# WHEN SHOULD AN AGENT COMMIT? PREFERENCE-CONDITIONED WORKFLOW CONTROL

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

## ABSTRACT

Stateful LLM agents repeatedly face a choice between replanning and committing to a multi-step plan. Frequent replanning preserves responsiveness to new observations but incurs repeated reasoning cost, whereas longer commitments amortize planning overhead while delaying high-level plan revision. We formulate this trade-off as *preference-conditioned adaptive commitment* and introduce PACE, a high-level controller that selects the next commitment horizon from the current workflow state and a quality–cost preference while keeping the executor frozen. PACE restores intermediate execution states to compare alternative horizons under a shared frozen continuation, yielding matched downstream quality–cost profiles. Horizon-specific predictors model relative quality retention and computational saving; the requested preference combines these predictions into a commitment policy, while a trainable residual is refined through on-policy interaction. We further analyze the induced semi-Markov control problem and use a finite controlled study to characterize when feedback-dependent horizon selection has value. The benchmark protocol evaluates the learned controller on ALFWorld and WebShop against fixed, query-level, and learned commitment strategies.

## 1 INTRODUCTION

Large language model (LLM) agents increasingly solve complex tasks through multi-step workflows that interleave reasoning, tool use, execution, and environmental feedback (Yao et al., 2023; Qin et al., 2024; Zhou et al., 2024). A canonical example is ReAct, which repeatedly alternates reasoning and acting so that new observations can continuously revise the agent's plan (Yao et al., 2023). While such reactive execution is flexible, repeatedly invoking reasoning and planning can incur substantial inference cost and latency (Kim et al., 2024). At the other extreme, an agent may generate a longer plan and execute several actions before replanning, amortizing planning overhead at the cost of reduced responsiveness to intermediate feedback.

A growing literature optimizes LLM pipelines, agent designs, and execution workflows (Khattab et al., 2024; Hu et al., 2025; Shang et al., 2025). AFlow searches code-represented workflows (Zhang et al., 2025b), MaAS learns query-dependent multi-agent architectures (Zhang et al., 2025a), and ARC configures workflows, tools, prompts, and compute budgets (Taparia et al., 2026). Flow and DyFlow adapt workflows from intermediate feedback (Niu et al., 2025; Wang et al., 2025), while FlowSteer learns state-conditioned workflow edits (Zhang et al., 2026). ParetoPO further studies accuracy–efficiency trade-offs for tool-integrated agents (Li et al., 2026). In contrast, we focus on *temporal commitment*: given the current execution history, when should the agent replan rather than continue a Plan-$k$ segment?

We view planning as a commitment decision that balances opportunities to revise the current plan against repeated reasoning cost (Figure 1). Longer commitments are attractive when the execution history supports a stable plan, while frequent replanning is valuable when new observations may alter it. This balance also depends on deployment preference: quality-oriented agents may prioritize responsiveness, whereas cost-oriented agents may accept longer commitments to save computation. We therefore study *preference-conditioned adaptive commitment*, where a single controller dynamically chooses between ReAct-style and Plan-$k$ execution according to the current history and the desired quality–cost trade-off.



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.

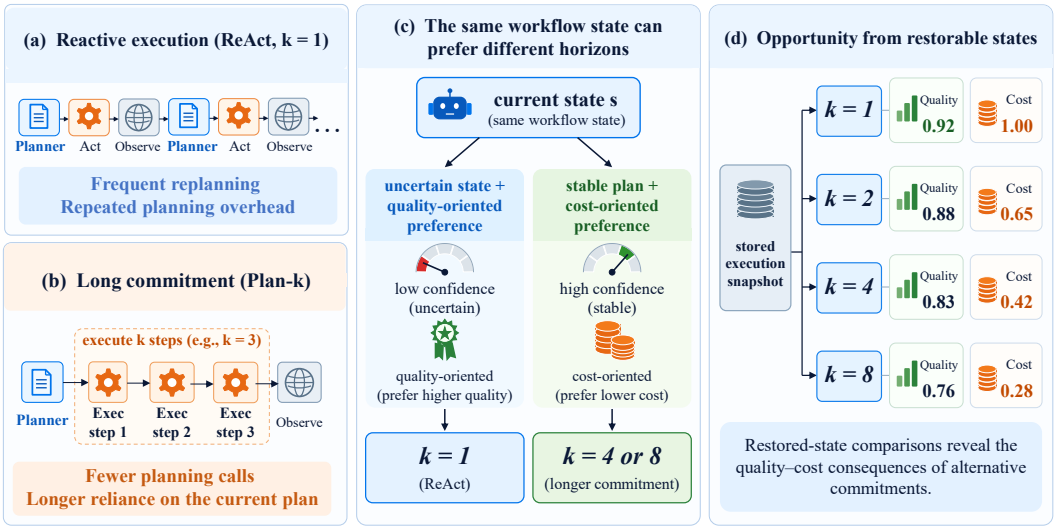



Figure 1: Adaptive execution commitment. (a,b) ReAct and Plan-$k$ differ in replanning frequency. (c) State and preference inform horizon selection. (d) Restored states support alternative-future comparisons.

We exploit a useful property of stateful workflows to learn this behavior: intermediate execution states can be restored. Starting from the same workflow history, we replay alternative commitment horizons and directly compare their downstream task quality and computational cost. These matched counterfactuals provide supervision for learning when additional feedback is worth waiting for and when the agent can safely commit further. The resulting controller can alternate between reactive and longer-horizon execution within the same trajectory, yielding different quality–cost behaviors under different deployment preferences. We call the resulting framework PACE. Our contributions are threefold: (i) we formulate *adaptive execution commitment* as a multi-objective workflow-control problem that unifies ReAct and Plan-$k$ within a single sequential policy; (ii) we introduce state-restored counterfactual evaluation to estimate the quality–cost consequences of alternative commitment horizons from identical workflow histories; and (iii) we develop a preference-conditioned commitment controller that combines horizon-specific quality retention and cost saving and refines commitment decisions through on-policy workflow optimization.

## 2 Related Work

**Agent workflow optimization and learned orchestration.** Automated workflow design has moved from search over static structures toward learned and execution-aware control. AFlow searches code-represented workflows with execution feedback (Zhang et al., 2025b), while MaAS learns a query-conditioned distribution over multi-agent architectures (Zhang et al., 2025a). ARC similarly treats workflows, tools, prompts, and compute budgets as query-wise configuration decisions (Taparia et al., 2026). Flow and DyFlow revise workflow structure during execution (Niu et al., 2025; Wang et al., 2025), and FlowSteer learns a lightweight policy that edits an executable workflow from intermediate states (Zhang et al., 2026). Closely related recent work also treats the execution harness around a frozen LLM as a learnable control layer (Yi & Song, 2026), while LEMON uses counterfactual re-execution to improve local orchestration decisions (Chen et al., 2026). PACE shares the frozen-executor/high-level-controller abstraction, but restricts the control variable to *when to replan*: the workflow structure is fixed and the controller selects only the next execution commitment.

**Temporal abstraction and replanning cost.** ReAct interleaves reasoning and acting so that observations can repeatedly revise the current plan (Yao et al., 2023). In reinforcement learning, options and semi-Markov decision processes formalize temporally extended actions (Sutton et al., 1999), and deliberation-cost methods explicitly penalize frequent option termination (Harb et al., 2018). PACE brings this temporal-abstraction view to stateful LLM workflows: Plan-$k$ is a temporally

extended high-level action, intermediate observations remain available to the executor, and the learned decision is the interval until the next high-level planning call.

**Multi-objective and preference-conditioned policy learning.** MORL studies sequential decisions with vector-valued returns and conflicting objectives (Roijers et al., 2013). Pareto Conditioned Networks, PD-MORL, and recent preference-controllable RL learn a single policy family that can respond to different desired trade-offs (Reymond et al., 2022; Basaklar et al., 2023; Yang et al., 2025). For tool-integrated LLM agents, ParetoPO explicitly optimizes accuracy–efficiency trade-offs (Li et al., 2026). PACE applies preference conditioning at the workflow-control level: quality retention and computational saving remain separate predictions until the requested preference is applied to select a commitment horizon.

# 3 PROBLEM FORMULATION

We consider a stateful LLM agent with a frozen executor and a trainable high-level workflow controller. At each workflow decision point, the controller determines how long the agent should proceed before invoking the planner again. We refer to this decision as *execution commitment*.

## 3.1 COMMITMENT-CONTROLLED WORKFLOWS

Let $s$ denote the current workflow state, including the interaction history and observations available to the agent. The controller selects a commitment horizon

$$k \in \mathcal{K} = \{1, 2, 4, \ldots, K_{\max}\}.$$

The case $k = 1$ corresponds to ReAct-style execution (Yao et al., 2023), where the agent replans after each action. For $k > 1$, the agent follows a Plan-$k$ workflow and executes multiple actions before returning control to the high-level planner.

Since each commitment may span multiple primitive environment steps, the resulting controller operates naturally at a semi-Markov time scale (Sutton et al., 1999). We seek a policy that may remain reactive in some parts of a trajectory while committing to longer execution segments in others.

## 3.2 PREFERENCE-CONDITIONED QUALITY–COST OBJECTIVE

We consider two deployment objectives: task quality and inference cost. A trajectory $\tau$ yields the vector outcome

$$\mathbf{G}(\tau) = \big[G_Q(\tau),\, -G_C(\tau)\big],$$

where $G_Q$ measures task performance and $G_C$ measures agent-side computational cost.

Let $C_0 > 0$ be a fixed environment-specific cost scale. A deployment preference $g = (g_Q, g_C)$ satisfies $g_Q, g_C \geq 0$ and $g_Q + g_C = 1$, and induces the scalarized episodic utility

$$U_g(\tau) = g_Q G_Q(\tau) - g_C\, G_C(\tau)/C_0.$$

Following the multi-policy view of multi-objective reinforcement learning (Roijers et al., 2013; Reymond et al., 2022; Yang et al., 2025), our goal is to learn a single preference-conditioned controller

$$\pi_\theta(k \mid s, g),$$

which selects the commitment horizon jointly from the current workflow state and the desired operating point. Varying $g$ therefore induces different quality–cost behaviors without retraining the executor. We report raw inference cost in the experiments; for compactness, cost targets in the learning equations below are expressed after the same fixed $C_0$ normalization.

# 4 METHOD

As illustrated in Figure 2, our method learns adaptive execution commitment through three components: counterfactual commitment profiles, a certainty-structured commitment policy, and on-policy workflow optimization. Restored-state comparisons supervise horizon-specific predictions of quality

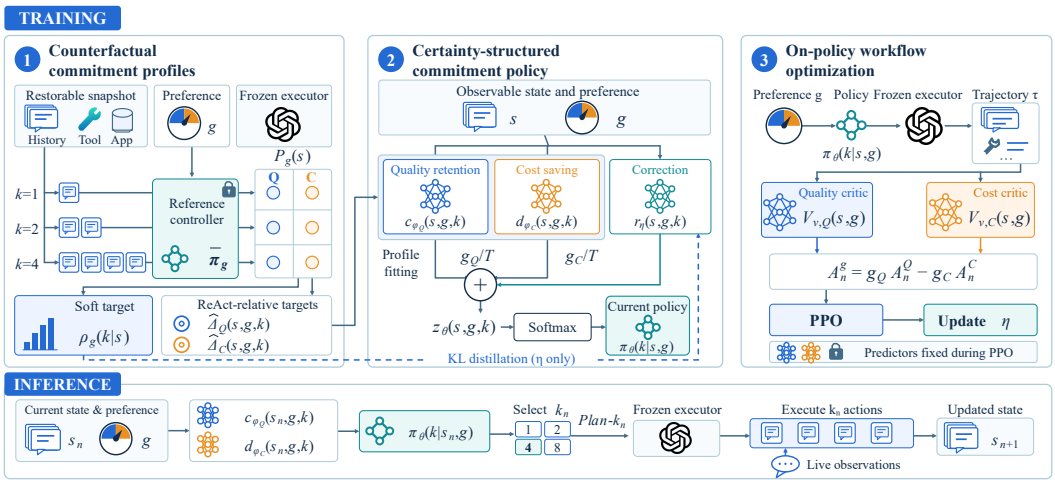



Figure 2: Overview of adaptive execution commitment. During training (top), restored-state comparisons produce quality–cost profiles that supervise horizon-specific predictors and policy distillation; on-policy optimization then updates the policy correction with predictors fixed. At inference (bottom), the controller selects a horizon $k_n$ from state $s_n$ and preference $g$, and the frozen executor follows that commitment while incorporating live observations. Counterfactual branching and value heads are used only during training.

retention and cost saving. These predictions guide preference-conditioned commitment selection, while a trainable correction is refined through distillation and on-policy interaction. Each training round refreshes the profiles under a newly frozen reference controller, allowing supervision to track the evolving policy. The executor remains fixed throughout.

### 4.1 Counterfactual Commitment Profiles

To obtain comparable supervision, we vary the next commitment horizon from the same workflow state while fixing the continuation policy. At each collection round, the current controller is frozen as $\bar{\pi}$. For a sampled state–preference pair $(s, g)$, the same execution snapshot is restored for every $k \in \mathcal{K}$, followed by Plan-$k$ and then the shared continuation $\bar{\pi}_g$ to termination. The snapshot preserves the environment, tool state, and interaction history needed for restoration, while learned models receive only deployment-observable information.

During Plan-$k$, observations remain available to the executor but do not trigger a new high-level plan. Control returns when the segment ends, the task terminates, or the plan becomes unexecutable. Thus, $k$ determines the interval between replanning decisions, while different branches may induce different downstream trajectories.

For each horizon, we collect $B$ suffix rollouts and estimate

$$\widehat{Q}_i^{\bar{\pi}_g}(s,k) = \frac{1}{B}\sum_{b=1}^{B} Y_i^{(b)}(s,g,k), \qquad i \in \{Q, C\}, \tag{1}$$

where $Y_Q^{(b)}$ is terminal task quality and $Y_C^{(b)}$ is normalized remaining inference cost from the restored state onward. These estimates form the *counterfactual commitment profile* $\mathcal{P}_g(s)$.

Under the collection preference, the profile defines

$$\begin{aligned} \widehat{U}_g(s,k) &= g_Q \widehat{Q}_Q^{\bar{\pi}_g}(s,k) - g_C \widehat{Q}_C^{\bar{\pi}_g}(s,k), \\ \rho_g(k \mid s) &= \frac{\exp\big(\widehat{U}_g(s,k)/T\big)}{\sum_{k' \in \mathcal{K}} \exp\big(\widehat{U}_g(s,k')/T\big)}, \end{aligned} \tag{2}$$

where $T > 0$ controls target sharpness. Each profile remains tied to its collection preference and frozen continuation, and is refreshed when the reference controller changes.

### 4.2 CERTAINTY-STRUCTURED COMMITMENT POLICY

Counterfactual profiles require executing multiple futures, so we distill them into horizon-specific predictors that can be evaluated without branching at deployment. We retain quality retention and computational saving as separate predictions for every candidate horizon.

**Horizon-specific planning certainty.** Let $k_1 = 1 < \cdots < k_M$ denote the candidate horizons. Taking ReAct as the reference, we construct the signed targets

$$\begin{aligned} \widehat{\Delta}_Q(s,g,k) &= \widehat{Q}_Q^{\bar{\pi}_g}(s,k) - \widehat{Q}_Q^{\bar{\pi}_g}(s,k_1), \\ \widehat{\Delta}_C(s,g,k) &= \widehat{Q}_C^{\bar{\pi}_g}(s,k_1) - \widehat{Q}_C^{\bar{\pi}_g}(s,k). \end{aligned} \tag{3}$$

Positive $\widehat{\Delta}_Q$ indicates a quality improvement, while positive $\widehat{\Delta}_C$ indicates a computational saving. Retaining signed, horizon-specific targets preserves both gains and losses across candidate commitments.

Two estimators, $c_{\phi_Q}(s,g,k)$ and $d_{\phi_C}(s,g,k)$, predict these quantities. We use *planning certainty* operationally as horizon-specific quality retention relative to ReAct under the reference continuation; $d_{\phi_C}$ predicts the corresponding cost saving. Both predictors are anchored at zero for $k_1$.

The estimators are fitted with separate regression objectives:

$$\begin{aligned} \mathcal{L}_Q &= \mathbb{E}_{(s,g)\sim\mathcal{D}}\left[\frac{1}{M-1}\sum_{j=2}^{M}\big(c_{\phi_Q}(s,g,k_j) - \widehat{\Delta}_Q(s,g,k_j)\big)^2\right], \\ \mathcal{L}_C &= \mathbb{E}_{(s,g)\sim\mathcal{D}}\left[\frac{1}{M-1}\sum_{j=2}^{M}\big(d_{\phi_C}(s,g,k_j) - \widehat{\Delta}_C(s,g,k_j)\big)^2\right]. \end{aligned} \tag{4}$$

Here $\mathcal{D}$ contains the latest collection round. The estimators have separate trainable parameters and receive only their respective outcome supervision. Their input $g$ identifies the continuation context under which those outcomes are evaluated. This preserves the quality and cost components as distinct predictions before they are combined for policy selection. Both estimators are refitted when the reference controller changes.

**Profile-guided commitment decisions.** The predicted quality retention and cost saving provide value-informed policy logits. We combine them under the requested preference and add a lightweight trainable correction $r_\eta(s,g,k)$:

$$\begin{aligned} z_\theta(s,g,k) &= \frac{g_Q c_{\phi_Q}(s,g,k) + g_C d_{\phi_C}(s,g,k)}{T} + r_\eta(s,g,k), \\ \pi_\theta(k \mid s,g) &= \frac{\exp z_\theta(s,g,k)}{\sum_{k'\in\mathcal{K}} \exp z_\theta(s,g,k')}, \end{aligned} \tag{5}$$

where $\theta = (\phi_Q, \phi_C, \eta)$. The correction receives the observable workflow state and preference and produces one output per horizon. It is initialized to zero and subsequently trained to adjust commitment selection, while the outcome estimators retain their supervised meanings.

We distill the counterfactual targets into the policy through

$$\mathcal{L}_{\text{pref}} = \mathbb{E}_{(s,g)\sim\mathcal{D}}\left[D_{\text{KL}}(\rho_g(\cdot \mid s) \,\|\, \pi_\theta(\cdot \mid s,g))\right]. \tag{6}$$

Profile predictions are detached during this update, so $\mathcal{L}_{\text{pref}}$ trains $\eta$, while $\phi_Q$ and $\phi_C$ are trained with $\mathcal{L}_Q$ and $\mathcal{L}_C$, respectively. This separation lets the policy incorporate preference-weighted supervision without changing the quantities represented by the outcome predictors.

### 4.3 ON-POLICY WORKFLOW OPTIMIZATION

Counterfactual targets describe local choices under a frozen reference. Prediction error and changes to subsequent decisions can make this reference differ from the policy preferred during continued execution. We therefore alternate profile fitting with on-policy optimization. Each episode begins

with a preference sampled from a fixed training distribution and retains it until termination. At every workflow boundary, the controller samples $k_n \sim \pi_\theta(\cdot \mid s_n, g)$, executes the selected segment, and chooses a new horizon when control returns.

We train separate value heads $V_{\nu,Q}(s,g)$ and $V_{\nu,C}(s,g)$ for quality and positive cost-to-go. The quality reward $r_n^Q$ is zero until completion, when it equals the terminal task score; $r_n^C$ aggregates the normalized inference cost incurred during the workflow segment. For finite episodes, we use undiscounted returns so that segment rewards sum to the task-quality and total-cost objectives. Generalized advantage estimation (Schulman et al., 2016) is applied at workflow boundaries:

$$\begin{aligned} \delta_n^i &= r_n^i + (1-d_n)V_{\nu_{\text{old}},i}(s_{n+1},g) - V_{\nu_{\text{old}},i}(s_n,g), \\ A_n^i &= \delta_n^i + \lambda(1-d_n)A_{n+1}^i, \qquad i \in \{Q,C\}, \\ A_n^g &= g_Q A_n^Q - g_C A_n^C. \end{aligned} \tag{7}$$

Here $d_n$ indicates task termination and $\lambda$ controls the trace over workflow decisions. The value heads are fitted to detached targets $\widehat{R}_n^i = A_n^i + V_{\nu_{\text{old}},i}(s_n,g)$, with zero continuation value at true terminal states.

During each on-policy phase, the profile estimators remain fixed and a PPO-style clipped update (Engstrom et al., 2020) updates the correction parameters $\eta$. With $\phi = (\phi_Q, \phi_C)$, define the workflow-level policy ratio $w_n(\eta) = \pi_{\phi,\eta}(k_n \mid s_n, g)/\pi_{\phi,\eta_{\text{old}}}(k_n \mid s_n, g)$. Writing $w_n = w_n(\eta)$ for brevity, we minimize

$$\begin{aligned} \mathcal{L}_{\text{policy}} = &-\mathbb{E}_{\tau\sim\mathcal{B}}\left[\sum_{n\in\tau} \min\{w_n A_n^g,\ \text{clip}(w_n, 1-\epsilon, 1+\epsilon)A_n^g\}\right] \\ &+ \lambda_{\text{pref}}\mathcal{L}_{\text{pref}}, \end{aligned} \tag{8}$$

where $\mathcal{B}$ contains fresh on-policy episodes and $\epsilon$ is the clipping parameter. The surrogate is summed over workflow decisions and averaged over episodes, with one policy ratio for each selected horizon regardless of its realized duration. The value heads are fitted separately; policy and value updates leave the profile estimators and executor fixed.

Each outer iteration freezes a reference controller, collects profiles, fits the outcome estimators, and distills the commitment policy. Fresh trajectories are collected after these updates for PPO optimization. Counterfactual targets remain fixed within the iteration, providing reference-policy supervision, and are replaced at the next collection round to track the evolving continuation. The correction is carried across rounds and refitted to the latest supervision.

## 5 Experiments

We evaluate PACE from three complementary perspectives. First, we examine the quality–cost operating points induced by different commitment policies and ask whether adaptive commitment improves the achievable trade-off over fixed and learned alternatives. Second, we study how a single conditional controller responds to different deployment preferences, including preferences not used during training, and isolate the roles of counterfactual outcome representation and on-policy refinement. Finally, a controlled finite-state study isolates the value of feedback-dependent commitment under known dynamics, allowing us to separate state-dependent scheduling from improvements that can already be obtained by a fixed or time-varying open-loop schedule.

### 5.1 Benchmark Setup

We evaluate on 134 unseen text-only ALFWorld tasks and 500 local WebShop test instructions (Shridhar et al., 2021; Yao et al., 2022). Quality is ALFWorld task success and WebShop native terminal reward, reported on a 0–100 scale. All controllers share the same frozen executor, observable state, $\mathcal{K} = \{1,2,4,8\}$, primitive-step limits, and early-return rules; observations remain available during committed execution. Training profiles exclude development and test tasks. We train on $\mathcal{G}_{\text{seen}} = \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and evaluate interpolation on $\mathcal{G}_{\text{held}} = \{0.2, 0.4, 0.6, 0.8\}$ using one validation-selected checkpoint per conditional method. Utility is

$$U_g = g_Q J_Q - (1-g_Q)J_C/C_0,$$

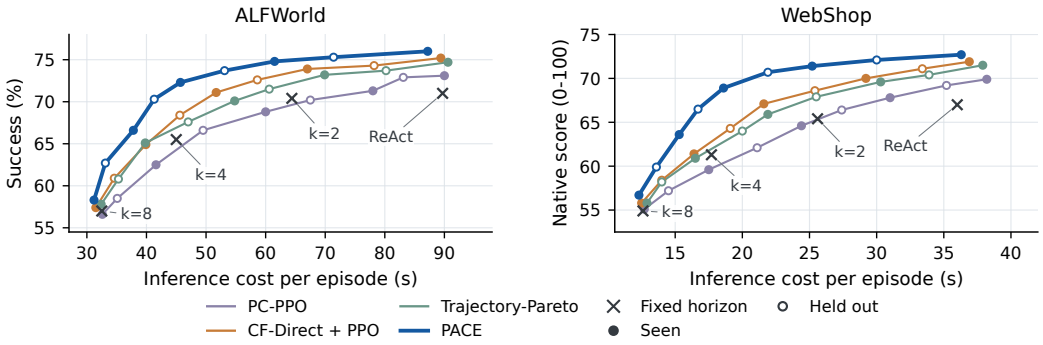



Figure 3: Quality–cost operating points of four conditional controllers across nine preferences. Crosses mark fixed horizons. Filled and open circles denote seen and held-out preferences.

with $C_0 = 120/48$ s for ALFWorld/WebShop. Deployment cost includes all executor and controller inference, including failed episodes; training budgets are 120/96 GPU hours and include counterfactual collection and optimization.

Core comparators are ReAct ($k = 1$) (Yao et al., 2023), a query-level horizon selector, PC-PPO, CF-Direct + PPO, and Trajectory-Pareto. PC-PPO and CF-Direct use the same PPO-style clipped update (Engstrom et al., 2020); CF-Direct uses counterfactual policy supervision without separate quality/cost prediction. Trajectory-Pareto adapts ParetoPO's trajectory-ranking mechanism to controller-level horizon decisions (Li et al., 2026). Representation comparisons use an identical profile bank; fixed horizons, Fixed mixture, and Multi-Scalar PPO are reported in Appendix A.

### 5.2 Main Quality–Cost Trade-offs

Figure 3 compares fixed execution horizons with learned commitment policies. Fixed horizons provide static reference points, whereas preference-conditioned controllers trace nine requested operating points. The relevant comparison is therefore not a single scalar score, but the quality attainable across different levels of inference expenditure.

The operating points place PACE above the principal learned comparators over much of the displayed cost range. At the common preference $g_Q = 0.5$, PACE achieves 72.3% success at 45.7 s on ALFWorld and a WebShop score of 68.9 at 18.6 s. Relative to CF-Direct + PPO, these points correspond to quality improvements of 1.2 percentage points and 1.8 score points together with cost reductions of 11.6% and 13.9%, respectively. Thus, at this operating preference, explicitly modeling the downstream quality and cost consequences of alternative commitments yields a simultaneous improvement in task outcome and deployment expenditure rather than merely moving along the same quality–cost trade-off.

Table 1 separates preference response from point-set coverage. Average utility,

$$\mathrm{AU}_{\mathcal{G}} = |\mathcal{G}|^{-1} \sum_{g \in \mathcal{G}} U_g(\pi_g),$$

scores the policy actually returned for each requested preference. $\mathrm{HV}_9$ instead measures the region dominated by the nine $(J_Q, -J_C/C_0)$ operating points relative to $(0, -1)$, independent of which request generated a particular point. Duplicate operating points add no additional area. AU therefore measures whether the controller responds appropriately to the requested trade-off, while HV summarizes the coverage of the resulting quality–cost set. HV is computed independently for each training seed and then averaged.

Across the full preference grid, PACE improves both request-conditioned utility and operating-set coverage. Relative to CF-Direct + PPO, seen/held-out AU increases from 0.175/0.160 to 0.191/0.182 on ALFWorld and from 0.151/0.133 to 0.173/0.161 on WebShop, while $\mathrm{HV}_9$ rises from 0.529 to 0.544 and from 0.501 to 0.521. The improvement therefore extends beyond a single operating point and is retained on held-out interpolation preferences.

Table 1: Preference-conditioned performance of the six core comparators. Larger values are better.

| | ALFWorld | | | WebShop | | |
|---|---|---|---|---|---|---|
| Method | $AU_{seen}$ | $AU_{held}$ | $HV_9$ | $AU_{seen}$ | $AU_{held}$ | $HV_9$ |
| ReAct | -0.019 | -0.019 | 0.179 | -0.040 | -0.040 | 0.168 |
| Query selector | 0.157 | 0.146 | 0.509 | 0.131 | 0.111 | 0.472 |
| PC-PPO | 0.148 | 0.136 | 0.502 | 0.128 | 0.109 | 0.479 |
| CF-Direct + PPO | 0.175 | 0.160 | 0.529 | 0.151 | 0.133 | 0.501 |
| Trajectory-Pareto | 0.167 | 0.151 | 0.519 | 0.144 | 0.126 | 0.491 |
| PACE | **0.191** | **0.182** | **0.544** | **0.173** | **0.161** | **0.521** |

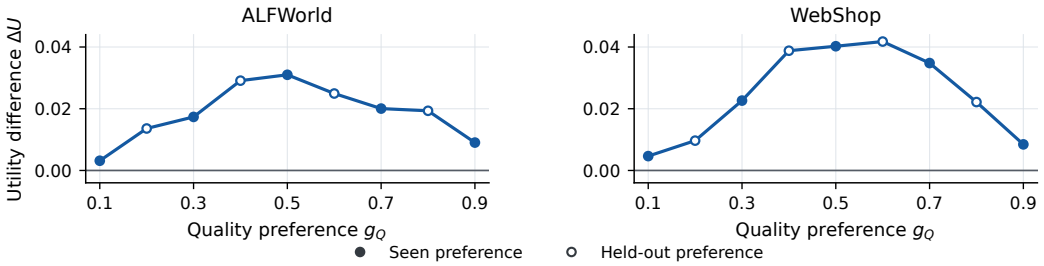



Figure 4: Utility difference between PACE and CF-Direct + PPO across preferences. Open markers denote held-out weights; the horizontal line marks zero difference.

Supplementary controls remain competitive on individual summaries: Fixed mixture reaches a WebShop seen AU of 0.159, while Multi-Scalar PPO reaches held-out AU values of 0.173 and 0.152 on ALFWorld and WebShop. These baselines either rely on static commitment or separate preference-specific controllers, whereas PACE represents state- and preference-dependent commitment within a single conditional policy. Full results are reported in Appendix A.

### 5.3 Preference Response and Components

Average utility can hide where gains occur across preferences. Figure 4 therefore reports

$$\Delta U(g) = U_g(\text{PACE}) - U_g(\text{CF-Direct + PPO})$$

at each requested preference. Gains are largest at intermediate weights and smaller near the extremes, indicating preference-dependent improvements rather than a uniform shift. Open markers denote held-out interpolation preferences.

Table 2 separately examines outcome representation and on-policy refinement. Because the two blocks use different control protocols, their gains should not be interpreted as additive.

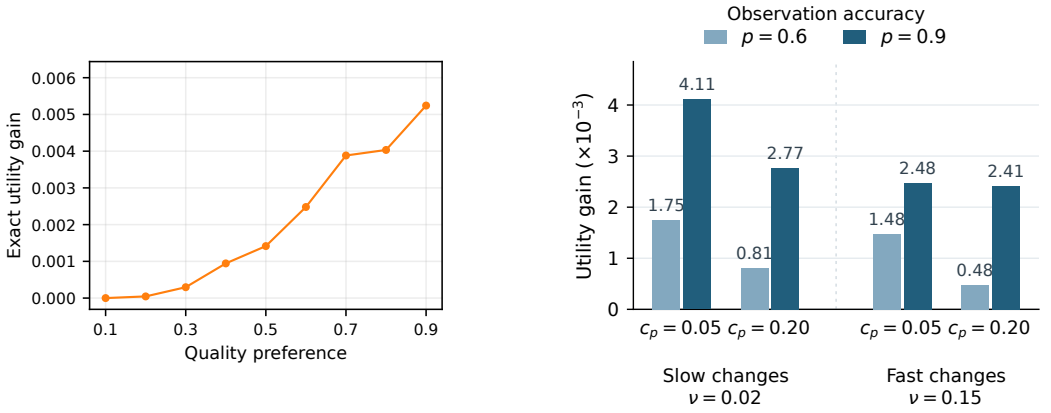
The first block holds the frozen-reference profile bank fixed across methods. CF-Direct fits the counterfactual commitment distribution directly, Scalar-Profile predicts one preference-weighted utility per horizon, and PACE retains the quality and cost components separately until the deployment preference is applied. Under identical profile data, the two-component representation achieves the highest AU on both environments, suggesting that preserving the structure of the quality–cost trade-off is useful beyond directly distilling a scalar policy target.

### 5.4 Controlled Analysis

A finite partially observed simulator isolates the value of feedback-dependent scheduling under known dynamics. Each episode contains $H = 12$ primitive steps. A hidden binary mode switches with probability $\nu$ and is observed with accuracy $p$. At a planning boundary, a frozen Bayesian planner selects an action from its current belief; after the controller selects $k \in \mathcal{K}$, the chosen action is repeated until the next planning boundary. Quality is mean action accuracy, while cost is $c_p N_{\text{plan}} + 0.01H$, normalized by 2.52.

Table 2: Outcome representation. Bold denotes the best AU within each protocol block.

| Variant | ALFWorld | | WebShop | |
|---|---|---|---|---|
| | $AU_9$ | Train (GPU h) | $AU_9$ | Train (GPU h) |
| *Outcome representation — identical profiles* | | | | |
| CF-Direct + PPO | 0.174 | 110.9 | 0.151 | 88.9 |
| Scalar-Profile + PPO | 0.179 | 113.0 | 0.160 | 91.4 |
| PACE | **0.191** | 116.4 | **0.171** | 94.0 |
| *On-policy refinement — common refresh schedule* | | | | |
| No-PPO | 0.174 | 98.4 | 0.157 | 78.4 |
| PACE | **0.188** | 120.0 | **0.168** | 96.0 |




(a) Feedback value across preferences.

(b) Adaptive advantage across conditions.

Figure 5: Exact utility gains of adaptive DP over open-loop DP. Left: gains across preferences, averaged over eight conditions. Right: gains in each condition, averaged over nine preferences. Colors denote observation accuracy $p$; groups distinguish switch probability $\nu$ and planning cost $c_p$.

We compare three nested scheduling classes: a fixed horizon, an open-loop DP that varies commitment with remaining time, and an adaptive DP that also conditions on the posterior belief from execution history. Thus, the adaptive–open-loop gap isolates the value of feedback-dependent commitment beyond time-varying scheduling. Full details appear in the supplementary material (Appendix C).

The adaptive gain over open-loop DP averages 0.00204, increasing from 0 at $g_Q = 0.1$ to 0.00524 at $g_Q = 0.9$. More accurate observations yield larger gains in every matched setting; the maximum is 0.00411 at $p = 0.9$, $\nu = 0.02$, and $c_p = 0.05$. Higher planning cost or faster mode changes reduce this gain, indicating that state-dependent commitment is most valuable when execution feedback is informative enough to affect replanning decisions.

## 6 Conclusion

We presented PACE, a framework for preference-conditioned adaptive execution commitment in LLM workflows. Treating commitment as a sequential decision unifies ReAct-style execution and longer Plan-$k$ segments, allowing planning frequency to adapt as observations accumulate. State-restored counterfactual evaluation compares horizons from identical workflow histories under a shared frozen continuation, providing comparable estimates of downstream quality and cost. These profiles support a single controller that combines horizon-specific quality-retention and cost-saving predictions under the requested preference and refines its decisions through on-policy optimization. Counterfactual branching remains confined to training. The controlled study quantifies how the value of feedback-dependent scheduling varies with observation reliability, environmental persistence, and planning cost. The ALFWorld and WebShop protocol evaluates this perspective against fixed, query-level, and learned commitment strategies.

## AI USE STATEMENT

In this work, we used generative AI tools to help develop conceptual frameworks, design and provide feedback on research methodology and experiments, and assist with method implementation. We also used generative AI tools to draft parts of the paper, summarize and analyze existing literature, and edit the manuscript to improve readability. We did not use generative AI tools for other tasks requiring disclosure. We reviewed all AI-assisted work. Implementation suggestions were tested through our experimental pipeline, and literature-based statements were verified against the cited sources. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

## REPRODUCIBILITY STATEMENT

We provide the methodological, experimental, and analytical details needed to reproduce the results reported in this work. The supplementary material specifies the benchmark protocol, comparator definitions, preference and cost metrics, component-analysis controls, and repeated-evaluation procedures. It also gives the complete controlled-simulator dynamics and exact dynamic programs, proofs and theoretical properties of adaptive commitment, the detailed training and execution procedure for PACE, and the statistical aggregation and reporting conventions used in the experiments. Additional implementation and reporting details, including profile freshness, deployment-cost accounting, early-return handling, and notation, are provided in the supplementary material.

## REFERENCES


Toygun Basaklar, Suat Gumussoy, and Umit Y. Ogras. PD-MORL: Preference-driven multi-objective reinforcement learning algorithm. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=zS9sRyaPFlJ.

Xudong Chen, Yixin Liu, Hua Wei, and Kaize Ding. LEMON: Learning executable multi-agent orchestration via counterfactual reinforcement learning. *arXiv preprint arXiv:2605.14483*, 2026. URL https://arxiv.org/abs/2605.14483.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep policy gradients: A case study on PPO and TRPO. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1etN1rtPB.

Jean Harb, Pierre-Luc Bacon, Martin Klissarov, and Doina Precup. When waiting is not an option: Learning options with a deliberation cost. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. doi: 10.1609/aaai.v32i1.11831.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, 2025.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, et al. Dspy: Compiling declarative language model calls into state-of-the-art pipelines. In *International Conference on Learning Representations*, 2024.

Sehoon Kim, Suhong Moon, Ryan Tabrizi, Nicholas Lee, Michael W. Mahoney, Kurt Keutzer, and Amir Gholami. An llm compiler for parallel function calling. In *International Conference on Machine Learning*, 2024.

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. *arXiv preprint arXiv:2606.16111*, 2026. URL https://arxiv.org/abs/2606.16111.

Boye Niu, Yiliao Song, Kai Lian, Yifan Shen, Yu Yao, Kun Zhang, and Tongliang Liu. Flow: Modularized agentic workflow automation. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/ba84da6921f3040b74ee163aa7451f53-Abstract-Conference.html.

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. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/28e50ee5b72e90b50e7196fde8ea260e-Abstract-Conference.html.

Mathieu Reymond, Eugenio Bargiacchi, and Ann Nowé. Pareto conditioned networks. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, pp. 1110–1118. International Foundation for Autonomous Agents and Multiagent Systems, 2022. URL https://ifaamas.org/Proceedings/aamas2022/pdfs/p1110.pdf.

Diederik M. Roijers, Peter Vamplew, Shimon Whiteson, and Richard Dazeley. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, 48:67–113, 2013. doi: 10.1613/jair.3987.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016.

Yu Shang, Yu Li, Keyu Zhao, Likai Ma, Jiahe Liu, Fengli Xu, and Yong Li. Agentsquare: Automatic llm agent search in modular design space. In *International Conference on Learning Representations*, 2025.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=0IOX0YcCdTn.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1–2):181–211, 1999. doi: 10.1016/S0004-3702(99)00052-1.

Aditya Taparia, Som Sagar, and Ransalu Senanayake. Learning to configure agentic ai systems. *arXiv preprint arXiv:2602.11574*, 2026. URL https://arxiv.org/abs/2602.11574.

Yanbo Wang, Zixiang Xu, Yue Huang, Xiangqi Wang, Zirui Song, Lang Gao, Chenxi Wang, Robert Tang, Yue Zhao, Arman Cohan, Xiangliang Zhang, and Xiuying Chen. DyFlow: Dynamic workflow framework for agentic reasoning. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/fe9910d2b03324faeb5371a9658277bb-Abstract-Conference.html.

Yucheng Yang, Tianyi Zhou, Mykola Pechenizkiy, and Meng Fang. Preference controllable reinforcement learning with advanced multi-objective optimization. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 71612–71647. PMLR, 2025. URL https://proceedings.mlr.press/v267/yang25ax.html.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, volume 35, 2022. doi: 10.52202/068431-1508. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/82ad13ec01f9fe44c01cb91814fd7b8c-Abstract-Conference.html.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. 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.

Haiwen Yi and Xinyuan Song. Learning to control LLM agent harnesses with offline reinforcement learning. *arXiv preprint arXiv:2607.05458*, 2026. URL https://arxiv.org/abs/2607.05458.

Guibin Zhang, Luyang Niu, Junfeng Fang, Kun Wang, Lei Bai, and Xiang Wang. Multi-agent architecture search via agentic supernet. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 75834–75852. PMLR, 2025a. URL https://proceedings.mlr.press/v267/zhang25bi.html.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xiong-Hui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. AFlow: Automating agentic workflow generation. In *International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=z5uVAKwmjf.

Mingda Zhang, Haoran Luo, Tiesunlong Shen, Qika Lin, Xiaoying Tang, Rui Mao, and Erik Cambria. FlowSteer: Interactive agentic workflow orchestration via end-to-end reinforcement learning. *arXiv preprint arXiv:2602.01664*, 2026. URL https://arxiv.org/abs/2602.01664.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning, acting, and planning in language models. In *International Conference on Machine Learning*, 2024.

# A BENCHMARK EVALUATION PROTOCOL

The benchmark evaluation asks whether a learned commitment controller can improve the quality–cost operating set while using the same frozen executor as its baselines. The main experiments use ALFWorld and WebShop.

## A.1 TASKS AND EVALUATION SETS

**ALFWorld.** The protocol uses the 134 unseen text-only ALFWorld evaluation tasks. Quality is binary task success and is reported as a percentage.

**WebShop.** The protocol uses 500 local WebShop test instructions. Quality is the native terminal reward and is reported as $100J_Q$. Exact-success rate can be reported as a complementary statistic, but native reward remains the primary WebShop quality measure.

Training profile collection excludes development and final evaluation tasks. All methods use the same observable state interface, commitment set $\{1, 2, 4, 8\}$, primitive-step limits, and early-return rules.

## A.2 PREFERENCE GRID AND CHECKPOINT SELECTION

Training preferences are

$$\mathcal{G}_{\text{seen}} = \{0.1, 0.3, 0.5, 0.7, 0.9\},$$

and held-out interpolation preferences are

$$\mathcal{G}_{\text{held}} = \{0.2, 0.4, 0.6, 0.8\}.$$

For every quality weight, $g_C = 1 - g_Q$. The preference remains fixed within an episode. Each conditional method uses one validation-selected checkpoint across all evaluation preferences. No preference-specific test-time retraining or test-time branching is used. The benchmark evaluation uses three independent training seeds and four executions per task and preference.

## A.3 DEPLOYMENT AND TRAINING COST ACCOUNTING

For a policy requested at preference $g$, deployment utility is

$$U_g(\pi) = g_Q J_Q(\pi) - g_C \frac{J_C(\pi)}{C_0}. \tag{9}$$

We use $C_0 = 120$ seconds for ALFWorld and $C_0 = 48$ seconds for WebShop. These are fixed within an environment and shared by all methods.

Deployment cost counts inference performed by the frozen executor and all inference-time controller components, including the state encoder, outcome predictors, and policy head. Failed episodes remain in the cost average.

Training expenditure is tracked separately. It includes ordinary trajectories, state restoration or reconstruction, counterfactual suffixes, predictor fitting, policy distillation, on-policy rollouts, and policy/value updates. The resource setting uses family budgets of 120 GPU-hours for ALFWorld and 96 GPU-hours for WebShop. The executor checkpoint, hardware, and decoding configuration are intentionally not specified in the current main text and are therefore not introduced here.

## A.4 COMPARATOR DEFINITIONS

The main benchmark table focuses on the core learned comparators and ReAct. Fixed Plan-$k$, Fixed mixture, and Multi-Scalar PPO are retained as supplementary controls because they answer different structural questions: whether static commitment is sufficient, whether random static mixing can reproduce the operating set, and whether a single conditional controller is competitive with separate preference-specific controllers.

Table 3: Comparator definitions used by the benchmark protocol.

| Method | Operational definition |
|---|---|
| Fixed-ReAct | Always select $k = 1$. |
| Fixed-Plan-$k$ | Always select the same $k \in \{2, 4, 8\}$ at every workflow boundary. |
| Fixed mixture | Sample one fixed-horizon policy at episode start and keep it throughout the episode. |
| Query selector | Choose one horizon from the initial observable task context and requested preference, then reuse it for the episode. |
| PC-PPO | Directly predict horizon logits from $(s, g)$ and optimize with the same workflow-level PPO objective, but without counterfactual profile supervision. |
| CF-Direct + PPO | Receive the same counterfactual policy targets as PACE, fit the commitment distribution directly, and retain on-policy PPO refinement; no separate quality/cost predictors are used. |
| Trajectory-Pareto | Adapt ParetoPO's trajectory-ranking mechanism to the same controller-level horizon actions and quality–cost outcomes. |
| Multi-Scalar PPO | Train separate PPO controllers for the five seen training preferences and account for their combined training expenditure. |
| PACE | Use restored-state profiles, separate horizon-specific quality/cost prediction, preference-conditioned selection, and on-policy residual refinement. |

Table 4: Quality and deployment cost at $g_Q = 0.5$. Bold marks the best entry in each column, including ties.

| | ALFWorld | | WebShop | | |
|---|---|---|---|---|---|
| Method | Success (%) ↑ | Cost (s) ↓ | Score ↑ | Success (%) ↑ | Cost (s) ↓ |
| Fixed-ReAct | 71.0 | 89.7 | 67.0 | 41.0 | 36.0 |
| Fixed-Plan-2 | 70.4 | 64.4 | 65.4 | 38.9 | 25.6 |
| Fixed-Plan-4 | 65.5 | 45.0 | 61.3 | 33.8 | 17.7 |
| Fixed-Plan-8 | 57.0 | **32.5** | 54.9 | 27.7 | **12.6** |
| Fixed mixture | 57.0 | **32.5** | 54.9 | 27.7 | **12.6** |
| Query selector | 68.7 | 56.8 | 63.2 | 37.6 | 22.4 |
| PC-PPO | 68.8 | 60.0 | 64.6 | 39.8 | 24.4 |
| CF-Direct + PPO | 71.1 | 51.7 | 67.1 | 42.6 | 21.6 |
| Trajectory-Pareto | 70.1 | 54.8 | 65.9 | 41.5 | 21.9 |
| Multi-Scalar PPO | 71.5 | 53.3 | 68.0 | 43.5 | 22.1 |
| PACE | **72.3** | 45.7 | **68.9** | **44.3** | 18.6 |

### A.5 AGGREGATE PREFERENCE METRICS

Achieved utility evaluates the policy actually assigned to each requested preference:

$$\mathrm{AU}_{\mathcal{G}} = \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} \left[ g_Q J_Q(\pi_g) - g_C \frac{J_C(\pi_g)}{C_0} \right]. \tag{10}$$

AU does not replace the requested policy by a more favorable point selected from the test operating set.

Hypervolume is computed from the nine operating points

$$(J_Q(\pi_g), -J_C(\pi_g)/C_0)\,, \qquad g_Q \in \{0.1, 0.2, \ldots, 0.9\},$$

with reference $(0, -1)$. Duplicate points add no area. In an experiment, hypervolume is computed per training seed and then aggregated across seeds.

Table 5: Preference response and operating-set coverage.

| | ALFWorld | | | WebShop | | |
|---|---|---|---|---|---|---|
| Method | $\mathrm{AU}_{\mathrm{seen}} \uparrow$ | $\mathrm{AU}_{\mathrm{held}} \uparrow$ | $\mathrm{HV}_9 \uparrow$ | $\mathrm{AU}_{\mathrm{seen}} \uparrow$ | $\mathrm{AU}_{\mathrm{held}} \uparrow$ | $\mathrm{HV}_9 \uparrow$ |
| Fixed mixture | 0.174 | 0.165 | 0.491 | 0.159 | 0.151 | 0.464 |
| Query selector | 0.157 | 0.146 | 0.509 | 0.131 | 0.111 | 0.472 |
| PC-PPO | 0.148 | 0.136 | 0.502 | 0.128 | 0.109 | 0.479 |
| CF-Direct + PPO | 0.175 | 0.160 | 0.529 | 0.151 | 0.133 | 0.501 |
| Trajectory-Pareto | 0.167 | 0.151 | 0.519 | 0.144 | 0.126 | 0.491 |
| Multi-Scalar PPO | 0.174 | 0.173 | 0.524 | 0.152 | 0.152 | 0.498 |
| PACE | **0.191** | **0.182** | **0.544** | **0.173** | **0.161** | **0.521** |

### A.6 COMMON-PREFERENCE RESULTS

### A.7 PREFERENCE RESPONSE AND COVERAGE

### A.8 REPEATED EVALUATION AND UNCERTAINTY

Table 5 gives the preference-level summary. The benchmark evaluation uses three independent training seeds and four execution replicates per task and preference. Seed-level summaries should be retained. Paired uncertainty calculations should preserve the dependence induced by using the same task identifiers and preference grid across methods.

## B COMPONENT ANALYSIS PROTOCOLS

The component analyses separate three questions: whether the counterfactual profile should be represented by separate quality and cost quantities, whether local counterfactual supervision benefits from sequential on-policy refinement, and whether the profile supervision must track the evolving continuation policy. The protocols differ and their estimated gains should not be summed as independent contributions.

### B.1 OUTCOME REPRESENTATION WITH AN IDENTICAL PROFILE BANK

The representation comparison uses one frozen-reference profile bank shared by all compared learners. Thus, the states, preferences, candidate horizons, reference continuation, and counterfactual outcomes are identical. Only the representation used to turn those profiles into a policy changes.

**CF-Direct + PPO.** The actor is fit directly to the counterfactual distribution $\rho_g(k \mid s)$ and then refined with PPO. It does not learn explicit quality-retention or cost-saving predictors.

**Scalar-Profile + PPO.** The actor predicts one preference-weighted relative utility for each candidate horizon and retains the same type of trainable residual. This control tests whether preserving quality and cost separately provides information beyond a single scalar profile.

**PACE.** The actor predicts horizon-specific relative quality and cost separately and combines them only after the requested preference is supplied.

Because all learners see the same profile bank but fitting costs differ, this is a matched-data comparison rather than an equal-compute comparison.

### B.2 SEQUENTIAL REFINEMENT

The No-PPO variant retains counterfactual profile collection, profile refresh, horizon-specific outcome prediction, and counterfactual distillation. It omits only the PPO refinement stage. The comparison therefore asks whether interaction under the updated controller provides additional value beyond the local counterfactual targets.

Table 6: The two blocks use different control protocols and are not additive decompositions.

| | ALFWorld | | WebShop | |
|---|---|---|---|---|
| Variant | $AU_9$ | Train (GPU h) | $AU_9$ | Train (GPU h) |
| *Outcome representation – identical profiles* | | | | |
| CF-Direct + PPO | 0.174 | 110.9 | 0.151 | 88.9 |
| Scalar-Profile + PPO | 0.179 | 113.0 | 0.160 | 91.4 |
| PACE | **0.191** | 116.4 | **0.171** | 94.0 |
| *On-policy refinement – common refresh schedule* | | | | |
| No-PPO | 0.174 | 98.4 | 0.157 | 78.4 |
| PACE | **0.188** | 120.0 | **0.168** | 96.0 |

The current protocol keeps the refresh schedule common rather than equalizing training compute. Training expenditure is therefore reported alongside AU.

### B.3 PROFILE REFRESH

The No-Refresh variant retains PPO but reuses the initial reference-profile supervision instead of recollecting profiles as the controller changes. Its purpose is to test whether local targets become stale when the continuation policy evolves.

### B.4 COMPONENT VALUES

For the equal-compute profile-refresh comparison, the utilities referenced are

| | ALFWorld | WebShop |
|---|---|---|
| Fixed profiles | 0.172 | 0.153 |
| Refreshed profiles | 0.192 | 0.170. |

### B.5 INTERPRETATION LIMITS

The three component comparisons should be read under their own controls. The representation comparison fixes the data but not fitting cost. The No-PPO comparison fixes the refresh schedule but reports different total training expenditure. The profile-refresh comparison fixes total compute. Consequently, the numerical differences cannot be added to obtain a decomposition of the full method's total gain.

## C CONTROLLED SIMULATOR AND EXACT REFERENCE POLICIES

The controlled study isolates the value of feedback-dependent commitment under known dynamics. Its exact policies serve as references for the commitment mechanism; they are not trained neural instances of PACE.

### C.1 ENVIRONMENT DYNAMICS

Each episode contains $H = 12$ primitive steps and a latent binary mode

$$z_t \in \{0, 1\}.$$

The initial mode is uniform. At each primitive step, the mode switches with probability

$$\nu \in \{0.02, 0.15\}.$$

An observation identifies the current mode correctly with probability

$$p \in \{0.6, 0.9\}.$$

The controller observes the accumulated observation history but not the latent mode or intermediate correctness labels.

At each planning boundary, a frozen Bayesian planner updates its posterior belief using the observations available so far, selects the more probable mode, and emits the corresponding primitive action. If the controller selects horizon $k$, the planner's action is repeated for up to $k$ primitive steps. Observations generated during the segment enter the history but are not used to revise the high-level plan until the next planning boundary.

Terminal quality is

$$G_Q = \frac{1}{H} \sum_{t=0}^{H-1} \mathbf{1}[a_t = z_t]. \tag{11}$$

Cost is

$$G_C = c_p N_{\text{plan}} + 0.01H, \tag{12}$$

where

$$c_p \in \{0.05, 0.20\}$$

is the cost of one high-level planning call. A common scale

$$C_0 = 2.52$$

is used across the eight simulator conditions. The full factorial combination of $(p, \nu, c_p)$ gives eight conditions, each evaluated at $g_Q \in \{0.1, 0.2, \ldots, 0.9\}$.

### C.2 Reference policy classes

**Fixed horizon.** A fixed policy always uses one horizon from $\{1, 2, 4, 8\}$. The best episode-fixed reference selects the best of these four policies for each condition and requested preference.

**Open-loop schedule.** The controller optimizes a time-varying horizon schedule in advance. The frozen Bayesian planner still updates its belief at the scheduled planning boundaries, but the future horizon choices do not condition on the realized observations.

**Adaptive belief policy.** The controller conditions the next horizon on the current posterior belief and remaining episode length. This policy class contains the open-loop and fixed classes as restrictions.

### C.3 Exact dynamic programming

Let $b$ denote the current posterior belief over the latent mode and $r$ the number of primitive steps remaining. The adaptive reference has recursion

$$V_g(b, r) = \max_{k \in \mathcal{K}_r} \mathbb{E}\left[R_g(b, k) + V_g(b', r - \ell)\right], \tag{13}$$

where $\mathcal{K}_r$ contains feasible commitment horizons at the remaining length, $\ell = \min(k, r)$, $R_g$ is the preference-weighted contribution of the current segment, and the expectation is over latent transitions and observations generated before the next planning boundary. The successor belief $b'$ incorporates those observations.

The open-loop reference uses the same dynamics but constrains horizon choice to be independent of the realized belief. The fixed policies impose the stronger restriction that the same horizon is reused throughout the episode.

### C.4 Bayesian filtering in the controlled simulator

The finite simulator permits an explicit belief-state description. Let

$$b_t = \Pr(z_t = 1 \mid o_{0:t})$$

be the posterior probability of latent mode 1 after observing $o_t$. Between primitive steps, the latent mode flips with probability $\nu$. Before seeing the next observation, the one-step predictive belief is

$$\widetilde{b}_{t+1} = (1 - \nu) b_t + \nu(1 - b_t). \tag{14}$$

For a binary observation $o_{t+1} \in \{0, 1\}$ with accuracy $p$, define

$$L_1(o) = \Pr(o \mid z = 1), \qquad L_0(o) = \Pr(o \mid z = 0),$$

where $L_1(1) = L_0(0) = p$ and $L_1(0) = L_0(1) = 1 - p$. The posterior update is

$$b_{t+1} = \frac{L_1(o_{t+1})\widetilde{b}_{t+1}}{L_1(o_{t+1})\widetilde{b}_{t+1} + L_0(o_{t+1})(1 - \widetilde{b}_{t+1})}. \tag{15}$$

At a planning boundary, the frozen Bayesian planner selects the more probable mode as the action. If the controller commits for $k$ steps, that action is held fixed throughout the segment even though observations continue to arrive. Those observations are filtered into the next belief only for the purpose of the next high-level planning decision.

### C.5 Segment value and exact recursion

Let $a(b) = \mathbf{1}[b \geq 1/2]$ denote the action selected by the frozen planner at belief $b$. For remaining horizon $r$ and requested commitment $k$, let $\ell = \min(k, r)$. The expected quality contribution of the segment is

$$q(b, k, r) = \frac{1}{H} \sum_{j=0}^{\ell-1} \Pr\left(z_{t+j} = a(b) \mid b\right). \tag{16}$$

The corresponding segment cost is

$$c(k, r) = c_p + 0.01\ell, \tag{17}$$

because selecting the segment invokes the high-level planner once and incurs the primitive execution charge for each realized step.

Let $\mathcal{O}^\ell$ denote the possible observation sequences received during the segment. For a sequence $\mathbf{o}$ of length $\ell$, repeated application of Eqs. equation 14– equation 15 gives the next boundary belief $F(b, \mathbf{o})$. The adaptive recursion can therefore be written more explicitly as

$$V_g(b, r) = \max_{k \in \mathcal{K}_r} \Bigg\{ g_Q q(b, k, r) - g_C \frac{c(k, r)}{C_0} + \sum_{\mathbf{o} \in \mathcal{O}^\ell} \Pr(\mathbf{o} \mid b, k) V_g(F(b, \mathbf{o}), r - \ell) \Bigg\}, \tag{18}$$

with terminal condition $V_g(b, 0) = 0$. This is equivalent to the more compact recursion used in the main appendix, but makes explicit that the only adaptive state variable is the belief induced by accumulated observations together with remaining time.

### C.6 Closed-form expected accuracy during a committed segment

The latent two-state dynamics yield a useful closed-form expression for the expected correctness of the action held fixed within a segment. Let the planner select action $a \in \{0, 1\}$ at the planning boundary and define

$$q_0 = \Pr(z_t = a \mid b) = \max\{b, 1 - b\}, \tag{19}$$

where the last equality follows from the MAP action used by the frozen planner. Under symmetric mode switching with probability $\nu$, the probability that the latent mode still matches the fixed action $j$ primitive transitions later is

$$q_j = \frac{1}{2} + \left(q_0 - \frac{1}{2}\right)(1 - 2\nu)^j. \tag{20}$$

*Derivation.* The match indicator itself follows a two-state Markov chain that flips with probability $\nu$. Therefore

$$q_{j+1} = (1 - \nu)q_j + \nu(1 - q_j) = \nu + (1 - 2\nu)q_j.$$

Subtracting $1/2$ from both sides gives $q_{j+1} - 1/2 = (1 - 2\nu)(q_j - 1/2)$, which yields Eq. equation 20 by recursion.

For a realized segment length $\ell$, the expected number of correct primitive actions contributed by the segment is

$$\sum_{j=0}^{\ell-1} q_j = \frac{\ell}{2} + \left(q_0 - \frac{1}{2}\right) \frac{1-(1-2\nu)^\ell}{2\nu}, \tag{21}$$

for $\nu > 0$, with the continuous limit $\ell q_0$ as $\nu \to 0$. Dividing by $H$ gives the segment's expected contribution to terminal quality.

Equation equation 21 isolates one aspect of the commitment trade-off. When the current belief is decisive ($q_0$ far from $1/2$) and the latent mode is persistent (small $\nu$), the quality of holding the current action decays slowly with commitment length. This does not by itself determine the optimal horizon, because future observations affect the belief available at the next planning boundary and planning calls incur cost. The dynamic program accounts for both effects jointly.

### C.7 Open-loop regret as the cost of suppressing belief dependence

At remaining length $r$, let

$$k^*(b, r) \in \arg\max_k Q_g(b, r, k)$$

be an adaptive Bellman-optimal horizon. For any belief-independent open-loop choice $\bar{k}_r$, define its local regret

$$\mathcal{R}_g(b, r; \bar{k}_r) = Q_g(b, r, k^*(b, r)) - Q_g(b, r, \bar{k}_r) \geq 0. \tag{22}$$

If $\mu_r$ is the distribution over beliefs reaching that decision depth under the open-loop schedule, the expected loss incurred at that depth is

$$\mathbb{E}_{b\sim\mu_r}[\mathcal{R}_g(b, r; \bar{k}_r)]. \tag{23}$$

The adaptive-open-loop value gap accumulates such belief-dependent losses over reachable decision depths, accounting for the fact that earlier horizon choices also change later belief distributions. This expression provides a useful interpretation of the exact DP gaps reported in the main text: they quantify the cost of suppressing dependence of the commitment decision on accumulated evidence, after already allowing the schedule to vary with time.

### C.8 Nested policy classes and reference dominance

Let $\Pi_{\text{fixed}}$, $\Pi_{\text{open}}$, and $\Pi_{\text{adapt}}$ denote the episode-fixed, open-loop, and belief-adaptive commitment classes defined above. They satisfy

$$\Pi_{\text{fixed}} \subseteq \Pi_{\text{open}} \subseteq \Pi_{\text{adapt}}. \tag{24}$$

**Proposition C.1 (reference-policy dominance).** For every simulator condition and preference,

$$V_g^{\text{adapt}} \geq V_g^{\text{open}} \geq V_g^{\text{fixed}}. \tag{25}$$

*Proof.* Each optimization is taken over a superset of the policies available to the preceding reference class. Therefore its optimum cannot be smaller. □

The proposition explains why the exact adaptive-minus-open-loop curve is nonnegative up to numerical precision. The numerical study is informative about the *size and location* of this gap, not about its sign.

### C.9 When does feedback-dependent adaptation have zero value?

The nesting result also clarifies when state-dependent horizon selection does not help. Let

$$Q_g(b, r, k)$$

denote the Bellman value of choosing $k$ at belief $b$ and remaining length $r$, including the optimal adaptive continuation afterward.

Table 7: Exact adaptive-DP utility gain across preferences, averaged over the eight simulator conditions.

| $g_Q$ | vs. best episode-fixed | vs. open-loop DP |
|---|---|---|
| 0.1 | 0.000625 | 0.000000 |
| 0.2 | 0.001296 | 0.000045 |
| 0.3 | 0.002172 | 0.000296 |
| 0.4 | 0.003020 | 0.000944 |
| 0.5 | 0.003229 | 0.001415 |
| 0.6 | 0.004458 | 0.002480 |
| 0.7 | 0.006132 | 0.003882 |
| 0.8 | 0.004519 | 0.004033 |
| 0.9 | 0.005808 | 0.005242 |

**Proposition C.2 (sufficient condition for zero adaptation value).** Suppose that for every reachable remaining length $r$ there exists a horizon $k_r$ that maximizes $Q_g(b, r, k)$ for *all* reachable beliefs $b$ at that $r$. Then an open-loop schedule can select $k_r$ at each remaining length and matches the adaptive value exactly.

*Proof.* Under the stated condition, the adaptive Bellman maximizer is independent of belief at every decision depth. Selecting the same maximizing horizon as a function of remaining length therefore reproduces the adaptive choice on every reachable history. □

Thus feedback-dependent adaptation is useful only when the information state changes which horizon is preferred at some decision point. This statement connects the controlled experiment directly to the paper's central question: when does the execution history actually require a different commitment decision?

**Proposition C.3 (strict advantage from heterogeneous local choices).** Assume there is a remaining length $r$ and two reachable beliefs $b_1, b_2$ with positive probability such that their sets of maximizing horizons are disjoint, and suppose the loss from choosing a nonmaximizing horizon is strictly positive at each belief. Then any open-loop rule that must choose the same horizon at that $r$ incurs positive expected regret relative to the adaptive controller.

*Proof.* An open-loop rule picks one horizon at that decision depth. Because the two maximizer sets are disjoint, that horizon is suboptimal for at least one of the two positive-probability beliefs. Strict suboptimality on an event of positive probability gives a strictly positive expected value gap. □

This proposition does not predict which simulator parameters produce the largest gap. Observation reliability, mode persistence, planning cost, and the preference jointly determine both the visited beliefs and the local regret from using a belief-independent horizon.

### C.10 Why no monotonic horizon rule is assumed

It can be tempting to infer that more accurate observations must always imply longer commitments, or that higher planning cost must always increase the adaptive advantage. Neither statement follows from the model. More informative future observations can increase the value of returning to the planner, while persistent latent modes can increase the value of committing to a plan formed from current evidence. Similarly, a higher planning-call cost can make both adaptive and open-loop schedules plan less frequently, reducing or increasing their gap depending on the state distribution. The controlled experiment therefore reports the exact DP solution rather than imposing a monotone certainty-to-horizon rule.

### C.11 Preference-wise exact adaptive gains

Table 7 reports the exact utility advantage of adaptive DP over the two restricted reference classes, averaged across the eight simulator conditions. These are the values summarized by the preference-wise controlled-analysis curve in the main text.

Table 8: Exact adaptive-DP utility gain by simulator condition, averaged across nine preferences.

| ID | $p$ | $\nu$ | $c_p$ | vs. best fixed | vs. open-loop |
|---|---|---|---|---|---|
| 1 | 0.6 | 0.02 | 0.05 | 0.002900 | 0.001752 |
| 2 | 0.6 | 0.02 | 0.20 | 0.003671 | 0.000813 |
| 3 | 0.6 | 0.15 | 0.05 | 0.001560 | 0.001482 |
| 4 | 0.6 | 0.15 | 0.20 | 0.000671 | 0.000485 |
| 5 | 0.9 | 0.02 | 0.05 | 0.006057 | 0.004107 |
| 6 | 0.9 | 0.02 | 0.20 | 0.008031 | 0.002766 |
| 7 | 0.9 | 0.15 | 0.05 | 0.002481 | 0.002481 |
| 8 | 0.9 | 0.15 | 0.20 | 0.002414 | 0.002414 |

The mean exact advantage over open-loop DP is $0.00204$. The adaptive and open-loop references tie at $g_Q = 0.1$, while the gap reaches $0.00524$ at $g_Q = 0.9$.

### C.12 Condition-level exact gains

Table 8 expands the eight conditions shown in the controlled-analysis figure. The values are exact utility gains averaged over the nine preferences.

These condition-level values support the qualitative interpretation in the main text. More accurate observations increase the adaptive opportunity in all matched settings in the current simulator. The magnitude also depends on how long information remains relevant and on the planning-call cost.

### C.13 Monte Carlo checks and exhaustive verification

In addition to exact expectations, the supplied numerical study uses $4{,}096$ independent simulator realizations for each of three sampling seeds and eight conditions, yielding $98{,}304$ independently generated episodes. Each realization is reused across the evaluated preferences and controller settings to form paired comparisons. This reuse improves comparison precision but does not increase the number of independent environment realizations.

The supplied study also includes an independent short-horizon verification at $H = 4$. All $256$ latent–observation sequences are enumerated for $144$ controller/parameter combinations. The largest absolute discrepancy between exhaustive enumeration and dynamic programming is $4.44 \times 10^{-16}$.

## D Theoretical Properties of Adaptive Commitment

Throughout this section, cost-valued action values and rewards use the same fixed $C_0$ normalization as the learning objective unless an equation explicitly includes $C_0$.

This section formalizes several properties that are implicit in the method construction in the main text. The purpose is to clarify what follows directly from the workflow-level formulation and what still depends on estimation, function approximation, or continued interaction. All statements are local to the existing PACE components. In particular, the results below do not claim that counterfactual distillation alone globally improves a policy, nor do they assume that the learned residual preserves a monotone ordering over commitment horizons.

### D.1 Workflow-boundary control as a semi-Markov process

Let $s_n$ be the observable workflow state at high-level decision boundary $n$, $k_n \in \mathcal{K}$ the selected commitment horizon, and $\tau_n$ the realized number of primitive executor steps before control returns. Because a Plan-$k$ segment can terminate early, $1 \le \tau_n \le k_n$ in general. For a frozen executor, a selected horizon induces a distribution over the next workflow state, realized duration, and segment-level outcomes. We write this kernel abstractly as

$$P_{\text{exec}}(s', \tau, r_Q, r_C \mid s, k). \tag{26}$$

Here $r_Q$ and $r_C$ denote the quality and positive cost contributions assigned to the workflow transition. In the episodic setting used in the paper, quality can remain zero until termination while cost accumulates across segments.

**Proposition D.1 (induced semi-Markov control problem).** Assume that $s_n$ contains the observable history used by the controller and that the executor/environment dynamics following a workflow boundary depend on the past only through this history and the selected horizon. Then the controller-level process

$$s_0 \xrightarrow{k_0,\tau_0} s_1 \xrightarrow{k_1,\tau_1} s_2 \cdots$$

is a semi-Markov decision process with action set $\mathcal{K}$ and transition kernel induced by Eq. equation 26.

*Proof.* Conditioned on $(s_n, k_n)$, the frozen executor together with the environment determines a distribution over the complete segment, including its duration, observations, costs, and terminal status. The next workflow state $s_{n+1}$ is constructed from the resulting observable history. No additional controller decision is taken within the segment. Hence a high-level action can span a random number of primitive transitions, which is exactly the semi-Markov structure. □

For the undiscounted episodic objectives used in the main text, the total controller-level return is unchanged by this temporal aggregation:

$$G_i(\tau) = \sum_{n=0}^{N(\tau)-1} r_n^i, \qquad i \in \{Q, C\}, \tag{27}$$

where $N(\tau)$ is the number of workflow decisions in the episode. Thus the controller may be trained at workflow boundaries without assigning separate policy actions to the executor's primitive steps.

### D.2 Counterfactual profiles as local action values

Fix a preference $g$ and a frozen continuation policy $\bar{\pi}_g$. For a restorable workflow state $s$ and forced current horizon $k$, define

$$Q_i^{\bar{\pi}_g}(s,k) = \mathbb{E}\left[Y_i(s,g,k) \mid s,\ K = k,\ \bar{\pi}_g\right], \qquad i \in \{Q, C\}, \tag{28}$$

where $Y_Q$ is terminal task quality and $Y_C$ is positive remaining cost from $s$ onward. The forced Plan-$k$ segment is part of the intervention; after it returns control, the same frozen continuation $\bar{\pi}_g$ is used for all candidate horizons.

**Proposition D.2 (unbiased profile estimation under exact restoration).** Suppose the restored snapshot reproduces the same state $s$, and the $B$ suffix rollouts for a fixed $(s, g, k, \bar{\pi}_g)$ are independent conditional on that state and continuation. Then

$$\widehat{Q}_i^{\bar{\pi}_g}(s,k) = \frac{1}{B}\sum_{b=1}^{B} Y_i^{(b)}(s,g,k) \tag{29}$$

is an unbiased estimator of Eq. equation 28, with

$$\mathrm{Var}\left[\widehat{Q}_i^{\bar{\pi}_g}(s,k)\right] = \frac{\mathrm{Var}[Y_i(s,g,k)]}{B}. \tag{30}$$

*Proof.* Both statements follow from linearity of expectation and the variance of an average of conditionally independent suffix outcomes. □

The proposition concerns the value under the frozen continuation used to collect the profile. It does not imply that the same numerical profile is the action value of a later updated controller. This distinction motivates the profile-refresh step studied below.

The relative targets used by PACE are

$$\Delta_Q(s,g,k) = Q_Q^{\bar{\pi}_g}(s,k) - Q_Q^{\bar{\pi}_g}(s,k_1), \tag{31}$$

$$\Delta_C(s,g,k) = Q_C^{\bar{\pi}_g}(s,k_1) - Q_C^{\bar{\pi}_g}(s,k), \tag{32}$$

with $k_1 = 1$. The sign convention in Eq. equation 32 means that positive $\Delta_C$ is a cost saving relative to ReAct.

If the two horizon values are estimated from independent suffix sets, the variance of the estimated quality difference is

$$\mathrm{Var}(\widehat{\Delta}_Q) = \frac{\sigma^2_{Q,k} + \sigma^2_{Q,1}}{B}, \tag{33}$$

and analogously for cost. If an implementation uses additional common random numbers across branches, a covariance term appears; PACE does not rely on such coupling for correctness.

### D.3 Relative utility decomposition

The profile utility under the frozen continuation is

$$U_g(s,k) = g_Q Q_Q^{\bar{\pi}_g}(s,k) - g_C Q_C^{\bar{\pi}_g}(s,k). \tag{34}$$

Subtracting the ReAct reference gives

$$U_g(s,k) - U_g(s,k_1) = g_Q \left[ Q_Q^{\bar{\pi}_g}(s,k) - Q_Q^{\bar{\pi}_g}(s,k_1) \right] \tag{35}$$

$$- g_C \left[ Q_C^{\bar{\pi}_g}(s,k) - Q_C^{\bar{\pi}_g}(s,k_1) \right] \tag{36}$$

$$= g_Q \Delta_Q(s,g,k) + g_C \Delta_C(s,g,k). \tag{37}$$

This identity explains why the method can predict quality retention and cost saving separately, then combine them only after the deployment preference is specified. It also shows that the use of a positive coefficient on the cost predictor in the policy logit is consistent with the fact that the predictor represents *saving*, not positive cost.

### D.4 Soft counterfactual target as an entropy-regularized local optimum

For a fixed profile, define $u_k = \widehat{U}_g(s,k)$ and consider a distribution $p$ over the finite horizon set $\mathcal{K}$. The target used in the method is

$$\rho_g(k \mid s) = \frac{\exp(u_k/T)}{\sum_{j \in \mathcal{K}} \exp(u_j/T)}. \tag{38}$$

**Proposition D.3 (variational characterization of the soft target).** For $T > 0$, Eq. equation 38 is the unique maximizer of

$$\max_{p \in \Delta(\mathcal{K})} \left[ \sum_k p_k u_k + T\,\mathcal{H}(p) \right], \qquad \mathcal{H}(p) = -\sum_k p_k \log p_k. \tag{39}$$

*Proof.* Introduce a Lagrange multiplier $\alpha$ for $\sum_k p_k = 1$. The first-order condition for an interior solution is

$$u_k - T(1 + \log p_k) - \alpha = 0,$$

which gives $p_k \propto \exp(u_k/T)$. Strict concavity of entropy on the simplex gives uniqueness. □

This proposition is deliberately local: the objective is defined using the counterfactual profile under a frozen continuation. It does not state that replacing the complete sequential policy by $\rho_g$ yields monotonic global policy improvement. In the algorithm, $\rho_g$ is used as a training target and the sequential controller is subsequently refined on policy.

### D.5 Exact recovery by the profile-derived policy

Let the learned outcome predictors equal the exact relative targets,

$$c_{\phi_Q}(s,g,k) = \Delta_Q(s,g,k), \qquad d_{\phi_C}(s,g,k) = \Delta_C(s,g,k),$$

and set the residual correction to zero. The profile-derived policy logit is then

$$z(s,g,k) = \frac{g_Q \Delta_Q(s,g,k) + g_C \Delta_C(s,g,k)}{T}. \tag{40}$$

By Eq. equation 35,

$$z(s,g,k) = \frac{U_g(s,k) - U_g(s,k_1)}{T}. \tag{41}$$

Softmax probabilities are invariant to subtracting the same constant from all logits. Therefore:

**Proposition D.4 (profile-target recovery).** If the two outcome predictors are exact and $r_\eta = 0$, the policy defined by Eq. equation 80 equals the counterfactual target $\rho_g(\cdot \mid s)$.

The proposition clarifies the role of the residual. The two predictors define a value-informed policy reference with an explicit quality/cost interpretation; $r_\eta$ is the component allowed to depart from that local reference during sequential optimization.

### D.6 Effect of predictor error on the profile-derived policy

We next quantify a local consequence of imperfect outcome prediction. Suppose for a fixed $(s, g)$ that

$$\max_k \left|c_{\phi_Q}(s,g,k) - \Delta_Q(s,g,k)\right| \le \varepsilon_Q, \qquad \max_k |d_{\phi_C}(s,g,k) - \Delta_C(s,g,k)| \le \varepsilon_C. \tag{42}$$

Define the preference-weighted prediction error

$$\varepsilon_g = g_Q \varepsilon_Q + g_C \varepsilon_C. \tag{43}$$

If the same residual is added to both the exact and approximate logit vectors, then each logit changes by at most

$$\delta_g = \frac{\varepsilon_g}{T}. \tag{44}$$

Let $p$ and $\widehat{p}$ be the softmax policies before and after this perturbation.

**Proposition D.5 (local distribution distortion).** Under Eq. equation 42,

$$\left|\log \frac{\widehat{p}(k)}{p(k)}\right| \le 2\delta_g \qquad \text{for every } k, \tag{45}$$

and consequently

$$D_{\mathrm{KL}}(p \,\|\, \widehat{p}) \le 2\delta_g. \tag{46}$$

*Proof.* Let the exact and perturbed logits be $z_k$ and $z_k + e_k$ with $|e_k| \le \delta_g$. Their log partition functions satisfy $|\log \widehat{Z} - \log Z| \le \delta_g$, because multiplying every unnormalized probability by a factor between $e^{-\delta_g}$ and $e^{\delta_g}$ changes the partition function by at most the same factor. Hence

$$\log \frac{\widehat{p}(k)}{p(k)} = e_k - (\log \widehat{Z} - \log Z),$$

whose magnitude is at most $2\delta_g$. Averaging the corresponding upper bound under $p$ gives Eq. equation 46. □

This result is intentionally a local statement. It describes the effect of profile-prediction error on the profile-derived action distribution at a fixed state and continuation. It does not account for the change in future state distribution caused by deploying the updated controller.

### D.7 Preference dependence of the profile-derived horizon ranking

For intuition, fix a collected profile under continuation context $\bar{g}$ and vary only the decision-time scalarization weight. Write $g_Q = w$ and $g_C = 1 - w$. For two candidate horizons $k$ and $k'$, define

$$a = \Delta_Q(s,\bar{g},k) - \Delta_Q(s,\bar{g},k'), \qquad b = \Delta_C(s,\bar{g},k) - \Delta_C(s,\bar{g},k').$$

The relative profile score is then

$$D_{k,k'}(w;\bar{g}) = wa + (1-w)b = b + w(a-b). \tag{47}$$

When $a \neq b$, the pairwise indifference point is

$$w^* = \frac{b}{b-a}. \tag{48}$$

A crossing affects the decision only when $w^* \in [0, 1]$. This simple linear structure explains why the same state can support different commitment horizons under different quality–cost preferences. The final PACE policy need not obey a single global threshold, because the learned residual is itself a function of $(s, g, k)$ and because the continuation policy used to define a profile is preference-conditioned.

### D.8 WHY STALE PROFILES CAN BECOME INACCURATE

A counterfactual profile is indexed by the continuation policy under which it was evaluated. Let $\mathbb{P}_{\pi}^{s,k}$ denote the distribution of suffix trajectories produced after forcing horizon $k$ at $s$ and then following continuation $\pi$. Suppose a downstream objective $Y_i$ lies in an interval $[m_i, M_i]$. For two continuations $\pi$ and $\bar{\pi}$,

$$\left|Q_i^{\pi}(s,k) - Q_i^{\bar{\pi}}(s,k)\right| \le (M_i - m_i)\,\mathrm{TV}\left(\mathbb{P}_{\pi}^{s,k}, \mathbb{P}_{\bar{\pi}}^{s,k}\right). \tag{49}$$

This is the standard total-variation bound on the difference of expectations of a bounded random variable.

For the scalarized utility, if normalized quality lies in an interval of width $R_Q$ and normalized positive cost in an interval of width $R_C$, then

$$\left|U_g^{\pi}(s,k) - U_g^{\bar{\pi}}(s,k)\right| \le (g_Q R_Q + g_C R_C)\,\mathrm{TV}\left(\mathbb{P}_{\pi}^{s,k}, \mathbb{P}_{\bar{\pi}}^{s,k}\right). \tag{50}$$

Equation equation 50 gives a precise interpretation of profile freshness: an old profile remains informative when the updated controller induces a similar suffix distribution, but its local target can become stale as the continuation drifts. Periodic refresh does not guarantee small total variation; it simply re-evaluates the target under the new reference continuation rather than treating the old value as policy-invariant.

### D.9 LINEARITY OF OBJECTIVE-SPECIFIC GAE

The main method estimates quality and cost advantages separately and combines them afterward. This is consistent with linear scalarization when the same workflow boundaries, termination masks, and GAE trace parameter are used for both objectives.

Define the scalarized reward and value

$$r_n^g = g_Q r_n^Q - g_C r_n^C, \qquad V_g(s,g) = g_Q V_Q(s,g) - g_C V_C(s,g). \tag{51}$$

With undiscounted episodic returns, the scalarized TD residual is

$$\delta_n^g = r_n^g + (1 - d_n) V_g(s_{n+1}, g) - V_g(s_n, g) \tag{52}$$

$$= g_Q \delta_n^Q - g_C \delta_n^C. \tag{53}$$

Because the GAE recursion is linear,

$$A_n^g = g_Q A_n^Q - g_C A_n^C. \tag{54}$$

**Proposition D.6 (exact linearity of the GAE operator).** If $A_n^Q$ and $A_n^C$ are computed with the same $\lambda$, the same termination masks, and value functions combined as in Eq. equation 51, then the GAE of the scalarized reward equals Eq. equation 54 exactly.

*Proof.* The TD residual is linear in reward and value. Starting from the final workflow boundary and applying the recursion $A_n = \delta_n + \lambda(1 - d_n)A_{n+1}$, induction gives the same linear combination at every earlier boundary. □

This result explains why keeping separate quality and cost value heads does not change the scalarized policy-gradient signal when the component values are exact. In practice, the separation also makes the two estimation problems observable individually and keeps the positive-cost sign convention explicit.

### D.10 Gradient separation and preservation of predictor semantics

The method deliberately routes different losses to different parameter groups. Let $\phi_Q$, $\phi_C$, and $\eta$ denote quality-predictor, cost-predictor, and residual-policy parameters. During profile distillation, predictor outputs are detached, so

$$\nabla_{\phi_Q}\mathcal{L}_{\text{pref}} = 0, \qquad \nabla_{\phi_C}\mathcal{L}_{\text{pref}} = 0. \tag{55}$$

During PPO, the predictors are fixed, giving the analogous equalities for the PPO policy loss. Consequently, preference-weighted policy optimization cannot silently redefine a quantity trained to mean quality retention or cost saving. The residual is the component that absorbs policy adjustments not explained by the two supervised outcome predictors.

This separation is a design property, not an identifiability theorem. The learned predictors are only as accurate as their counterfactual targets and function approximation permit, and the residual can change the final horizon ranking when on-policy evidence supports doing so.

### D.11 Training and deployment complexity

Let $N$ be the number of restored state–preference pairs profiled in a collection round, $K = |\mathcal{K}|$ the number of candidate horizons, and $B$ the number of suffix repetitions per horizon. Profile construction requires

$$NKB \tag{56}$$

forced-horizon suffix rollouts, in addition to source trajectories and later on-policy interaction. Thus counterfactual collection scales linearly with the size of the horizon bank and the number of repeated suffix samples.

At deployment, no counterfactual suffixes are executed. The controller scores the $K$ candidate horizons from the current state and preference, samples or selects one action, and executes only the selected Plan-$k$ segment. Abstractly, the horizon-scoring overhead is $O(K)$ outputs per workflow decision, while the expensive executor interaction remains single-path. This distinction is why counterfactual collection is reported as training expenditure rather than deployment cost.

### D.12 Finite-sample concentration of counterfactual profiles

The profile estimator in Proposition D.2 is an empirical mean. When suffix outcomes are bounded, its sampling error can be characterized without making a parametric assumption about the return distribution. Suppose an objective $Y_i(s, g, k)$ lies almost surely in an interval of width $R_i = M_i - m_i$. Then Hoeffding's inequality gives, for one horizon,

$$\Pr\left(\left|\widehat{Q}_i^{\bar{\pi}_g}(s,k) - Q_i^{\bar{\pi}_g}(s,k)\right| \geq \epsilon\right) \leq 2\exp\left(-\frac{2B\epsilon^2}{R_i^2}\right). \tag{57}$$

Applying a union bound over the $K = |\mathcal{K}|$ candidate horizons gives

$$\Pr\left(\max_{k\in\mathcal{K}}\left|\widehat{Q}_i(s,k) - Q_i(s,k)\right| \geq \epsilon\right) \leq 2K\exp\left(-\frac{2B\epsilon^2}{R_i^2}\right). \tag{58}$$

Consequently, with probability at least $1-\delta$, it is sufficient that

$$B \geq \frac{R_i^2}{2\epsilon^2}\log\frac{2K}{\delta} \tag{59}$$

to make every horizon value accurate to $\epsilon$ for that state, objective, and continuation.

The relative targets compare a candidate horizon with the ReAct reference. On the event in Eq. equation 58,

$$\max_k\left|\widehat{\Delta}_i(s,g,k) - \Delta_i(s,g,k)\right| \leq 2\epsilon, \tag{60}$$

because both the candidate and reference estimates can deviate by at most $\epsilon$. The bound is conservative because it ignores covariance that may arise when an implementation deliberately pairs branch randomness.

This calculation should not be interpreted as a prescription for the suffix count $B$ in the benchmark. The outcome range, variance, and cost of an additional suffix differ by environment and state. It instead makes explicit why the profile bank trades additional training interaction for more stable local supervision.

### D.13 Robustness of the locally preferred horizon

Let

$$S_g(s,k) = g_Q \Delta_Q(s,g,k) + g_C \Delta_C(s,g,k) \tag{61}$$

be the exact profile-derived relative utility, and let $\widehat{S}_g$ be the same quantity formed from approximate predictors. Suppose $|\widehat{S}_g(s,k) - S_g(s,k)| \le \varepsilon_g$ for every candidate horizon. Let $k^*$ be the unique maximizer of $S_g$ and define its local margin

$$m(s,g) = S_g(s,k^*) - \max_{k \neq k^*} S_g(s,k). \tag{62}$$

**Proposition D.7 (margin condition for horizon recovery).** If $m(s,g) > 2\varepsilon_g$, then $k^*$ is also the unique maximizer of $\widehat{S}_g$.

*Proof.* For any $k \neq k^*$,

$$\widehat{S}_g(s,k^*) - \widehat{S}_g(s,k) \ge S_g(s,k^*) - S_g(s,k) - 2\varepsilon_g \ge m(s,g) - 2\varepsilon_g > 0.$$

Hence no competing horizon can overtake $k^*$. □

The result explains why close counterfactual choices are intrinsically harder to supervise than large-margin choices. The method therefore uses a soft target rather than converting every noisy profile into a hard best-horizon label.

### D.14 Effect of the learned residual on the profile reference

The final policy adds $r_\eta(s,g,k)$ to the profile-derived logits. This makes the deployed controller more expressive than a fixed quality–cost scalarization of the two predictors. To separate the role of the residual from prediction error, let $p_0$ denote the policy obtained from the predicted quality/cost logits alone and $p_r$ the policy after adding the residual. If

$$\max_k |r_\eta(s,g,k)| \le R, \tag{63}$$

then the same softmax perturbation argument used in Proposition D.5 yields

$$\left|\log \frac{p_r(k)}{p_0(k)}\right| \le 2R, \qquad D_{\mathrm{KL}}(p_0 \| p_r) \le 2R. \tag{64}$$

No bound on $R$ is imposed by PACE; Eq. equation 64 is therefore an interpretation rather than an algorithmic guarantee. It shows, however, that the residual has a clear mathematical role: it parameterizes the departure from the profile-derived policy that is learned from sequential interaction.

### D.15 Bellman equation for the preference-conditioned controller

For a fixed preference $g$, define the scalarized segment reward

$$r_n^g = g_Q r_n^Q - g_C r_n^C. \tag{65}$$

The preference-conditioned controller induces a workflow-level value

$$V^{\pi_g}(s) = \mathbb{E}_{\pi_g}\left[\sum_{m=n}^{N-1} r_m^g \,\middle|\, s_n = s\right]. \tag{66}$$

Its action value satisfies

$$Q^{\pi_g}(s,k) = \mathbb{E}\left[r_n^g + (1-d_n)V^{\pi_g}(s_{n+1}) \mid s_n = s, k_n = k\right], \tag{67}$$

where the expectation integrates over the random segment duration, executor outputs, and environment observations. The policy value is

$$V^{\pi_g}(s) = \sum_{k \in \mathcal{K}} \pi_g(k \mid s) Q^{\pi_g}(s,k). \tag{68}$$

The restored-state profile in the method is a sampled estimate of an action value of the form in Eq. equation 67, except that the continuation is deliberately frozen to $\bar{\pi}_g$ so alternative current horizons can be compared under a common downstream policy.

### D.16 A DECOMPOSITION OF TRAINING AND DEPLOYMENT EXPENDITURE

The distinction between training-time branching and deployment-time single-path execution can be written explicitly. For one outer round, let $N_s$ be the number of source trajectories used to obtain candidate workflow states, $N_p$ the number of profiled state–preference pairs, $K = |\mathcal{K}|$, and $B$ the suffix repetitions per horizon. Abstractly,

$$C_{\text{train}} = C_{\text{source}} + \sum_{j=1}^{N_p} \sum_{k=1}^{K} \sum_{b=1}^{B} C_{\text{suffix}}^{(j,k,b)} + C_{\text{fit}} + C_{\text{PPO}}. \tag{69}$$

The suffix term includes the forced segment, continuation, and any restoration or reconstruction expenditure charged by the training harness.

For a deployed episode with workflow decisions $n = 0, \ldots, N-1$,

$$C_{\text{deploy}} = \sum_{n=0}^{N-1} \left( C_{\text{controller}}(s_n, g) + C_{\text{executor}}(s_n, k_n) \right). \tag{70}$$

There is no factor of $KB$ in Eq. equation 70, because only the selected horizon is executed. This is the accounting reason for reporting profile collection as training expenditure and the predictor forward passes as deployment expenditure.

### D.17 LIMITS OF THE PRECEDING THEORY

The preceding statements deliberately stop short of a global convergence or Pareto-optimality theorem. Three limitations are important.

First, profile values are local action values under a frozen continuation. A profile-target improvement does not by itself imply end-to-end improvement after the continuation policy changes. Second, the residual is trained from sequential interaction and can alter the profile-derived ordering; the final policy is therefore more expressive than the linear quality–cost score used to initialize it. Third, the theoretical error bounds above are conditional on a fixed state and continuation and do not control distribution shift across visited workflow states. These limitations are precisely why the algorithm combines local counterfactual supervision with on-policy refinement and periodic profile refresh instead of treating any single module as sufficient.

## E DETAILED LEARNING AND EXECUTION PROTOCOL

Unless otherwise stated, cost quantities in this section use the same fixed $C_0$ normalization as the main objective; raw deployment cost is reported separately in the experiments.

This section expands the three components of PACE without changing the method specified in the main text: (i) counterfactual commitment profiles, (ii) a certainty-structured commitment policy, and (iii) on-policy workflow optimization. The executor is frozen throughout. Learning is confined to the quality/cost profile estimators, the policy residual, and the two value heads.

### E.1 WORKFLOW BOUNDARIES AND EXECUTION COMMITMENT

The controller acts only at workflow boundaries. Let $s_n$ denote the observable execution history available at boundary $n$, including the task instruction and all observations accumulated so far. A deployment preference is denoted by

$$g = (g_Q, g_C), \qquad g_Q, g_C \geq 0, \qquad g_Q + g_C = 1.$$

At the boundary, the controller selects a commitment horizon

$$k_n \in \mathcal{K}, \qquad \mathcal{K} = \{1, 2, 4, 8\}$$

in the current experiments. The policy is

$$k_n \sim \pi_\theta(\cdot \mid s_n, g).$$

The selected horizon controls how long the executor follows the current high-level plan before the controller is invoked again. The case $k = 1$ corresponds to ReAct-style execution. For $k > 1$, the frozen executor follows a multi-step plan for up to $k$ primitive steps. Intermediate observations remain available for argument binding and execution. They do not, by themselves, trigger a new high-level planning decision before the current commitment ends. A segment may terminate before the requested horizon when the task finishes, the current plan is exhausted, or the remaining plan becomes unexecutable under the shared stopping rules. Hence, a requested Plan-$k$ segment need not realize exactly $k$ primitive actions.

After the segment returns control, the next observable workflow state is $s_{n+1}$. The resulting controller trajectory is therefore

$$s_0 \xrightarrow{k_0} s_1 \xrightarrow{k_1} s_2 \xrightarrow{k_2} \cdots,$$

where each high-level transition can contain a different number of primitive executor steps.

### E.2 Reference controller and restored-state collection

Counterfactual profile collection is performed in outer rounds. At the beginning of a collection round, the complete current controller is frozen and stored as a reference continuation $\bar{\pi}$. A collected profile is always associated with the preference $g$ and the frozen continuation $\bar{\pi}_g$ under which its suffixes were evaluated.

For a sampled state–preference pair $(s, g)$, the training harness restores the same execution snapshot for every candidate $k \in \mathcal{K}$. The restored snapshot contains the mutable state needed to reproduce the same workflow history. A single branch proceeds as follows:

1. restore the common execution snapshot associated with $s$;
2. force the next commitment decision to be $k$;
3. execute the corresponding Plan-$k$ segment with the frozen executor;
4. when the forced segment returns control, resume the same frozen continuation policy $\bar{\pi}_g$;
5. continue until task termination and record terminal quality together with the positive remaining inference cost.

All candidate horizons therefore share the same prefix and reference continuation, while their subsequent observations and states may differ. This construction compares the downstream consequence of changing the current commitment decision while holding the continuation policy fixed.

With $B$ suffix rollouts for each horizon, define

$$\widehat{Q}_i^{\bar{\pi}_g}(s,k) = \frac{1}{B}\sum_{b=1}^{B} Y_i^{(b)}(s,g,k), \qquad i \in \{Q, C\}, \tag{71}$$

where $Y_Q^{(b)}$ is terminal task quality and $Y_C^{(b)}$ is positive cost incurred from the restored state onward. The cost includes the forced segment and its continuation. Cost already incurred before the restored state is common to all branches and is excluded from the profile comparison.

The collection

$$\mathcal{P}_g(s) = \left\{ \left( \widehat{Q}_Q^{\bar{\pi}_g}(s,k), \widehat{Q}_C^{\bar{\pi}_g}(s,k) \right) : k \in \mathcal{K} \right\} \tag{72}$$

is the counterfactual commitment profile for $(s, g)$ under the frozen continuation $\bar{\pi}_g$.

### E.3 Preference-weighted profile target

For the collection preference, the estimated utility of a candidate horizon is

$$\widehat{U}_g(s,k) = g_Q \widehat{Q}_Q^{\bar{\pi}_g}(s,k) - g_C \widehat{Q}_C^{\bar{\pi}_g}(s,k). \tag{73}$$

We convert these profile utilities into the soft target

$$\rho_g(k \mid s) = \frac{\exp(\widehat{U}_g(s,k)/T)}{\sum_{k' \in \mathcal{K}} \exp(\widehat{U}_g(s,k')/T)}, \tag{74}$$

where $T > 0$ controls target sharpness. The temperature smooths the policy target and is not interpreted as statistical confidence in the suffix estimates.

Because changing $g$ can change both outcome weighting and the continuation policy, a profile collected under $(g, \bar{\pi}_g)$ is not relabeled as though it had been collected under a different preference. Profiles are replaced when a new reference controller is frozen.

### E.4 HORIZON-SPECIFIC RELATIVE OUTCOMES

PACE uses $k_1 = 1$ as the ReAct reference. For any candidate horizon $k$, define signed relative targets

$$\widehat{\Delta}_Q(s, g, k) = \widehat{Q}_Q^{\bar{\pi}_g}(s, k) - \widehat{Q}_Q^{\bar{\pi}_g}(s, k_1), \tag{75}$$

$$\widehat{\Delta}_C(s, g, k) = \widehat{Q}_C^{\bar{\pi}_g}(s, k_1) - \widehat{Q}_C^{\bar{\pi}_g}(s, k). \tag{76}$$

Positive $\widehat{\Delta}_Q$ indicates that the longer commitment improves expected quality relative to ReAct. Positive $\widehat{\Delta}_C$ indicates a cost saving. The signed targets retain both improvements and degradations and remain horizon-specific.

Two estimators predict these quantities:

$$c_{\phi_Q}(s, g, k) \approx \widehat{\Delta}_Q(s, g, k), \qquad d_{\phi_C}(s, g, k) \approx \widehat{\Delta}_C(s, g, k).$$

The quality predictor is the operational quantity referred to as horizon-specific planning certainty in the main text: it measures expected quality retention of a candidate commitment relative to ReAct under the reference continuation. The cost predictor estimates the corresponding computational saving.

Their regression objectives are

$$\mathcal{L}_Q = \mathbb{E}_{(s,g)}\left[\frac{1}{M-1}\sum_{j=2}^{M}\Big(c_{\phi_Q}(s, g, k_j) - \widehat{\Delta}_Q(s, g, k_j)\Big)^2\right], \tag{77}$$

$$\mathcal{L}_C = \mathbb{E}_{(s,g)}\left[\frac{1}{M-1}\sum_{j=2}^{M}\Big(d_{\phi_C}(s, g, k_j) - \widehat{\Delta}_C(s, g, k_j)\Big)^2\right]. \tag{78}$$

The two predictors have distinct trainable parameters and receive only their corresponding outcome supervision.

### E.5 PREFERENCE-CONDITIONED POLICY AND PROFILE DISTILLATION

The predicted quality retention and cost saving are combined only after the requested preference is supplied. PACE defines

$$z_\theta(s, g, k) = \frac{g_Q c_{\phi_Q}(s, g, k) + g_C d_{\phi_C}(s, g, k)}{T} + r_\eta(s, g, k), \tag{79}$$

where $r_\eta$ is a trainable policy residual. The commitment distribution is

$$\pi_\theta(k \mid s, g) = \frac{\exp z_\theta(s, g, k)}{\sum_{k' \in \mathcal{K}} \exp z_\theta(s, g, k')}. \tag{80}$$

The residual lets sequential policy optimization adjust commitment selection without changing the supervised meaning of the outcome predictors. The counterfactual target is distilled through

$$\mathcal{L}_{\text{pref}} = \mathbb{E}_{(s,g)}\left[D_{\text{KL}}\left(\rho_g(\cdot \mid s) \,\|\, \pi_\theta(\cdot \mid s, g)\right)\right]. \tag{81}$$

Predictor outputs are detached in this update. Thus $\mathcal{L}_{\text{pref}}$ updates the residual parameters $\eta$, while $\phi_Q$ and $\phi_C$ retain the meanings imposed by $\mathcal{L}_Q$ and $\mathcal{L}_C$.

Table 9: Parameter-update partition. The executor is frozen throughout.

| Signal / stage | $\phi_Q$ | $\phi_C$ | $\eta$ | $\nu_Q, \nu_C$ | Executor |
|---|---|---|---|---|---|
| Quality-profile regression $\mathcal{L}_Q$ | ✓ | | | | |
| Cost-profile regression $\mathcal{L}_C$ | | ✓ | | | |
| Profile distillation $\mathcal{L}_{\text{pref}}$ | | | ✓ | | |
| PPO policy update | | | ✓ | | |
| Value fitting | | | | ✓ | |
| Executor parameters | | | | | frozen |

### E.6 On-policy workflow optimization

After profile fitting and distillation, fresh trajectories are collected using the updated controller. The preference is sampled at episode start and remains fixed until termination. At workflow boundary $n$,

$$k_n \sim \pi_\theta(\cdot \mid s_n, g).$$

The chosen Plan-$k_n$ segment is executed by the frozen executor, producing the next workflow state $s_{n+1}$ and the cost accumulated during that segment.

Separate value heads estimate quality and positive cost-to-go:

$$V_{\nu,Q}(s, g), \qquad V_{\nu,C}(s, g).$$

The episodic objectives are undiscounted in the current formulation. For $i \in \{Q, C\}$, the workflow-boundary TD residual is

$$\delta_n^i = r_n^i + (1 - d_n)V_{\nu_{\text{old}},i}(s_{n+1}, g) - V_{\nu_{\text{old}},i}(s_n, g), \tag{82}$$

where $d_n$ indicates true task termination. Generalized advantage estimation is applied across workflow boundaries:

$$A_n^i = \delta_n^i + \lambda(1 - d_n)A_{n+1}^i. \tag{83}$$

Quality and cost advantages are combined only after they are estimated:

$$A_n^g = g_Q A_n^Q - g_C A_n^C. \tag{84}$$

During the PPO phase, the outcome predictors and executor remain frozen. Let

$$w_n(\eta) = \frac{\pi_{\phi,\eta}(k_n \mid s_n, g)}{\pi_{\phi,\eta_{\text{old}}}(k_n \mid s_n, g)}, \qquad \phi = (\phi_Q, \phi_C).$$

The policy residual is optimized with the clipped surrogate used in the main text:

$$\begin{aligned}\mathcal{L}_{\text{policy}} = & -\mathbb{E}_\tau \left[ \sum_{n \in \tau} \min \left\{ w_n A_n^g, \text{clip}(w_n, 1 - \epsilon, 1 + \epsilon) A_n^g \right\} \right] \\ & + \lambda_{\text{pref}} \mathcal{L}_{\text{pref}}.\end{aligned} \tag{85}$$

Value heads are fitted separately to their detached return targets. One controller-level policy ratio is used per commitment decision, independent of the number of primitive executor actions contained in the selected segment.

### E.7 Parameter-update partition

Table 9 makes explicit which training signal updates each learned component.

### E.8 Outer-loop training schedule

A complete outer iteration follows the order below.

1. **Freeze the reference controller.** Copy the current complete controller to obtain $\bar{\pi}$.

2. **Collect restored-state profiles.** For selected $(s, g)$ pairs, evaluate all candidate horizons and continue every branch with the same $\bar{\pi}_g$.
3. **Fit the two outcome predictors.** Update $\phi_Q$ and $\phi_C$ using the signed relative outcomes.
4. **Distill counterfactual policy targets.** Update $\eta$ against $\rho_g$ while detaching the predictor outputs.
5. **Collect fresh trajectories.** Roll out the newly updated controller. Old trajectories are not reused as fresh PPO data after the policy changes.
6. **Perform on-policy refinement.** Fit $V_Q, V_C$ and update $\eta$ with the preference-weighted PPO objective while keeping the executor and outcome predictors frozen.
7. **Refresh the next collection round.** Freeze the updated controller as the next reference and replace the previous profile targets.

This ordering keeps every profile tied to the continuation policy under which its outcomes were evaluated.

### E.9 Algorithmic view of one training round

For reproducibility, the following procedural description restates one outer training round using only the modules defined in the main text.

**Stage 1: freeze and collect.** Given current parameters $(\phi_Q, \phi_C, \eta)$, instantiate a frozen reference copy $\bar{\pi}$. Roll out the current controller on training tasks and select workflow boundaries to profile. For every selected $(s, g)$ and every $k \in \mathcal{K}$, restore $s$, force the next horizon to $k$, and then continue with $\bar{\pi}_g$ until termination. Store the individual suffix outcomes as well as the profile means.

**Stage 2: construct supervised targets.** For each stored profile, compute $\rho_g(k \mid s)$ from the scalarized suffix outcomes and compute the signed relative targets $\widehat{\Delta}_Q$ and $\widehat{\Delta}_C$ against $k_1 = 1$. No hard best-horizon label is required.

**Stage 3: fit outcome predictors.** Update $\phi_Q$ only with the quality regression loss and $\phi_C$ only with the cost regression loss. Their outputs retain the semantics of relative quality retention and cost saving.

**Stage 4: distill the local policy reference.** With the two predictor outputs detached, update $\eta$ to reduce $D_{\mathrm{KL}}(\rho_g \| \pi_\theta)$. This stage teaches the residual to reproduce the measured local profile without allowing the preference-weighted policy loss to redefine either predictor.

**Stage 5: collect fresh on-policy trajectories.** After the profile-fitting updates, execute the updated complete controller to obtain trajectories for PPO. Reusing trajectories generated before the policy update would no longer be on policy.

**Stage 6: fit values and refine the residual.** Estimate $A^Q$ and $A^C$ at workflow boundaries, form $A^g = g_Q A^Q - g_C A^C$, fit the two value heads to their detached targets, and update $\eta$ with the PPO objective plus the profile-distillation regularizer. The executor and outcome predictors remain fixed during this stage.

**Stage 7: advance the reference.** Carry the learned residual and predictor parameters forward. At the beginning of the next outer round, freeze the updated complete controller and recollect profiles under that new continuation. Old profiles are not relabeled as though they were outcomes under the new reference.

### E.10 Minimal data objects required by the training loop

The method can be implemented using three conceptually distinct data objects. This separation is useful for avoiding accidental leakage between local counterfactual supervision and on-policy updates.

Table 10: Conceptual data objects used by PACE. The table specifies information already required by the method; it does not prescribe a storage format.

| Object | Required contents |
|---|---|
| Restorable snapshot | Observable execution history at a workflow boundary, the task state required to reproduce that boundary, and the associated preference $g$. |
| Profile record | Reference-controller identifier, restored state, preference, candidate horizon, individual suffix quality/cost outcomes, and their aggregate estimates. |
| On-policy workflow transition | $(s_n, g, k_n, r_n^Q, r_n^C, s_{n+1}, d_n)$ together with the behavior-policy probability needed by PPO. |

Profile records are tied to the frozen continuation under which they were collected. On-policy transitions are tied to the behavior policy that generated them. Keeping these identities distinct implements the same freshness requirements stated in the main algorithm.

### E.11 DEPLOYMENT-TIME COMPUTATION

At deployment, state restoration and counterfactual branching are removed. At each workflow boundary, the trained controller:

1. receives the current observable workflow state $s$ and preference $g$;
2. predicts horizon-specific relative quality retention and cost saving;
3. combines these predictions with $g$ and the learned residual;
4. selects one horizon from $\pi_\theta(k \mid s, g)$;
5. executes the selected Plan-$k$ segment with the frozen executor;
6. repeats at the next workflow boundary.

The value heads are used for training and are not required for deployment action selection.

## F STATISTICAL EVALUATION AND UNCERTAINTY CALCULATIONS

This section expands the uncertainty calculations referenced by the benchmark and controlled-study protocols. It introduces no additional performance claims; its purpose is to make the aggregation rules explicit.

### F.1 PAIRED MONTE CARLO ESTIMATION IN THE CONTROLLED SIMULATOR

The controlled simulator reuses the same exogenous realization when comparing reference policies. For environment condition $c$, realization $j$, preference $g$, and policies $\pi$ and $\pi'$, define the paired utility difference

$$D_{c,j,g} = U_g(\pi; \omega_{c,j}) - U_g(\pi'; \omega_{c,j}), \tag{86}$$

where $\omega_{c,j}$ contains the latent transitions and observations of the shared simulator realization. Pairing removes variation that is common to the two policies on the same exogenous path.

The condition-level statistic used for the sampled controlled figure first averages over the nine preferences inside a realization,

$$\overline{D}_{c,j} = \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} D_{c,j,g}, \tag{87}$$

then averages over the $N_c$ independent realizations,

$$\widehat{\mu}_c = \frac{1}{N_c} \sum_{j=1}^{N_c} \overline{D}_{c,j}. \tag{88}$$

With sample standard deviation $s_c$ of the $\overline{D}_{c,j}$ values, the reported normal-approximation interval is

$$\widehat{\mu}_c \pm 1.96 \frac{s_c}{\sqrt{N_c}}. \tag{89}$$

These intervals quantify Monte Carlo path sampling only. They do not include neural-training variability, benchmark task sampling, or uncertainty in model selection.

### F.2 Hierarchical benchmark resampling

The benchmark protocol includes multiple training seeds, tasks, and stochastic executions. These sources of variation are nested rather than exchangeable at a single level. A bootstrap draw should therefore preserve the dependence induced by sharing a trained checkpoint across many tasks and preferences.

A single hierarchical bootstrap replicate can be constructed as follows:

1. sample training-run identifiers with replacement;
2. within each selected run, sample task identifiers with replacement;
3. within each selected task, sample its execution replicates with replacement;
4. retain all requested preferences associated with the resampled task/run block so that cross-preference dependence is preserved;
5. recompute quality, cost, achieved utility, and the complete hypervolume statistic from the resampled data.

Paired method comparisons should use the same resampled task and replicate identifiers whenever those evaluations are available, so the bootstrap targets the difference between methods rather than two independently resampled means.

### F.3 Achieved utility and preference averaging

For an evaluation set $\mathcal{G}$, achieved utility is

$$\mathrm{AU}_{\mathcal{G}} = \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} \left[ g_Q J_Q(\pi_g) - g_C \frac{J_C(\pi_g)}{C_0} \right]. \tag{90}$$

The policy $\pi_g$ is the policy actually requested at preference $g$ from the single selected checkpoint. The evaluator does not search the test-time operating set and substitute a different policy with larger hindsight utility. This distinction makes AU a measure of *preference response* rather than archive quality.

Seen and held-out AU average different preference sets, so their numerical difference is not by itself a generalization gap. Preference transfer should instead be evaluated directly on the held-out requests using the same frozen checkpoint-selection rule.

### F.4 Hypervolume computation

Each requested preference yields a two-objective point

$$y_g = \left( J_Q(\pi_g), -J_C(\pi_g)/C_0 \right), \tag{91}$$

so both coordinates are maximized. We use reference point $(0, -1)$. Hypervolume is computed on the nondominated subset of the nine requested points. Duplicate points contribute no additional area.

### F.5 Matched-quality comparisons

A matched-quality cost comparison requires the target quality and the operating point selection rule to be fixed using development data. Let $q^*$ be a target quality chosen before final test evaluation. For each method, a preference or mixture is selected on validation data to attain $q^*$ as closely as allowed

by the declared rule. The selected setting is then frozen and evaluated on the test set, reporting both its realized test quality and its test cost.

This protocol differs from visually selecting two nearby points after seeing the test frontier. If a method cannot attain the target quality under the predeclared selection rule, the matched-quality entry should be reported as unavailable rather than extrapolated.

# G Reproducibility, Data Freshness, and Reporting Details

This section consolidates reporting conventions already implied by the method and experiment sections.

## G.1 Training-time and deployment-time separation

Counterfactual branching, restored-state suffix evaluation, and the two value heads are training-time mechanisms. Deployment executes one selected horizon at each workflow boundary and does not branch over alternative futures. Outcome predictors and the policy residual remain part of the inference-time controller and their inference expenditure is included in deployment cost.

## G.2 State and outcome leakage

The controller is restricted to information observable at deployment. The restoration mechanism is used to reproduce training states and does not expose future observations, latent simulator modes, benchmark evaluator labels, or outcomes of unexecuted branches to the deployed controller. In the controlled simulator, the latent mode and intermediate correctness labels are available to the simulator/evaluator but not to the controller.

## G.3 Freshness of profile supervision

A counterfactual profile is labeled by the preference and reference continuation under which it was collected. When the controller changes, the next outer iteration freezes the updated controller and recollects profile supervision. The main method therefore does not interpret a suffix outcome collected under one continuation as though it were a value under another continuation.

## G.4 Evaluation summaries

For every benchmark method, the final report should retain raw task quality and raw deployment cost in addition to AU and hypervolume. Conditional methods are evaluated at every requested preference using the same frozen checkpoint. Fixed-horizon baselines have preference-invariant behavior, so the same raw quality–cost point can be reweighted under different preferences without rerunning the executor.

For benchmark results, seed-level statistics should be retained. Any uncertainty interval reported for a difference should preserve pairing among methods evaluated on the same tasks and among preference points produced by the same trained checkpoint.

## G.5 Requested horizon, realized duration, and early return

The controller action is the requested commitment horizon $k$, whereas the executor may return control after fewer primitive actions. Let $\tau_n$ denote the realized duration of workflow decision $n$. The execution protocol permits $\tau_n < k_n$ when the task terminates, the generated plan is exhausted, or the remaining plan becomes unexecutable under the shared stopping rule. Therefore reported commitment behavior should distinguish at least

$$k_n \qquad \text{and} \qquad \tau_n.$$

Counting only requested horizons can overstate the amount of execution actually performed without replanning. Conversely, counting only realized duration can hide the controller's intended level of commitment when early termination is caused by successful task completion.

The same distinction applies to cost accounting. The cost attributed to a workflow segment includes controller inference and executor calls actually incurred before control returns; it is not multiplied by the requested horizon when fewer steps are realized. A counterfactual branch similarly records the actual cost of the forced segment plus its continuation rather than assigning a fixed nominal cost to Plan-$k$.

### G.6 Prefix exclusion and suffix attribution in profile collection

All candidate horizons in a counterfactual profile begin from the same restored workflow state. Cost incurred before that state is therefore common to every candidate and cannot be changed by the current commitment decision. Profile values consequently use remaining cost from the restored state onward. If $C_{\mathrm{prefix}}(s)$ denotes already-incurred expenditure and $C_{\mathrm{suffix}}(s,k)$ the random future expenditure after forcing $k$, then the full-episode cost can be written

$$C_{\mathrm{episode}}(s,k) = C_{\mathrm{prefix}}(s) + C_{\mathrm{suffix}}(s,k). \tag{92}$$

For two candidate horizons $k$ and $k'$, their difference is

$$C_{\mathrm{episode}}(s,k) - C_{\mathrm{episode}}(s,k') = C_{\mathrm{suffix}}(s,k) - C_{\mathrm{suffix}}(s,k'). \tag{93}$$

Thus excluding the common prefix preserves the local cost comparison while avoiding repeated accounting of expenditure that the current decision cannot change. Restoration or prefix-reconstruction overhead used by the training harness is still charged to *training expenditure*; it is simply not part of the downstream deployment cost target predicted by the controller.

### G.7 Failures and incomplete episodes

Failed episodes are not dropped from benchmark averages. Their evaluator score contributes to quality and all model inference performed before termination contributes to deployment cost. This convention is important in a quality–cost study because a strategy that terminates cheaply by failing should not appear preferable merely because expensive unsuccessful trajectories were removed. Administrative truncation of a training batch is treated separately from true task termination for value bootstrapping, as specified by the workflow-level GAE equations.

## H Notation Summary

Table 11: Notation used by the commitment controller and the supplementary protocols.

| Symbol | Meaning |
|---|---|
| $s$ | Observable workflow state / execution history available at a workflow boundary. |
| $g = (g_Q, g_C)$ | Deployment preference over quality and cost. |
| $\mathcal{K}$ | Candidate commitment horizons; current experiments use $\{1,2,4,8\}$. |
| $k$ | Commitment horizon selected at the current workflow boundary. |
| $\bar{\pi}_g$ | Frozen reference continuation used for profile collection at preference $g$. |
| $\widehat{Q}_Q, \widehat{Q}_C$ | Estimated downstream quality and positive cost after forcing horizon $k$. |
| $\rho_g(k \mid s)$ | Counterfactual soft policy target derived from the profile. |
| $\widehat{\Delta}_Q$ | Relative quality change of horizon $k$ versus ReAct. |
| $\widehat{\Delta}_C$ | Relative cost saving of horizon $k$ versus ReAct. |
| $c_{\phi_Q}$ | Horizon-specific quality-retention predictor. |
| $d_{\phi_C}$ | Horizon-specific cost-saving predictor. |
| $r_\eta$ | Trainable residual/correction in the policy logits. |
| $\pi_\theta(k \mid s, g)$ | Preference-conditioned commitment policy. |
| $V_Q, V_C$ | Quality and positive cost-to-go value heads used in on-policy training. |
| $A^Q, A^C, A^g$ | Quality, cost, and preference-weighted advantages. |
| $C_0$ | Environment-specific fixed cost normalization scale. |