**Training pipeline.** We train agents on repository-repair tasks from SWE-Gym (Pan et al., 2025). For CNO, on-policy trajectories provide candidate decision states, stratified by episode phase and tool-route entropy. At most four states per trajectory are restored into paired tool and reasoning branches, which complete under a frozen suffix policy. Their return differences supervise the necessity head alongside policy optimization. Algorithm 1 specifies the update, and Appendix C describes branch sampling and execution.

**Baselines.** We compare CNO with outcome-only RL, call-cost RL, preference routing, Tool-Light (Chen et al., 2026), ParetoPO (Li et al., 2026a), and a tool-call reward model (TRM; Ma et al., 2026). These methods cover outcome supervision, call penalties, efficiency preferences, joint accuracy–cost optimization, and invocation-level rewards. We also include mask-only RL and an unpaired value head to isolate the contributions of obligation enforcement and paired necessity supervision.

Table 2: Repository-repair point estimates. Resolution and protocol validity are percentages. Calls/res. is total optional calls divided by resolved issues; optional calls and tokens are trajectory means. Bold denotes the best value in each column.

| | Resolved (%) ↑ | | | | | |
|---|---|---|---|---|---|---|
| Method | Verified | Pro | Calls/res. ↓ | Opt. calls ↓ | Tokens ↓ | Protocol ↑ |
| Outcome RL | 38.8 | 17.5 | 37.2 | 14.43 | 37,810 | 95.2 |
| Call-cost RL | 36.6 | 15.8 | 26.8 | 9.79 | 32,516 | 94.4 |
| Preference routing | 40.8 | 18.4 | 29.1 | 11.88 | 34,407 | 96.2 |
| Tool-Light | 33.6 | 14.5 | 24.1 | **8.11** | 28,357 | 93.6 |
| ParetoPO | 41.6 | 19.1 | 26.4 | 11.00 | **27,223** | 96.4 |
| TRM | 42.4 | 19.6 | 27.1 | 11.47 | 33,273 | 96.6 |
| Mask-only RL | 39.8 | 17.8 | 35.6 | 14.19 | 38,188 | **99.4** |
| Unpaired value head | 41.4 | 19.1 | 29.3 | 12.14 | 34,029 | 96.8 |
| **CNO** | **45.8** | **22.3** | **22.7** | 10.38 | 31,004 | 99.2 |

**Metrics.** We report issue resolution, mean optional calls and tokens per trajectory, and optional calls per resolved issue. The last metric divides total optional calls, including those from failed trajectories, by the number of resolved issues. Unnecessary-call and skipped-necessary-call rates measure routing errors at negative- and positive-gap states, respectively. Protocol validity and mandated-call retention assess obligation adherence; necessity expected calibration error (ECE) and value-gap mean absolute error (MAE) assess the routing signal. Appendix D provides statistical contrasts and horizon-specific diagnostics.

**Implementation details.** All methods share the tool schema, context limit, decision horizon, decoding schedule, and training-instance budget. The routing threshold is 0.50, selected on development data and fixed for evaluation. Temperature scaling and equal-mass calibration bins are also fitted on development predictions. Reported confidence intervals use 5,000 repository-stratified issue-cluster bootstrap samples. Appendix D gives the testing protocol and noninferiority margins.

### 5.1 Task completion and tool use

CNO achieves the highest resolution in both benchmarks (Table 2). On Verified, it resolves 45.8% of issues compared with 38.8% for outcome-only RL, a gain of 7.0 percentage points (95% CI [5.4, 8.7], $p = 2 \times 10^{-5}$). The gain is positive for each training seed. On Pro, resolution increases from 17.5% to 22.3%. Compared with TRM, the strongest baseline on resolution, CNO improves Verified and Pro by 3.4 and 2.7 points, respectively.

Optional calls per resolved issue decrease from 37.19 to 22.66, a 39.1% reduction. The reported difference is $-14.54$ calls (95% CI $[-17.10, -12.45]$). This improvement combines higher resolution with fewer optional calls per trajectory: 10.38 for CNO versus 14.43 for outcome-only RL.

The resource tradeoffs remain visible in Figure 2(a). Tool-Light uses the fewest optional calls, 8.11 per trajectory, with 33.6% Verified resolution. Call-cost RL also uses fewer calls than CNO (9.79) and resolves 36.6%. ParetoPO uses the fewest tokens, 27,223 compared with 31,004 for CNO, and resolves 41.6%. These comparisons favor CNO when task completion and call efficiency are considered jointly, while identifying lower-resource alternatives.

### 5.2 Routing errors, protocol validity, and calibration

**Both types of routing error decrease.** Among decisive negative-gap states, unnecessary calls fall from 31.81% for outcome-only RL to 7.02% for CNO. The difference is $-24.79$ points (95% CI $[-28.29, -21.25]$). Skipped necessary calls also fall, from 10.17% to 6.78%; their difference is $-3.40$ points (95% CI $[-6.82, 0.00]$). The interval's upper endpoint is below the $+1.0$-point noninferiority margin.

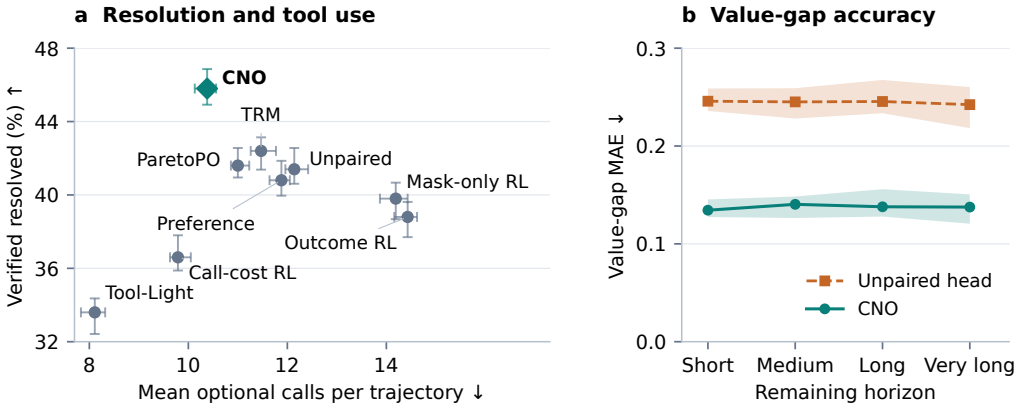



Figure 2: Task efficiency and routing accuracy. (a) Optional calls per trajectory versus Verified resolution for all methods in Table 2. (b) Value-gap MAE by remaining-horizon stratum. CNO has lower error than the unpaired head in all four strata.

Table 3: Mechanism ablation point estimates on the Verified ablation suite. Resolution and protocol validity are percentages. Gap MAE and protocol validity are evaluated within this suite, separately from the main comparison.

| Variant | Resolved ↑ | Gap MAE ↓ | Protocol ↑ |
|---|---|---|---|
| Full CNO | 45.8 | 0.085 | 99.7 |
| Unpaired target | 41.4 | 0.183 | 98.9 |
| Without exact restoration | 43.2 | 0.149 | 99.1 |
| Without matched suffix | 44.2 | 0.131 | 99.3 |
| Without full completion | 42.8 | 0.144 | 98.6 |
| Without necessity loss | 43.8 | 0.170 | 99.1 |
| Without route consistency | 44.4 | 0.119 | 99.2 |
| Without obligation mask | 43.4 | 0.100 | 96.5 |

**Protocol validity remains close to mask-only training.** CNO reaches 99.2% protocol validity versus 99.4% for mask-only RL. The difference is $-0.20$ points (95% CI $[-0.73, 0.33]$), satisfying the $-1.0$-point margin. Mandated-call retention is 99.58% and 99.69%, respectively; the reported difference is $-0.10$ points (95% CI $[-0.42, 0.21]$). The lower endpoint also satisfies the noninferiority criterion.

**Paired supervision improves the routing signal.** On held-out branch states, ECE falls from 0.0949 for the unpaired value head to 0.0462 for CNO, with a difference of $-0.0487$ (95% CI $[-0.0563, -0.0406]$). Gap MAE falls from 0.2446 to 0.1383, with a difference of $-0.1063$ (95% CI $[-0.1119, -0.1007]$). Both Holm-adjusted $p$-values are 0.0008. Figure 2(b) shows that the gap-error advantage persists across all four remaining-horizon strata.

### 5.3 Mechanism ablations and failure analysis

Table 3 isolates the components of CNO. Replacing paired credit with an unpaired target reduces resolution by 4.4 points. Removing exact restoration or matched suffix seeds increases gap error and reduces resolution by 2.6 and 1.6 points. Replacing full suffix completion with a one-step target costs 3.0 points. Removing the necessity loss or route consistency costs 2.0 and 1.4 points, respectively. These results associate the gains with both the quality of counterfactual targets and their use in policy training.

The mask ablation reduces protocol validity from 99.7% to 96.5% and resolution from 45.8% to 43.4% within the ablation suite. The mask therefore contributes to valid completion as well as invocation control. Its effect is distinct from the comparison with mask-only RL, which measures the value of learned routing given enforcement.

Failed tests account for 30.9% of CNO trajectories on Verified, compared with 35.9% for outcome-only RL. Context exhaustion is higher for CNO: 10.6% versus 9.3%. The latter result identifies a remaining budget-allocation weakness despite the reduction in optional calls. Agent timeouts and invalid actions count as failures; infrastructure faults are rerun under the same seed.

## 6 Related work

**Learning tool use.** ReAct interleaves reasoning and actions, Toolformer learns API insertion, and ToolLLM broadens tool instruction following (Yao et al., 2023; Schick et al., 2023; Qin et al., 2023). ReTool, ToRL, ToolRL, and Nemotron-Research-Tool-N1 study tool-integrated RL and reward design (Feng et al., 2026; Li et al., 2025; Qian et al., 2025a; Zhang et al., 2026b); AgentFlow optimizes interacting agent components (Li et al., 2026b). Process supervision assigns more local credit: TRM scores calls and estimates turn-level advantages, while DeRL-SWE separates localization from repair (Ma et al., 2026; Zhang et al., 2026a). CNO assigns credit to a specific tool-versus-reason intervention and includes its consequences over the remaining episode.

**Necessity and efficient invocation.** When2Call and SMART evaluate whether external assistance is warranted (Ross et al., 2025; Qian et al., 2025b). OTC-PO rewards correct answers with fewer calls, Tool-Light learns efficiency preferences, ParetoPO optimizes accuracy and tool use jointly, and Agent-Omit suppresses redundant observations or thoughts (Wang et al., 2025; Chen et al., 2026; Li et al., 2026a; Ning et al., 2026). Analyses of tool overuse distinguish encoded knowledge about necessity from executed routing behavior (Zeng et al., 2026; Sun et al., 2026; Cheng et al., 2026); a decision framework separates necessity, utility, and affordability (Wu et al., 2026). The closest counterfactual approach compares forced-search and no-search outcomes for the same query (Kim, 2026). CNO makes repeated interventions within an episode and restores the mutable environment, protocol state, and remaining horizon for each pair.

**Software-agent environments.** SWE-bench and SWE-agent establish repository repair and its interaction interface (Jimenez et al., 2024; Yang et al., 2024). SWE-Gym, SWE-RL, and long-context multi-turn RL provide training environments and recipes (Pan et al., 2025; Wei et al., 2025; Golubev et al., 2025). SWE-Bench Pro extends evaluation to longer software tasks, and repository-navigation RL studies evidence gathering (Deng et al., 2026; Zhang et al., 2026c). Sensitivity to prompts, history, and seeds motivates controlled comparisons (Liu et al., 2026); trajectory-aware evaluation also measures action order and dependencies (He et al., 2026; Patil et al., 2025). These concerns inform the shared-state intervention and separate reporting of task success and protocol validity.

## 7 Conclusion

CNO learns tool routing from the downstream return difference between a proposed call and one reasoning step at the same software-agent state. Paired supervision and obligation masking improve resolution while reducing optional calls per resolved issue and retaining protocol validity. We further identify value-gap accuracy as a sufficient condition for low routing regret, and the experiments distinguish improved invocation decisions from reductions in raw call count.

**Limitations.** Our evaluation focuses coding agent on repository repair, with state-restoration interfaces and obligation rules tailored to this setting. Applying CNO to other tool-use domains requires adapting these environment-specific components; its empirical benefits in those settings remain to be evaluated.

## AI DISCLOSURE

In this work, we used generative AI tools for polish writing. We have not used generative AI tools for other tasks. We have reviewed all AI-assisted work, and we take responsibility for the final content of this work.

## REPRODUCIBILITY STATEMENT

Section 3 and Algorithm 1 specify the estimator and training objective. Appendix A gives complete proofs, Appendix C describes branch sampling and runtime conditions, and Appendix D records the statistical contrasts.

## REFERENCES

Yifei Chen, Guanting Dong, and Zhicheng Dou. Toward effective tool-integrated reasoning via self-evolved preference learning. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=mNeitRAdWV.

Yize Cheng, Chenrui Fan, Mahdi JafariRaviz, Keivan Rezaei, and Soheil Feizi. Model-adaptive tool necessity reveals the knowing-doing gap in LLM tool use. *arXiv preprint arXiv:2605.14038*, 2026. URL https://arxiv.org/abs/2605.14038.

Xiang Deng, Jeff Da, Edwin Pan, Yannis Yiming He, Charles Ide, Kanak Garg, Niklas Lauffer, Andrew Park, Nitin Pasari, Chetan Rane, Karmini Sampath, Maya Krishnan, Srivatsa Kundurthy, Sean Hendryx, Zifan Wang, Vijay Bharadwaj, Jeff Holm, Raja Aluri, Chen Bo Calvin Zhang, Noah Jacobson, Bing Liu, and Brad Kenstler. SWE-Bench Pro: Can AI agents solve long-horizon software engineering tasks? In *International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=uEVTdoAbnK.

Jiazhan Feng, Shijue Huang, Xingwei Qu, Ge Zhang, Yujia Qin, Baoquan Zhong, Chengquan Jiang, Jinxin Chi, and Wanjun Zhong. ReTool: Reinforcement learning for strategic tool use in LLMs. In *International Conference on Learning Representations*, 2026. URL https://iclr.cc/virtual/2026/poster/10006978.

Alexander Golubev, Maria Trofimova, Sergei Polezhaev, Ibragim Badertdinov, Maksim Nekrashevich, Anton Shevtsov, Simon Karasik, Sergey Abramov, Andrei Andriushchenko, Filipp Fisin, Sergei Skvortsov, and Boris Yangel. Training long-context, multi-turn software engineering agents with reinforcement learning. *arXiv preprint arXiv:2508.03501*, 2025. URL https://arxiv.org/abs/2508.03501.

Pengfei He, Zhenwei Dai, Bing He, Hui Liu, Xianfeng Tang, Hanqing Lu, Juanhui Li, Jiayuan Ding, Subhabrata Mukherjee, Suhang Wang, Yue Xing, Jiliang Tang, and Benoit Dumoulin. TRAJECT-Bench: A trajectory-aware benchmark for evaluating agentic tool use. In *International Conference on Learning Representations*, 2026. URL https://iclr.cc/virtual/2026/poster/10009304.

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? In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=VTF8yNQM66.

Minho Kim. When should LLMs search? counterfactual supervision for search routing. *arXiv preprint arXiv:2607.05752*, 2026. URL https://arxiv.org/abs/2607.05752.

Junyi Li, Xiaowei Qian, Yingyi Zhang, Wenlin Zhang, Guojing Li, Sheng Zhang, Xiao Han, Yichao Wang, and Xiangyu Zhao. Towards pareto-optimal tool-integrated agents with pareto ranking policy optimization. In *International Conference on Machine Learning*, 2026a. URL https://openreview.net/forum?id=9sg34nj13o.

Xuefeng Li, Haoyang Zou, and Pengfei Liu. ToRL: Scaling tool-integrated reinforcement learning. *arXiv preprint arXiv:2503.23383*, 2025. URL https://arxiv.org/abs/2503.23383.

Zhuofeng Li, Haoxiang Zhang, Seungju Han, Sheng Liu, Jianwen Xie, Yu Zhang, Yejin Choi, James Zou, and Pan Lu. In-the-flow agentic system optimization for effective planning and tool use. In *International Conference on Learning Representations*, 2026b. URL https://arxiv.org/abs/2510.05592.

Tong Liu, Cheng Qian, Matej Cief, Yuan He, Daniele Dan, Nikolaos Aletras, and Gabriella Kazai. On effectiveness and efficiency of agentic tool-calling and reinforcement learning training. In *International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=3oMO1ZQNwT.

Da Ma, Ziyue Yang, Hongshen Xu, Haotian Fang, Kai Yu, and Lu Chen. Empowering LLM tool invocation with tool-call reward model. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=LnBEASInVr.

Yansong Ning, Jun Fang, Naiqiang Tan, and Hao Liu. Agent-omit: Adaptive context omission for efficient LLM agents. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.04284.

Jiayi Pan, Xingyao Wang, Graham Neubig, Navdeep Jaitly, Heng Ji, Alane Suhr, and Yizhe Zhang. Training software engineering agents and verifiers with SWE-Gym. In *International Conference on Machine Learning*, 2025. URL https://arxiv.org/abs/2412.21139.

Shishir G. Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The berkeley function calling leaderboard (BFCL): From tool use to agentic evaluation of large language models. In *International Conference on Machine Learning*, 2025. URL https://proceedings.mlr.press/v267/patil25a.html.

Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. ToolRL: Reward is all tool learning needs. *arXiv preprint arXiv:2504.13958*, 2025a. URL https://arxiv.org/abs/2504.13958.

Cheng Qian, Emre Can Acikgoz, Hongru Wang, Xiusi Chen, Avirup Sil, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. SMART: Self-aware agent for tool overuse mitigation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 4604–4621, 2025b. doi: 10.18653/v1/2025.findings-acl.239.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. *arXiv preprint arXiv:2307.16789*, 2023. URL https://arxiv.org/abs/2307.16789.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Hayley Ross, Ameya Sunil Mahabaleshwarkar, and Yoshi Suhara. When2Call: When (not) to call tools. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3391–3409, 2025. doi: 10.18653/v1/2025.naacl-long.174.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2302.04761.

Chung-En Sun, Linbo Liu, Ge Yan, Zimo Wang, and Tsui-Wei Weng. LLM agents already know when to call tools, even without reasoning. *arXiv preprint arXiv:2605.09252*, 2026. URL https://arxiv.org/abs/2605.09252.

Hongru Wang, Cheng Qian, Wanjun Zhong, Xiusi Chen, Jiahao Qiu, Shijue Huang, Bowen Jin, Mengdi Wang, Kam-Fai Wong, and Heng Ji. Acting less is reasoning more! teaching model to act efficiently. *arXiv preprint arXiv:2504.14870*, 2025. URL https://arxiv.org/abs/2504.14870.

Yuxiang Wei, Olivier Duchenne, Jade Copet, Quentin Carbonneaux, Lingming Zhang, Daniel Fried, Gabriel Synnaeve, Rishabh Singh, and Sida I. Wang. SWE-RL: Advancing LLM reasoning via reinforcement learning on open software evolution. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2502.18449.

Qinyuan Wu, Soumi Das, Mahsa Amani, Arijit Nag, Seungeon Lee, Krishna P. Gummadi, Abhilasha Ravichander, and Muhammad Bilal Zafar. To call or not to call: A framework to assess and optimize LLM tool calling. *arXiv preprint arXiv:2605.00737*, 2026. URL https://arxiv.org/abs/2605.00737.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2405.15793.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Yirong Zeng, Shen You, Yufei Liu, Qunyao Du, Xiao Ding, Yutai Hou, Yuxian Wang, Wu Ning, Haonan Song, Dandan Tu, Bibo Cai, and Ting Liu. The tool-overuse illusion: Why does LLM prefer external tools over internal knowledge? *arXiv preprint arXiv:2604.19749*, 2026. URL https://arxiv.org/abs/2604.19749.

Fuxiang Zhang, Xinyu Guan, Jujie He, Liang Zeng, Jiacheng Xu, Yang Liu, and Bo An. DeRL-SWE: Decoupled reinforcement learning with nested credit assignment for software engineering. In *ICML Workshop on Deep Learning for Code*, 2026a. URL https://openreview.net/forum?id=MisdGC6Jlh.

Shaokun Zhang, Yi Dong, Jieyu Zhang, Jan Kautz, Bryan Catanzaro, Andrew Tao, Qingyun Wu, Zhiding Yu, and Guilin Liu. Nemotron-Research-Tool-N1: Exploring tool-using language models with reinforced reasoning. In *International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=yiE16lWzDj.

Zhaoxi Zhang, Yitong Duan, Yanzhi Zhang, Yiming Xu, Zhixiang Wang, Kun Liang, Weikang Li, Jiahui Liang, Deguo Xia, Jizhou Huang, Jiyan He, Shuxin Zheng, and Yunfang Wu. One tool is enough: Reinforcement learning of LLM agents for repository-level code navigation. In *International Conference on Machine Learning*, 2026c. URL https://openreview.net/forum?id=WTziQZdpTV.

# A PROOFS

## A.1 PROOF OF LEMMA 1

Condition on the restored state $s$. Correct action marginals imply

$$\mathbb{E}[G_k^u \mid s] = Q^\pi(s, u^*(s)), \qquad \mathbb{E}[G_k^r \mid s] = Q^\pi(s, \mathsf{reason}) \tag{12}$$

for each $k$. Linearity of conditional expectation then gives

$$\mathbb{E}[\widehat{\Delta}(s) \mid s] = \frac{1}{K}\sum_{k=1}^{K}(\mathbb{E}[G_k^u \mid s] - \mathbb{E}[G_k^r \mid s]) \tag{13}$$

$$= Q^\pi(s, u^*(s)) - Q^\pi(s, \mathsf{reason}) = \Delta^\pi(s). \tag{14}$$

Within a pair, the two returns may be dependent. Their difference has variance

$$\mathrm{Var}[G_k^u - G_k^r \mid s] = \sigma_u^2 + \sigma_r^2 - 2\,\mathrm{Cov}[G^u, G^r \mid s]. \tag{15}$$

Independence across the $K$ pairs removes cross-pair covariance terms, so

$$\mathrm{Var}[\widehat{\Delta} \mid s] = \frac{1}{K^2}\sum_{k=1}^{K}\mathrm{Var}[G_k^u - G_k^r \mid s] = \frac{\sigma_u^2 + \sigma_r^2 - 2\,\mathrm{Cov}[G^u, G^r \mid s]}{K}. \tag{16}$$

Independent suffix sampling with the same action marginals has zero within-pair covariance and variance $(\sigma_u^2 + \sigma_r^2)/K$. The claimed comparison follows whenever the paired covariance is nonnegative. □

## A.2 PROOF OF THEOREM 1

We use induction on the remaining horizon $h$. At $h = 0$, the continuation values coincide and the regret is zero. Terminal states also have zero continuation regret because their terminal outcome and finalization rules are shared. Suppose the bound holds at all reachable states with $h - 1$ remaining decisions.

At a nonterminal state $s$, let $a^*$ maximize $Q_h^*(s, a)$ over admissible routes and let $\hat{a}$ be the route chosen by $\widehat{\pi}$. If there is only one admissible route, or both policies choose the same route, then $Q_h^*(s, a^*) - Q_h^*(s, \hat{a}) = 0$. Otherwise the estimated-gap rule selects the wrong sign of the optimal gap. If $\hat{a} = u$ and $a^* = \mathsf{reason}$, then $\widehat{\Delta}_h(s) \geq 0$ and $\Delta_h^*(s) \leq 0$, giving

$$Q_h^*(s, a^*) - Q_h^*(s, \hat{a}) = -\Delta_h^*(s) \leq \widehat{\Delta}_h(s) - \Delta_h^*(s) \leq \epsilon. \tag{17}$$

If $\hat{a} = \mathsf{reason}$ and $a^* = u$, then $\widehat{\Delta}_h(s) < 0$ and $\Delta_h^*(s) \geq 0$, so

$$Q_h^*(s, a^*) - Q_h^*(s, \hat{a}) = \Delta_h^*(s) \leq \Delta_h^*(s) - \widehat{\Delta}_h(s) \leq \epsilon. \tag{18}$$

A true tie has zero action-value loss under either choice. Thus the action-value loss is at most $\epsilon$ in every case.

Let $s'$ denote the next state after taking $\hat{a}$. Adding and subtracting $Q_h^*(s, \hat{a})$ and applying the Bellman equations gives

$$V_h^*(s) - V_h^{\widehat{\pi}}(s) = Q_h^*(s, a^*) - Q_h^*(s, \hat{a}) + Q_h^*(s, \hat{a}) - Q_h^{\widehat{\pi}}(s, \hat{a}) \tag{19}$$

$$= Q_h^*(s, a^*) - Q_h^*(s, \hat{a}) + \mathbb{E}\Big[V_{h-1}^*(s') - V_{h-1}^{\widehat{\pi}}(s') \mid s, \hat{a}\Big] \tag{20}$$

$$\leq \epsilon + (h-1)\epsilon = h\epsilon. \tag{21}$$

The immediate reward cancels because both continuations start with the same state and action. The induction hypothesis applies to each reachable successor; early termination contributes zero continuation regret. Setting $h = H$ proves the theorem. □

# B TRAINING ALGORITHM

**Algorithm 1** One CNO training iteration

```
Roll out π_θ^m and collect candidate states S
Stratify eligible states by episode phase and tool-route entropy
Sample at most four states per trajectory; freeze the suffix policy
for each sampled state s do
    Save the prefix, environment, obligations, horizon, and budgets
    for k = 1, ..., K do
        Restore s in an isolated branch; run u*(s) and suffix with randomness ξ_k
        Restore s in an isolated branch; run reason and suffix with matched ξ_k
    end for
    Compute Δ̂(s) and z(s) using Equations 6 and 8
end for
Update the policy and necessity head with Equation 10
```

## C BRANCH SAMPLING AND RUNTIME

**Sampling.** Environment events divide episodes into discovery, localization, mutation, verification, and finalization phases. Within each phase, states are binned by tool-route entropy. The inverse probability of selecting a state's phase and entropy stratum enters its necessity-loss weight. Retaining confident states permits supervision of consistently unnecessary calls; limiting the number of branch states prevents long episodes from dominating the sampling budget.

**Uncertainty and policy versions.** The training record retains the paired returns and their standard error. A bounded confidence factor reduces the weight of targets whose uncertainty spans a broad interval around zero. Additional suffix pairs can refine uncertain targets while the originating policy version remains fixed. The branch suffix, soft target, and action log probabilities belong to that same version, and the targets are consumed in one policy-update batch.

**State restoration and isolation.** Both actions begin from the same transcript, memory, repository contents, process state, obligations, and remaining horizon and budgets. Branches use separate working trees, process groups, and episode-local caches. The runner retains tool outputs, exit codes, protocol events, and post-action diffs. Pre-action state equality, equal starting budgets, fixed evaluator and policy versions, and isolated writable environments are conditions for accepting a pair.

**Nondeterminism.** Each branch pair uses common random numbers where supported. Calls to external services require responses that can be replayed within the episode-local environment. For other nondeterministic tool behavior, valid action marginals preserve the interpretation of the estimator, although variance reduction is conditional on the resulting covariance. The randomness mode is retained with the pair.

**Compute accounting.** Counterfactual supervision adds up to $2K$ suffix completions per selected state. Its cost depends on the remaining horizon and tool workload at those states. Training tokens and environment steps are complementary measures because test execution can be expensive even when its transcript is short. Comparisons at equal total environment steps must include the counterfactual branches. The deployed policy evaluates the shared representation and necessity head without these branches.

## D STATISTICAL CONTRASTS AND ADDITIONAL DIAGNOSTICS

Resolution has a paired cluster-randomization $p$-value of $2 \times 10^{-5}$. The two calibration contrasts each have an adjusted $p$-value of 0.0008. For validity and retention, noninferiority requires the lower confidence bound to exceed $-1.0$ percentage point. For skipped necessary calls, it requires the upper bound to remain below $+1.0$ point. The intervals in Table 4 satisfy these directional criteria.

Table 4: Reported contrasts for CNO. Resolution and routing/protocol differences are in percentage points; the other rows retain their native units. Intervals are 95% confidence intervals. The two calibration tests use Holm-adjusted $p$-values.

| Metric | Comparator | Difference | 95% CI |
|---|---|---|---|
| Verified resolution | Outcome RL | $+7.0$ | $[5.4, 8.7]$ |
| Calls per resolved issue | Outcome RL | $-14.54$ | $[-17.10, -12.45]$ |
| Unnecessary-call rate | Outcome RL | $-24.79$ | $[-28.29, -21.25]$ |
| Skipped-necessary-call rate | Outcome RL | $-3.40$ | $[-6.82, 0.00]$ |
| Protocol validity | Mask-only RL | $-0.20$ | $[-0.73, 0.33]$ |
| Mandated-call retention | Mask-only RL | $-0.10$ | $[-0.42, 0.21]$ |
| Necessity ECE | Unpaired head | $-0.0487$ | $[-0.0563, -0.0406]$ |
| Value-gap MAE | Unpaired head | $-0.1063$ | $[-0.1119, -0.1007]$ |