PRA: CERTIFYING SKILLS BEFORE THEY BECOME TOOL-AGENT ACTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Tool agents repeat short action sequences, and every step of such a sequence costs a policy call. Recurring behaviour, which we call a skill, is therefore an opportunity to reduce calls. A large body of work asks what is worth constructing from demonstrations. However, whether a deployed policy should then launch that skill with one decision is a different question. Existing criteria cannot answer it, because recurrence and executability describe the demonstrator and the environment, not the policy that will pay. On ALFWorld, catalogues selected for recurrence or executability realise their declared effect more than 90% of the time yet cut task success from 0.861 to as low as 0.493. **To address this, we propose policy-relative admission (PRA)**. PRA separates skill construction from **action promotion** and settles promotion as a decision about one frozen policy. Candidates are lifted into operator sequences that a deterministic binder executes blind, so runnability is measured rather than assumed. Certification then asks two questions on held-out games. One asks whether committed execution realises the declared effect, and the other whether exposing the action lowers decisions without losing task success beyond a non-inferiority margin. Experimental results show that the same behaviour is an action for one policy and not for another, since the admitted set moves with policy capacity and family and with how the policy was trained. Both certificates reject distinct failure modes, and neither subsumes the other. Under a behaviour-cloned policy and a released RL-trained agent alike, PRA is the only method that cuts deployment-time decisions at non-inferior success in both environments. On WebShop it removes 54% of the cloned policy's decisions and raises its official score from 0.248 to 0.395. Our work turns promotion from a property assumed at construction time into a decision a deployment can audit.

1 INTRODUCTION

A tool agent (Yao et al., 2023; Schick et al., 2023) repeats short action sequences across episodes. Walking to a cabinet before opening it recurs from task to task, and so does selecting a product option before purchasing. Each step still costs a full policy call, because action units come from the interface and not from the task. Recurring behaviour is therefore an opportunity to reduce the decisions a policy makes, which is why temporally extended actions have been studied since options (Sutton et al., 1999; Dietterich, 2000).

Two lines of work supply such units for language agents. Action tokenizers segment demonstrated action sequences into reusable chunks (Yunis et al., 2024). Macro actions group tokens for policy optimization (Chai et al., 2025), and offline hierarchies learn plans that a low-level controller follows (Hu et al., 2025). Skill libraries instead accumulate and retrieve behaviours as an agent works (Wang et al., 2024; 2025; Xia et al., 2026; Wang et al., 2026; Yu et al., 2026). Throughout, a *skill* is a candidate reusable behavioural segment. A skill *becomes an action* when a frozen policy may launch the whole segment with one decision. Both lines select skills by recurrence, replayability or retrieval utility, and typically re-optimise the policy afterwards. Selection therefore closes before anyone asks what the result costs a particular deployed agent. What neither line tests, however, is the deployment question:

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.

CENTRAL QUESTION

When should a constructed behaviour be promoted into a single action for a particular deployed policy?

A reusable skill is not yet an action. *Skill construction* discovers a reusable segment, whereas *action promotion* decides whether a particular deployed policy may invoke it as one action. Promotion, unlike construction, must be judged against that policy. Figure 1(a) shows the example that carries the rest of the paper. In ALFWorld (Shridhar et al., 2021) the sequence "go to a drawer and open it" recurs in almost every kitchen episode and usually executes as declared. Every construction criterion admits it. Committed blindly, however, this candidate opens every drawer the policy walks up to whether or not the policy needed it open. For Qwen2.5-1.5B (Yang et al., 2024) the candidate saves about five decisions per episode and clears all three admission inequalities on two of three seeds. For Qwen2.5-3B it saves as many decisions, yet on every seed falls below the effect threshold or the success non-inferiority margin. Conversely, a catalogue of skills can realise its declared effect over 90% of the time and still cost 0.368 success (Section 5.3). Frequency measures the demonstrator and executability measures the environment, but **neither measures the policy that would call the result**.

To address this, we propose **policy-relative admission (PRA)**[1]. PRA is a decision procedure for promotion rather than another construction score. Promotion changes the action space a frozen policy decides in, so PRA tests every candidate against that policy and never re-optimises it. Candidates become *executable action abstractions* run by a deterministic binder. PRA's binder never consults the environment's admissible-action oracle, so runnability becomes a measured property. Certification then estimates two complementary criteria on games disjoint from policy training, one for the declared effect and one for the paired decision saving and success difference against the same policy. Finally, we deploy the frozen catalogue behind a typed guard, with the primitive policy as fallback.

Across two public benchmarks and four frozen policies, **the same behaviour is an action for one policy and not for another**. Cloned Qwen policies of different capacity share eight of the ten actions they admit between them, whereas a released RL-trained agent shares two of fifteen with the smaller one. Each certificate also rejects a failure the other misses, since rules that keep executability alone realise their declared effect more than 0.90 of the time and still cost 0.368 success. Certified catalogues meanwhile keep the task at a one-off certification cost. On ALFWorld they remove 1.60 decisions per episode at an unchanged success of 0.861. On WebShop (Yao et al., 2022) they cut the cloned policy's decisions from 22.11 to 10.13 while its official score rises from 0.248 to 0.395. Passed through PRA with every threshold unchanged, SkillRL's released agents also save decisions at non-inferior success in both environments.

Our work separates skill construction from action promotion and settles promotion as a policy-relative decision. PRA certifies a candidate along two complementary axes. One asks whether committed execution realises the declared effect, and the other whether exposing the action lowers decisions for the frozen policy inside a success non-inferiority bound. Both are estimated without re-training, on a representation whose runnability we measure. What a deployment gains is a promotion rule it can audit before an action is enabled.

2 Related Work

Temporal abstraction and action tokenization. Options formalise a temporally extended action through an initiation set, an intra-option policy and a termination condition (Sutton et al., 1999). MAXQ decomposes value over a task hierarchy (Dietterich, 2000) and Option-Critic differentiates through option policies (Bacon et al., 2017; Smith et al., 2018). Later work states when an abstraction preserves near-optimal behaviour (Abel et al., 2020), while unsupervised discovery searches for distinguishable or structured behaviours (Eysenbach et al., 2019; Machado et al., 2017). A second line derives action units directly from data: Subwords as Skills tokenizes demonstrated action sequences and uses the resulting chunks as a sparse-reward action space (Yunis et al., 2024). MA-RLHF groups language tokens into macro actions (Chai et al., 2025) and GLIDER learns a hierarchy offline (Hu et al., 2025), while related tokenizers act on reasoning traces and robot actions (Zhang et al., 2025; Pertsch et al., 2025; Dong et al., 2026). Across this line the unit is chosen by a construction statistic,

[1]Code: https://anonymous.4open.science/r/policy-relative-admission-2A4A

and its worth is established afterwards by re-optimising a policy on the new action space. What a unit does to a policy that is not re-optimised is therefore left to that later stage. We instead settle that question before the unit is enabled.

Skill learning and reuse in language and tool agents. Language agents accumulate skills as they work (Wang et al., 2024; 2025). SkillRL extracts a hierarchical skill bank and retrieves relevant entries from it. SkillRL's bank evolves alongside a policy trained with those skills present (Xia et al., 2026). SAGE trains skill use with policy gradients (Wang et al., 2026) and PolySkill transfers web skills across interfaces by separating goal from implementation (Yu et al., 2026). Entries enter these libraries by retrieval, by accumulation or by task-level utility measured with the policy in the loop. What rarely gates entry is a test of the specific frozen policy that will call them, so a library can grow while that policy's decisions get worse. Our work, in contrast, makes a realised declared effect and a lower decision count preconditions for entry.

3 Problem Setting

An environment M exposes structured primitive actions. At step t the environment is in state s_t with interaction history h_t, and a language-model policy π_θ emits one primitive action. Each emission costs one *decision*: $|A(s_t)|$ scored sequences where the environment enumerates an admissible set $A(s_t)$ (Huang et al., 2022; Shridhar et al., 2021), and a single generation where it does not. **Decisions rather than environment steps** are the resource an abstraction removes, and we compare compute only within one interface. Effects are read through an *observable projection* ϕ_M, which maps what the environment already returns to the agent onto a symbolic tuple over the entities in play. Admissible commands supply that readout in a text-adventure interface, rendered page controls on the web. Nothing enters ϕ_M that a deployed agent could not see, and neither instantiation uses evaluator state.

An executable action abstraction is a pair $z = (\sigma, g)$, a lifted operator sequence of length L with a task guard. Initiation happens when π_θ selects a primitive action matching its entry under a satisfied guard, and a binder β_z then emits positions $1, \ldots, L-1$ without consulting $A(s)$ or calling π_θ. Each abstraction declares a symbolic effect ψ_z, and a commit is *faithful* when every emitted command takes effect and the realised projection equals ψ_z. Executing z costs one decision instead of L, and its risk is that a blind emission is wrong. Given a proposal pool $\mathcal{Z}$ and a frozen π_θ, admitting z changes both what the agent does and what it pays, which raises two questions:

$$\underbrace{\Pr\big[z \text{ realises } \psi_z \mid \text{commit}\big]}_{\text{effect fidelity: is } z \text{ an action at all?}} \qquad \underbrace{\mathbb{E}\big[D^{\pi} - D^{\pi \cup \{z\}}\big],\ \mathbb{E}\big[S^{\pi \cup \{z\}} - S^{\pi}\big]}_{\text{marginal value: is } z \text{ worth it for this } \pi_\theta?} \tag{1}$$

where D counts decisions and S is official task success, both measured on paired episodes of the same game. Neither quantity implies the other, since a perfectly executable abstraction can still steer the policy into wasted work. Admission therefore seeks a catalogue $\mathcal{Z}_{\text{adm}} \subseteq \mathcal{Z}$ that lowers deployment decision cost without lowering success,

$$\max_{\mathcal{Z}_{\text{adm}}} \mathbb{E}\big[D^{\pi} - D^{\pi \cup \mathcal{Z}_{\text{adm}}}\big] \quad \text{subject to} \quad \mathbb{E}\big[S^{\pi \cup \mathcal{Z}_{\text{adm}}} - S^{\pi}\big] \geq -\varepsilon_S, \tag{2}$$

estimated with one-sided bounds on games disjoint from policy training, with construction and certification charged to the method. Primitives stay available, and $\mathcal{Z}_{\text{adm}} = \varnothing$ is an admissible outcome.

4 Method

Promotion changes one policy's decision process, so the difficulty is to measure that change for a candidate the policy has never used. We do so without re-training the policy and without relying on the construction statistics that proposed the candidate.

Holding the policy fixed isolates the effect of admission from the effect of training, and we ask two questions of every candidate. Does committing to it realise the effect its construction implied? Effect depends on the binder and the environment, and a candidate whose effect holds can still be worthless to this policy. Does exposing it lower this policy's decisions without losing tasks? Value depends on where the policy enters the candidate, and a candidate with task-level value can still leave the policy

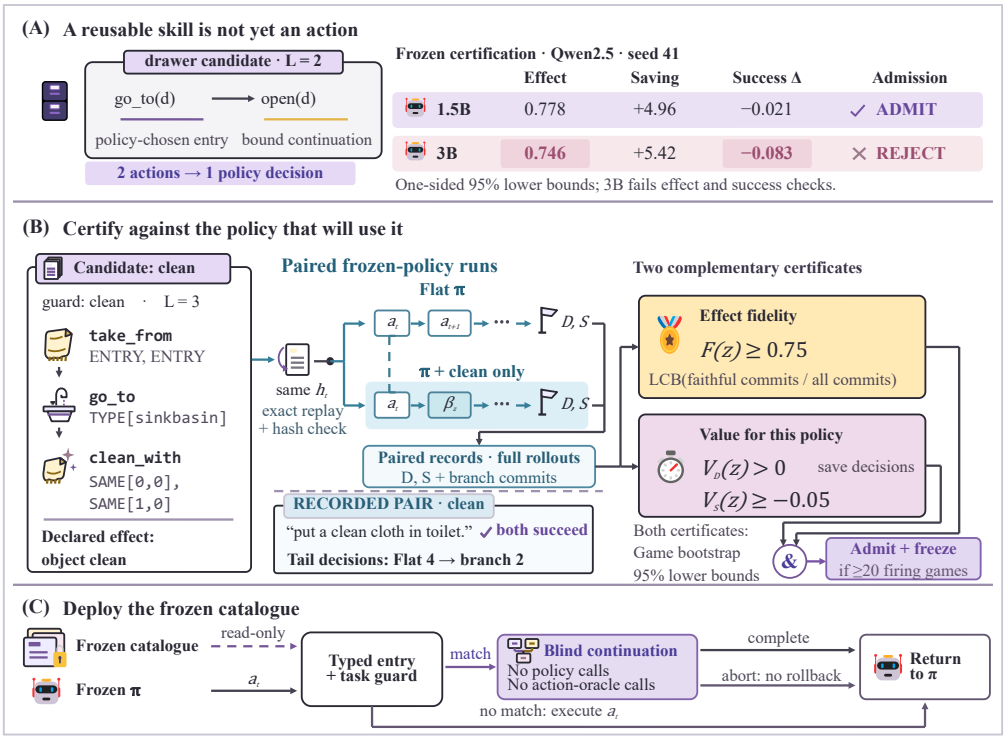


Figure 1: From reusable skill to policy-relative action. (a) A recurring drawer sequence is only a candidate. Qwen2.5-1.5B admits it, while Qwen2.5-3B rejects it because its effect or success bound misses a threshold on every seed. (b) PRA pairs continuations from an exactly replayed history and requires both certificates. A separate recorded example shows the clean skill. (c) A frozen catalogue is read-only, and a typed dispatcher commits to a blind continuation on a match. Bounds are one-sided 95% lower bounds on certification games.

in an unpredictable state on individual commits. Estimating both on the same held-out games makes them complementary rather than independent.

Making those questions measurable takes two representational choices. We lift candidates into parameterised operator sequences, the form macro-operators take in classical planning (Botea et al., 2005). A segment that recurs across episodes recurs up to its arguments. Execution is open-loop, so runnability becomes a measured property of the binder instead of a promise of the admissible-action oracle. An admitted abstraction is an option (Sutton et al., 1999) with a typed predicate as initiation set and a fixed sequence as intra-option policy, and we borrow only that vocabulary.

Figure 1(b) carries a running example through the whole procedure. Cleaning an object in ALFWorld takes three primitive decisions that lift to one abstraction, which declares that the entry object ends up clean at one decision instead of three. Whether that trade is good is what certification must decide.

4.1 Executable action abstractions

Parsing a grounded command gives an operator and its entity arguments. An abstraction is a pair $z = (\sigma, g)$ where $\sigma = \langle (o_0, \rho_0), \ldots, (o_{L-1}, \rho_{L-1}) \rangle$ is a lifted operator sequence and g is a task guard. We restrict arguments to four roles, chosen so that a binder can resolve each of them without consulting the admissible-action oracle. ENTRY at position 0 is bound by the policy's own chosen action, optionally restricted to an entity type τ or to a type the task names. SAME$[j, k]$ co-refers to the entity bound at position j, argument k. TYPE$[\tau]$ denotes the unique entity of type τ mentioned so far in the episode and not yet bound, and HELD denotes the object the agent carries. Together the last two let one abstraction say "go to the target and put down what you hold" for every object and

target. A guard g collects task keywords that must all occur in the task string, so an abstraction can be conditioned on the task phase it belongs to.

Execution is *committed* and *blind*, and we make it so deliberately: a binder that consulted $A(s)$ would inherit the oracle's guarantee and hide exactly the executability failure we want to measure. An abstraction fires when the policy's chosen action a_t takes effect and matches its entry under a satisfied guard and its first continuation step can be bound. Firing, the binder instantiates positions $1, \ldots, L-1$ from its current bindings and steps the environment with each rendered command (Appendix B). An abstraction whose precondition does not hold emits an ineffective command and burns a step, and an unresolvable role or an ineffective command aborts the commit and returns control to π_θ. A commit that runs to completion saves exactly $L-1$ decisions.

4.2 Candidate proposal

On ALFWorld the observable projection ϕ_M reads the admissible commands, which are a faithful readout of the underlying TextWorld state (Côté et al., 2019; Shridhar et al., 2021). For a bound entity e we record the predicate tuple

$$\phi(e, A) = \big(\textsc{at}(e),\ \textsc{open}(e),\ \textsc{closed}(e),\ \textsc{held}(e)\big), \tag{3}$$

read off A by the presence of `go to`, `close`, `open`, and `move` commands naming e. We declare effects as post-conditions over this projection, in the form of STRIPS operators (Fikes & Nilsson, 1971). A declared effect ψ_z is the modal tuple over all bound roles observed when z executes to completion on construction traces. We take the mode because construction traces are noisy, and we record the agreement rate so that an ambiguous declaration stays visible. A commit is *faithful* when it completes with every emitted command effective and the realised post-state equals ψ_z.

Certification should carry the whole burden of deciding what becomes an action. We therefore propose candidates by frequency alone, the criterion the construction line already relies on (Yunis et al., 2024). Frequency is counted over what the frozen policy itself does: it plays construction games, and every contiguous window of length $2 \le L \le L_{\max}$ in one of its successful episodes is lifted under each entry-typing mask and each guard choice. Demonstrations would instead propose the demonstrator's habits, such as a fixed exploration order the policy never follows. On WebShop we propose from human trajectories, and certification still decides. A template surviving $\ge n_{\text{game}}$ distinct games is ranked by expected saving $\text{support} \times (L-1)$, under a per-entry diversity quota that stops one frequent entry filling the pool with near-duplicates certification would pay for one by one. Besides these quotas, our only filter is *constructibility*: the template must blind-execute to completion on at least n_{con} construction games. **Neither effect fidelity nor policy value enters proposal.**

4.3 Two-axis certification

Certification uses N held-out train games that never entered policy training or proposal. Pairing every certification branch with a primitive rollout that shares an exactly replayed history h_t cancels the between-game variance a two-sample comparison would carry. Each game gives a Flat trajectory with success S^π and decisions D^π. For abstraction z we replay h_t up to the first Flat step t that fires z, consuming no decisions and verified by a state hash. Running π_θ to termination with z as the only admitted abstraction then makes the difference between the two branches the whole-episode effect of admitting z (Appendix C).

Let $c_i(z)$ be the commits and $u_i(z)$ the faithful commits on game i, and let $\Delta D_i(z) = D_i^\pi - D_i^z$ and $\Delta S_i(z) = S_i^z - S_i^\pi$ be the paired decision saving and success difference. Write LCB_α for the α-quantile of a statistic under B bootstrap resamples of games. Commits on the same game share its state and policy trajectory, so we resample games and never individual commits. Both certificates are then the lower bounds

$$F(z) = \text{LCB}_\alpha\left(\frac{\sum_i u_i(z)}{\sum_i c_i(z)}\right), \qquad V_D(z) = \text{LCB}_\alpha\Big(\overline{\Delta D(z)}\Big), \qquad V_S(z) = \text{LCB}_\alpha\Big(\overline{\Delta S(z)}\Big), \tag{4}$$

where the effect fidelity F pools commits within a game, and the value bounds V_D and V_S are paired against the same policy and are therefore policy-relative. We test success as non-inferiority against a fixed margin instead of as equality, as a trial tests a new treatment against a standard

one (Piaggio et al., 2006). A bounded loss is the claim a deployment can act on. An abstraction is admitted exactly when it fires on at least $n_{\min}$ certification games, a minimum firing count, and

$$\underbrace{F(z) \geq \tau_F}_{\text{effect certificate}} \quad \text{and} \quad \underbrace{V_D(z) > 0 \text{ and } V_S(z) \geq -\varepsilon_S}_{\text{value certificate}} \tag{5}$$

with τ_F the effect threshold and ε_S the loss of success we are prepared to tolerate rather than a promise of none. Our effect certificate asks whether an action does what it claims. Our value certificate asks whether exposing that action is worth the cost for this particular policy, and **both must hold**. Admission is a *selection rule*, and its claim about the admitted catalogue is tested on evaluation games disjoint from certification. We evaluate Equation 5 at a per-abstraction level α_{sel}, and report a Bonferroni alternative $\alpha_{\text{fam}} = \alpha_0/|\mathcal{Z}_{\text{est}}|$ as a separate and more conservative arm. Here α_0 is the family level and $\mathcal{Z}_{\text{est}}$ the candidates with estimable bounds. Neither level was chosen after seeing evaluation games.

4.4 Deployment

Initiation is a deterministic typed predicate: an abstraction is applicable when its guard holds and the action π_θ just chose matches its entry types, and it fires once that action takes effect and its first continuation step binds (Section 4.1). At deployment, shown in Figure 1(c), the admitted catalogue is frozen and read-only. Every admitted abstraction is enabled wherever it is applicable, and none is added, removed or modified while the agent runs. A deterministic dispatcher fires the longest applicable abstraction, and ties break by guard specificity (Appendix B). Primitive actions stay available, and an abort returns control without undoing the commands already executed. Admission therefore changes the states the policy is asked to decide in, not only how often it is asked.

5 Experiments

Our experiments test the construction-promotion split from three sides: whether certified promotion saves decisions at non-inferior success (Section 5.2), whether each certificate is needed (Section 5.3) and whether promotion depends on the policy (Section 5.4).

5.1 Protocol

ALFWorld 0.4.2 (Shridhar et al., 2021) supplies the official TextWorld interface and `won` signal with 134 valid-unseen games at a 50-step horizon, and WebShop (Yao et al., 2022) supplies its full product corpus and official split by human-goal index. Neither evaluator nor split semantics is modified (Appendix L). Each environment is evaluated under two frozen policies. One is a LoRA-adapted (Hu et al., 2022) Qwen2.5-1.5B policy (Yang et al., 2024) behaviour-cloned on expert trajectories, which we call the cloned policy. Another is the released SkillRL agent (Xia et al., 2026), an RL-trained Qwen2.5-7B checkpoint that we did not train and that tables list as SkillRL-7B. We prompt it exactly as its own environment manager does (Appendix G). Every arm within a panel shares that policy and differs only in which abstractions it may invoke. On ALFWorld PRA proposes from the frozen policy's own successful episodes, and on WebShop from the public human trajectories. Decisions cost differently across panels, so arms are compared only within one. Certification uses train games disjoint from policy training and proposal, and each candidate must fire on at least $n_{\min} = 20$ of them. Before any held-out evaluation we fixed the admission rule and every catalogue, threshold, seed and endpoint (Nosek et al., 2018). Every comparison is paired by episode and pooled over three seeds by a hierarchical bootstrap (Agarwal et al., 2021) with one-sided 95% lower bounds.

Flat invokes no abstraction and is the paired reference for every other arm. *Handcrafted* admits the proposals a domain expert would write on ALFWorld, namely those ending in an object-processing or placing operator. Our *SaS BPE port* builds the byte-pair vocabulary of Subwords-as-Skills (Yunis et al., 2024; Sennrich et al., 2016) over the construction traces with the original code and its main settings, and enables every merged unit without certification. We port only its action-tokenization mechanism and not the reinforcement-learning stage that re-optimizes a policy on the new action space. Our method is *PRA*, and we compare it with recent agents separately. HiPER (Peng et al., 2026) and SkillRL retrain the policy instead of selecting its actions, so we reproduce both under their released protocols in Appendix G and pass SkillRL's own agents through PRA.

Table 1: Main comparison on the held-out splits under two frozen policies per environment, our behaviour-cloned Qwen2.5-1.5B and the released RL-trained SkillRL agent. Arms within a panel share the policy and are paired on the same episodes. Parentheses report one-sided 95% lower bounds on the paired difference. Bold marks the best level in each column, and † marks an arm whose success bound falls below -0.05.

Arm	#Adm.	Success ↑	ΔSucc. (LCB) ↑	ΔDec. (LCB) ↑	Fidelity ↑
Panel A: ALFWorld valid-unseen, Qwen2.5-1.5B, 134 tasks × 3 seeds					
Flat	–	**0.861**	+0.000 (+0.000)	+0.00 (+0.00)	–
Handcrafted†	11.7	0.811	–0.050 (–0.100)	–0.11 (–2.04)	0.600
SaS BPE port†	16.0	0.418	–0.443 (–0.565)	+7.16 (+5.85)	0.195
PRA (ours)	7.0	**0.861**	+0.000 (–0.010)	+1.60 (+1.15)	**0.875**
Panel B: ALFWorld valid-unseen, released SkillRL-7B, 134 tasks × 3 seeds					
Flat	–	**0.769**	+0.000 (+0.000)	+0.00 (+0.00)	–
Handcrafted†	22.0	0.537	–0.231 (–0.274)	+1.47 (+0.23)	0.052
SaS BPE port†	16.0	0.682	–0.087 (–0.117)	+2.03 (+1.02)	0.333
PRA (ours)	4.0	0.761	–0.007 (–0.030)	+0.67 (+0.19)	**0.908**

Arm	#Adm.	Success ↑ (Δ LCB)	Score ↑ (Δ LCB)	ΔDec. (LCB) ↑	Fidelity ↑
Panel C: WebShop official test range, Qwen2.5-1.5B, 200 sessions × 3 seeds					
Flat	–	**0.135** (+0.000)	0.248 (+0.000)	+0.00 (+0.00)	–
SaS BPE port†	16.0	0.017 (–0.145)	0.040 (–0.238)	+14.66 (+13.61)	0.000
PRA (ours)	2.3	**0.135** (–0.035)	**0.395** (+0.035)	+11.98 (+8.56)	**0.955**
Panel D: WebShop official test range, released SkillRL-7B, 200 sessions × 3 seeds					
Flat	–	0.365 (+0.000)	**0.618** (+0.000)	+0.00 (+0.00)	–
SaS BPE port†	16.0	0.073 (–0.343)	0.131 (–0.526)	+3.36 (+2.80)	0.000
PRA (ours)	0.7	**0.372** (–0.005)	**0.618** (–0.010)	+1.34 (+0.51)	**1.000**

5.2 Main results

Only PRA saves decisions at non-inferior success under every policy and in both environments. PRA meets both admission conditions fixed before evaluation in all four panels of Table 1. PRA's success bound never falls below -0.035, while its saving bound stays above zero. Every catalogue selected without testing it on the policy instead loses tasks somewhere. Handcrafted falls outside the margin on both ALFWorld panels. Because Handcrafted selects from the same proposals as PRA, what separates the two is the selection rule and not the candidates. SaS units fire wherever their first action matches, six times per ALFWorld episode against 1.4 for PRA. SaS falls outside the margin on all four panels. Certifying against the deploying policy is what makes promotion safe.

An untested catalogue overrides an RL-trained policy more. Handcrafted costs 0.050 success under the cloned policy but 0.231 under the released agent. For that agent its abstractions fire about eleven times per episode, and only 5% of those commits realise their declared effect. Chosen without testing, a catalogue cannot know which routines an RL-trained policy already performs well.

Savings track the redundancy a policy leaves. On WebShop the cloned policy spends 22 decisions per session, largely browsing on after it has reached a buyable page. PRA removes 54% of them and raises the official score from 0.248 to 0.395, because every seed admits the entry that selects an option and then buys (Appendix M). Needing only 8.9 decisions, the released agent already buys when it should, so PRA removes 15% of them at non-inferior success. On ALFWorld both policies spend about 21 decisions, yet PRA saves 7.5% under the cloned policy and only 3.1% under the released agent. Certification therefore admits fewer actions for a policy that already acts well, and still none that cost tasks.

A released agent gains from certification, whereas its skill bank harms a policy that was not trained with it. Panels B and D pass SkillRL's own checkpoints through PRA with every threshold unchanged. Both admission conditions hold in both environments, and all 911 WebShop commits realise their declared effect. Handing our cloned policy SkillRL's released skill bank through the original retrieval instead lowers valid-unseen success from 0.861 to 0.766, with a lower bound of

Table 2: What each certificate buys on ALFWorld valid-unseen. Every row is a variant of PRA, and w/o both admits on construction statistics alone. A check marks a kept certificate, and the last row applies both at the Bonferroni level.

	certificate						
Variant of PRA	effect	value	#Adm.	Success ↑	ΔSucc. (LCB) ↑	ΔDec. (LCB) ↑	Fidelity ↑
PRA w/o both	×	×	12.3	0.639	–0.221 (–0.264)	–2.10 (–3.49)	0.926
PRA w/o value	✓	×	19.3	0.493	–0.368 (–0.448)	–1.19 (–2.94)	0.945
PRA w/o effect	×	✓	8.0	0.846	–0.015 (–0.032)	+1.82 (+0.99)	0.668
PRA (ours)	✓	✓	7.0	0.861	+0.000 (–0.010)	+1.60 (+1.15)	0.875
PRA (ours), family-wise	✓	✓	5.7	0.863	+0.002 (+0.000)	+0.81 (+0.58)	0.994

Table 3: PRA (ours) run end to end under four frozen policies on the ALFWorld development split, 140 tasks × three seeds. Each row proposes, certifies and deploys a catalogue for one policy and compares it with that policy's own Flat run. Parentheses report one-sided 95% lower bounds on the paired difference.

		Flat	**PRA (ours)** against Flat			
Frozen policy	Training	Success	Success	#Adm.	ΔSucc. (LCB) ↑	ΔDec. (LCB) ↑
Qwen2.5-1.5B	cloned	0.895	0.893	7.0	–0.002 (–0.014)	+2.31 (+1.53)
Qwen2.5-3B	cloned	0.864	0.862	5.7	–0.002 (–0.014)	+0.76 (+0.55)
SmolLM2-1.7B	cloned	0.824	0.821	6.0	–0.002 (–0.021)	+2.83 (+2.06)
SkillRL-7B	RL, released	0.805	0.824	4.0	+0.019 (+0.005)	+1.11 (+0.65)

-0.164 (Table 9 in Appendix D). A skill is valuable to a policy rather than in itself. PRA is therefore complementary to agents that retrain their policy, since any of them can pass through it.

Certification is a one-off cost for each frozen policy. Proposing and certifying a 40-abstraction pool costs 67,470 policy decisions per seed for the cloned ALFWorld policy, and its held-out saving would return them within about 42,000 episodes (Appendix J). Such a charge suits repeated deployment of a frozen policy.

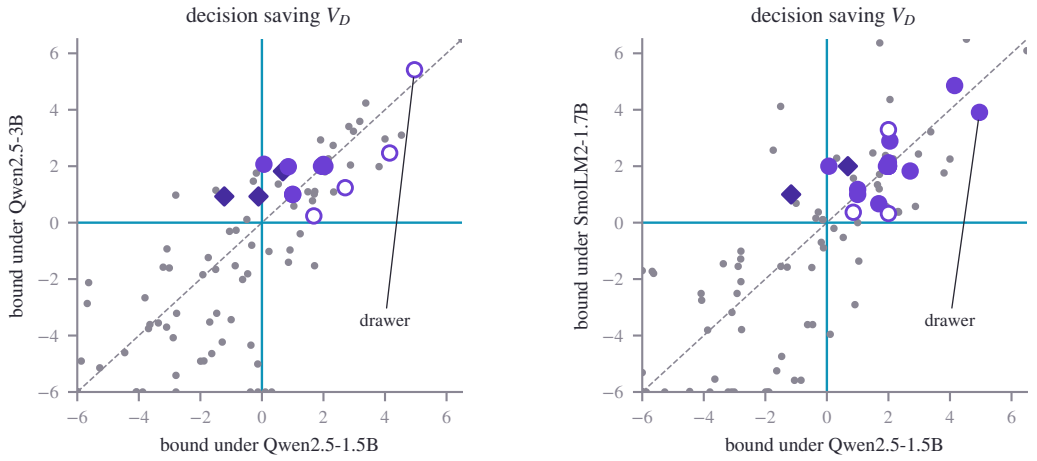
5.3 Certificate ablation

Executability is not value. Table 2 removes each certificate in turn on ALFWorld valid-unseen. Variants without the value certificate admit abstractions that realise their declared effect on 93% to 95% of commits. Yet success falls by 0.221 to 0.368 and decisions rise. A reliable abstraction can still carry the policy somewhere it did not need to go, and the policy then pays in decisions and tasks to recover. Only a test against the policy's own outcomes exposes that cost, which is why the value certificate protects the task.

Value alone admits actions whose meaning the policy cannot rely on. Without the effect certificate, success stays within the margin and 1.82 decisions are saved. Only 0.668 of those commits realise their declared effect, however, and 337 of 1,051 abort against 72 of 576 for PRA. No PRA commit completes without its declared effect. What the effect certificate buys is dependable action semantics for 0.22 decisions per episode. Neither certificate subsumes the other (Table 4), and under every cloned policy the value certificate guards success and the effect certificate fidelity (Table 5).

5.4 Policy-relative promotion

What transfers across policies is the test, not the catalogue. Re-running proposal, certification and deployment end to end for four frozen policies keeps both admission conditions for every one of them (Table 3). What moves with the policy is which candidates pass. Cloned Qwen policies of different capacity share eight of the ten abstractions they admit between them. SmolLM2, with another pretraining corpus and tokenizer, admits all nine of the smaller Qwen's and one more. By contrast the RL-trained agent shares only two of the fifteen abstractions it and the smaller Qwen admit, partly because its own episodes propose different candidates. **Capacity and pretraining move**



(a) Qwen2.5-3B, same family

(b) SmolLM2-1.7B, another family

Figure 2: Decision-saving bounds for the candidates that both policies of a panel proposed, one mark per candidate and seed. Each panel pairs a policy with Qwen2.5-1.5B. Filled circles are admitted under both policies and dots under neither. Hollow circles are admitted under Qwen2.5-1.5B only, and diamonds under the other policy only. Bounds outside the axes sit on their edge, and success bounds behave the same way (Appendix D).

a catalogue at its margin, whereas how a policy was trained moves it wholesale. A borrowed catalogue either forfeits saving or spends success (Table 8 in Appendix D).

Effect belongs to the binder and value to the policy. Between the two Qwen policies effect bounds move by a median of 0.02, because the binder and the environment decide them. Saving bounds move by a median of 1.14 decisions (Figure 2(a)), because the policy decides where it enters a candidate and what it would have done instead. Success bounds move by a median of 0.048, as much as the margin itself (Appendix D). A candidate near the margin can therefore pass for one policy and fail for the other. Section 1's drawer candidate shows the consequence, since two of three seeds admit it for the smaller policy. Every seed rejects it for the larger one, whose effect or success bound falls below its threshold.

On WebShop one entry is an action for one policy and not for the other. Selecting an option and then buying realises its declared effect on every commit under both the cloned policy and the released SkillRL agent. For the cloned policy it raises paired success by 0.086 to 0.098 per seed and removes about 17 decisions, so every seed admits it. For SkillRL it lowers success by 0.061 to 0.156 with lower bounds between -0.122 and -0.234, so no seed admits it.

6 Conclusion

In this work, we argue that a reusable skill is not yet an action. We separate skill construction from action promotion and give the second its own criterion. Promotion is a decision about one deployed policy, so we certify a candidate against that policy rather than against the statistics that proposed it. Policy-relative admission asks whether a candidate's declared effect holds once the action is committed, and whether exposing it pays for the policy that will call it. Evidence from two benchmarks and four frozen policies separates compression from policy value. Catalogues selected for recurrence, executability or expert judgement lose the task while certified ones keep it. Both certificates reject distinct failure modes and neither subsumes the other, so requiring them together buys reliability at a measured price in compression. Admitted sets move with policy capacity and family and with how the policy was trained, so a catalogue certified for one agent is not a catalogue for another. An agent trained elsewhere still gains from the same test. Our evidence covers two text interfaces and policies of up to 7B parameters. Richer binders would widen what can be proposed without changing the certification rule. Treating promotion as a decision that can be tested turns a property assumed at construction time into one a deployment can audit.

AI USE STATEMENT

We conceived the research idea and the central question of this work. We also directed the design of the method and the experiments and supervised every experiment. Under this direction we used generative AI tools, namely Claude through Claude Code and OpenAI Codex, for tasks with required disclosure. Claude and Codex helped refine hypotheses and gave feedback on the methodology and experiment design. Both tools also implemented the methods and the data preparation code to our specification and assisted in interpreting results. We have not used generative AI tools to generate synthetic data. Mathematical proofs and qualitative data analysis are not applicable to this work. Additionally, we used these tools to create figures and to write and edit code. We also used them to draft and edit our text, to search and summarise literature and to format references. We have reviewed all AI-assisted work. An audit script recomputes every headline number from the stored run records, and unit tests cover the implemented components. We checked every cited reference against its primary source. We take responsibility for the final content of this work and for any text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Code for proposal, certification and deployment is available at the anonymous repository linked in Section 1. Every frozen setting appears in Appendix G and Table 13. Section 4.3 and Appendix C specify the certification protocol and its bootstrap.

REFERENCES

David Abel, Nate Umbanhowar, Khimya Khetarpal, Dilip Arumugam, Doina Precup, and Michael L. Littman. Value preserving state-action abstractions. In Silvia Chiappa and Roberto Calandra (eds.), *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pp. 1639–1650. PMLR, 2020. URL http://proceedings.mlr.press/v108/abel20a.html.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C. Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 29304–29320, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/f514cec81cb148559cf475e7426eed5e-Abstract.html.

Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In Satinder Singh and Shaul Markovitch (eds.), *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pp. 1726–1734. AAAI Press, 2017. doi: 10.1609/AAAI.V31I1.10916. URL https://doi.org/10.1609/aaai.v31i1.10916.

Adi Botea, Markus Enzenberger, Martin Müller, and Jonathan Schaeffer. Macro-FF: Improving AI planning with automatically learned macro-operators. *J. Artif. Intell. Res.*, 24:581–621, 2005. doi: 10.1613/JAIR.1696. URL https://doi.org/10.1613/jair.1696.

Yekun Chai, Haoran Sun, Huang Fang, Shuohuan Wang, Yu Sun, and Hua Wu. MA-RLHF: reinforcement learning from human feedback with macro actions. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=WWXjMYZxfH.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew J. Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. TextWorld: A learning environment for text-based games. In Tristan Cazenave, Abdallah Saffidine, and Nathan R. Sturtevant (eds.), *Computer Games - 7th Workshop, CGW 2018, Held in Conjunction with the 27th International Conference on Artificial Intelligence, IJCAI 2018, Stockholm, Sweden, July 13, 2018, Revised Selected Papers*, volume 1017 of *Communications in Computer and Information Science*, pp. 41–75. Springer, 2019. doi: 10.1007/978-3-030-24337-1_3. URL https://doi.org/10.1007/978-3-030-24337-1_3.

Thomas G. Dietterich. Hierarchical reinforcement learning with the MAXQ value function decomposition. *J. Artif. Intell. Res.*, 13:227–303, 2000. doi: 10.1613/JAIR.639. URL https://doi.org/10.1613/jair.639.

Zibin Dong, Yicheng Liu, Shiduo Zhang, Baijun Ye, Yifu Yuan, Fei Ni, Jingjing Gong, Xipeng Qiu, Hang Zhao, Yinchuan Li, and Jianye Hao. ActionCodec: What makes for good action tokenizers. *CoRR*, abs/2602.15397, 2026. doi: 10.48550/ARXIV.2602.15397. URL https://doi.org/10.48550/arXiv.2602.15397.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=SJx63jRqFm.

Richard Fikes and Nils J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artif. Intell.*, 2(3/4):189–208, 1971. doi: 10.1016/0004-3702(71)90010-5. URL https://doi.org/10.1016/0004-3702(71)90010-5.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Zican Hu, Wei Liu, Xiaoye Qu, Xiangyu Yue, Chunlin Chen, Zhi Wang, and Yu Cheng. Divide and conquer: Grounding LLMs as efficient decision-making agents via offline hierarchical reinforcement learning. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/hu25q.html.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9118–9147. PMLR, 2022. URL https://proceedings.mlr.press/v162/huang22a.html.

Jing Lei, Max G'Sell, Alessandro Rinaldo, Ryan J Tibshirani, and Larry Wasserman. Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, 113(523): 1094–1111, 2018.

Marlos C. Machado, Marc G. Bellemare, and Michael H. Bowling. A Laplacian framework for option discovery in reinforcement learning. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2295–2304. PMLR, 2017. URL http://proceedings.mlr.press/v70/machado17a.html.

Brian A Nosek, Charles R Ebersole, Alexander C DeHaven, and David T Mellor. The preregistration revolution. *Proceedings of the National Academy of Sciences*, 115(11):2600–2606, 2018.

Jiangweizhi Peng, Yuanxin Liu, Ruida Zhou, Charles Fleming, Zhaoran Wang, Alfredo García, and Mingyi Hong. HiPER: Hierarchical reinforcement learning with explicit credit assignment for large language model agents. *CoRR*, abs/2602.16165, 2026. doi: 10.48550/ARXIV.2602.16165. URL https://doi.org/10.48550/arXiv.2602.16165.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. FAST: efficient action tokenization for vision-language-action models. *CoRR*, abs/2501.09747, 2025. doi: 10.48550/ARXIV.2501.09747. URL https://doi.org/10.48550/arXiv.2501.09747.

Gilda Piaggio, Diana R Elbourne, Douglas G Altman, Stuart J Pocock, Stephen JW Evans, and for the CONSORT Group. Reporting of noninferiority and equivalence randomized trials: an extension of the CONSORT statement. *JAMA*, 295(10):1152–1160, 2006.

Yaniv Romano, Evan Patterson, and Emmanuel J. Candès. Conformalized quantile regression. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 3538–3548, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/5103c3584b063c431bd1268e9b5e76fb-Abstract.html.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In Katrin Erk and Noah A. Smith (eds.), *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://aclanthology.org/P16-1162/.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew J. Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=0IOX0YcCdTn.

Matthew J. A. Smith, Herke van Hoof, and Joelle Pineau. An inference-based policy gradient method for learning options. In Jennifer G. Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4703–4712. PMLR, 2018. URL http://proceedings.mlr.press/v80/smith18a.html.

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

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Trans. Mach. Learn. Res.*, 2024, 2024. URL https://openreview.net/forum?id=ehfRiF0R3a.

Jiongxiao Wang, Qiaojing Yan, Yawei Wang, Yijun Tian, Soumya Smruti Mishra, Zhichao Xu, Megha Gandhi, Panpan Xu, and Lin Lee Cheong. Reinforcement learning for self-improving agent with skill library. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2026, San Diego, California, United States, July 2-7, 2026*, pp. 1529–1550. Association for Computational Linguistics, 2026. doi: 10.18653/V1/2026.ACL-LONG.69. URL https://doi.org/10.18653/v1/2026.acl-long.69.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/wang25bx.html.

Peng Xia, Jianwen Chen, Hanyang Wang, Jiaqi Liu, Kaide Zeng, Yu Wang, Siwei Han, Yiyang Zhou, Xujiang Zhao, Haifeng Chen, Zeyu Zheng, Cihang Xie, and Huaxiu Yao. SkillRL: Evolving agents via recursive skill-augmented reinforcement learning. *CoRR*, abs/2602.08234, 2026. doi: 10.48550/ARXIV.2602.08234. URL https://doi.org/10.48550/arXiv.2602.08234.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *CoRR*, abs/2412.15115, 2024. doi: 10.48550/ARXIV.2412.15115. URL https://doi.org/10.48550/arXiv.2412.15115.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.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 *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Simon Yu, Gang Li, Weiyan Shi, and Peng Qi. PolySkill: Learning generalizable skills through polymorphic abstraction for continual learning. In *International Conference on Learning Representations*, 2026.

David Yunis, Justin Jung, Falcon Z. Dai, and Matthew R. Walter. Subwords as skills: Tokenization for sparse-reward reinforcement learning. In Amir Globerson, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/7d0c7a899224cd178b2e0cbecf39b5a5-Abstract-Conference.html.

Shenao Zhang, Donghan Yu, Yihao Feng, Bowen Jin, Zhaoran Wang, John Peebles, and Zirui Wang. Learning to reason as action abstractions with scalable mid-training RL. *CoRR*, abs/2509.25810, 2025. doi: 10.48550/ARXIV.2509.25810. URL https://doi.org/10.48550/arXiv.2509.25810.

A Abstraction representation

A stored abstraction records its lifted operator sequence, the role of every argument and its task guard. Each record also carries a content-addressed identifier derived from those fields and its declared symbolic post-state with the agreement rate of that mode. Effects outside the predicate tuple of Equation 3 are confirmed by the effective execution of the command that causes them. A cleaning command confirms that its object becomes clean. Construction statistics complete the record by counting how often its entry fired in the proposing episodes, how often blind execution completed and how often the emitted continuation matched the recorded one. Natural-language descriptions are metadata and never define identity. Nothing measured on certification or evaluation games enters the identifier, so an abstraction proposed in one run is the same object in another.

B Binder and dispatcher

At commit time the binder holds the entry action's arguments and the set of entities the episode has mentioned in observations and executed commands. A SAME role resolves to the entity already

bound at the named position. A TYPE role resolves to the unique mentioned entity of that type not yet bound and fails when zero or several exist. A HELD role resolves to the object the last effective take put in hand. An abstraction whose first continuation position cannot be resolved does not fire, which leaves the entry to a shorter abstraction or to the policy. Each resolved position is rendered into a command string by the operator's surface pattern and stepped directly. Because the binder never reads the admissible-command set, an inapplicable abstraction consumes an environment step and its feedback marks the commit ineffective. At dispatch the longest admitted abstraction whose guard holds and whose entry matches fires. Ties break by guard specificity, then by expected saving and then by identifier, so no run-time measurement enters dispatch.

C CERTIFICATION PROTOCOL

Certification games come from the frozen train ordering after the construction prefix, so they never enter policy training or proposal. Each game contributes one Flat rollout. For each abstraction we locate the first Flat step whose action fires it and replay that prefix action by action. Replay consumes environment steps but no policy decisions and is verified against the recorded state hash. We then run the policy to termination with that abstraction as the only admitted one. Each pair's record contains the trigger index, commits, completions and faithful commits. Each record also holds the executed continuation steps and the branch's own success, with paired differences in decisions, tokens and success. A frequently firing abstraction stops consuming paired games once it reaches the frozen cap, and the number of games it was offered is stored so that intention-to-treat denominators remain correct. Thresholds, the dispatcher, and the statistics are fixed before any evaluation game is inspected.

D ADDITIONAL RESULTS

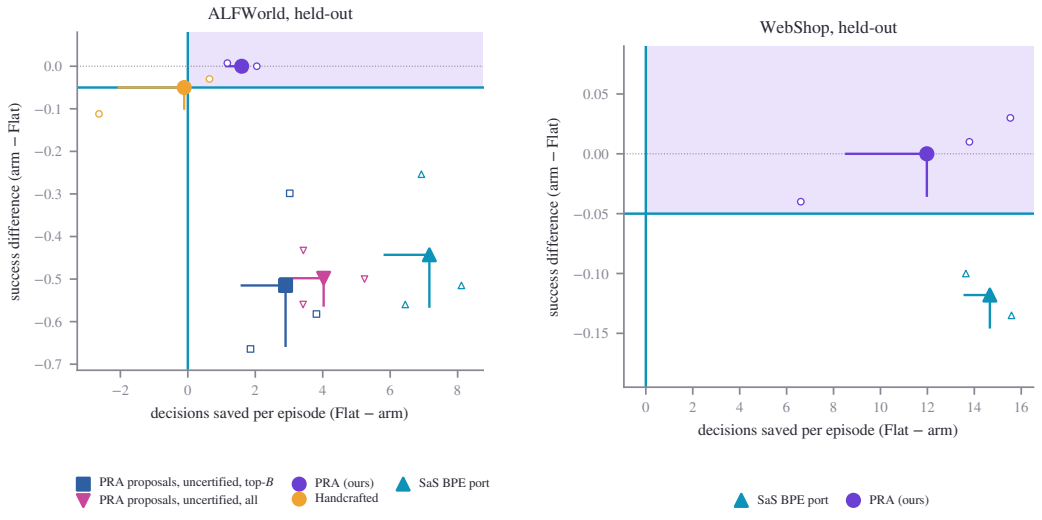


Figure 3: Held-out arms under the cloned policies, placed by their paired decision saving and success difference. ALFWorld is on the left and WebShop on the right. Large marks are pooled means, hollow marks the three seeds, and the segments run to one-sided 95% lower bounds. A tinted band marks the success margin, so an arm inside it keeps the task.

Held-out arms. Figure 3 places the held-out arms under the cloned policies by paired saving and success difference. Only PRA sits inside the success band with a positive saving on both benchmarks. Uncertified catalogues and the SaS BPE port save more decisions but fall far below the band. Handcrafted selects from the same proposal pool as PRA by an expert rule, so the pair isolates the selection rule. On the held-out split PRA saves 1.60 decisions at unchanged success. Handcrafted saves none and falls 0.050 below Flat, with a lower bound of -0.100 beyond the margin. An expert

rule of this kind enables every placing and processing abstraction, and only 0.600 of those commits realise their declared effect under the cloned policy.

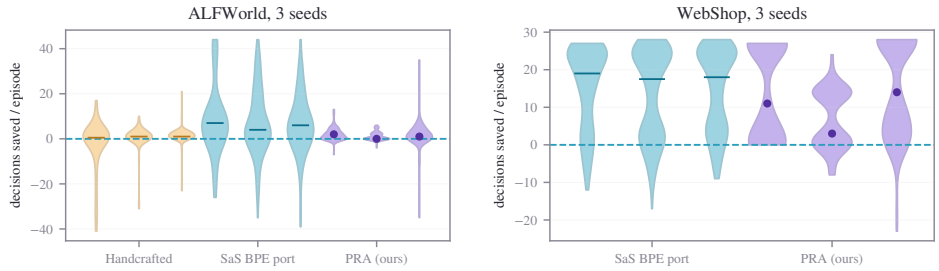


Figure 4: Per-episode dispersion behind Panels A and C of Table 1: one violin per seed per arm, decisions saved against the primitive policy on the same episode. On ALFWorld, units of the SaS BPE port save more on average but with a dispersion several times wider.

Figure 4 shows the per-episode spread behind these means. Units of the SaS BPE port either save a long stretch or add decisions when they fail, so their savings spread several times wider than PRA's. PRA's savings stay close to their mean on both benchmarks.

Table 4: Candidates proposed under the first seed, 40 in all, split by the effect and saving bounds averaged over seeds. Every region is populated, so neither bound orders the other. Nine of the 10 candidates that clear both bounds also clear the success margin.

	Saving bound > 0	Saving bound ≤ 0
Effect bound $\geq \tau_F$	10	8
Effect bound $< \tau_F$	8	14

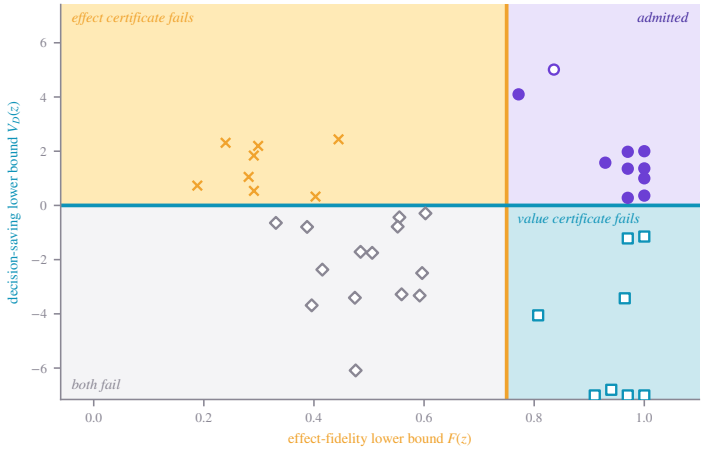


Figure 5: Why promotion is not a one-dimensional ranking, plotting the counts of Table 4. Each mark is one of the 40 candidates, placed by its effect-fidelity and decision-saving lower bounds. Filled circles are admitted and a hollow mark fails success non-inferiority. Squares fail the saving bound, crosses the effect bound and diamonds both. Saving bounds below the lower axis limit are drawn on that limit, and coinciding marks are shifted left.

Both certificates. Table 4 splits the 40 candidates of the first seed by the effect and saving bounds, and Figure 5 places each of them. All four regions are populated. Eight candidates execute faithfully yet save nothing, and eight save decisions without a faithful effect. Ranking by either bound alone would therefore admit one of these failure types. Of the 10 candidates that clear both bounds, nine also clear the success margin and one has a success bound of -0.24.

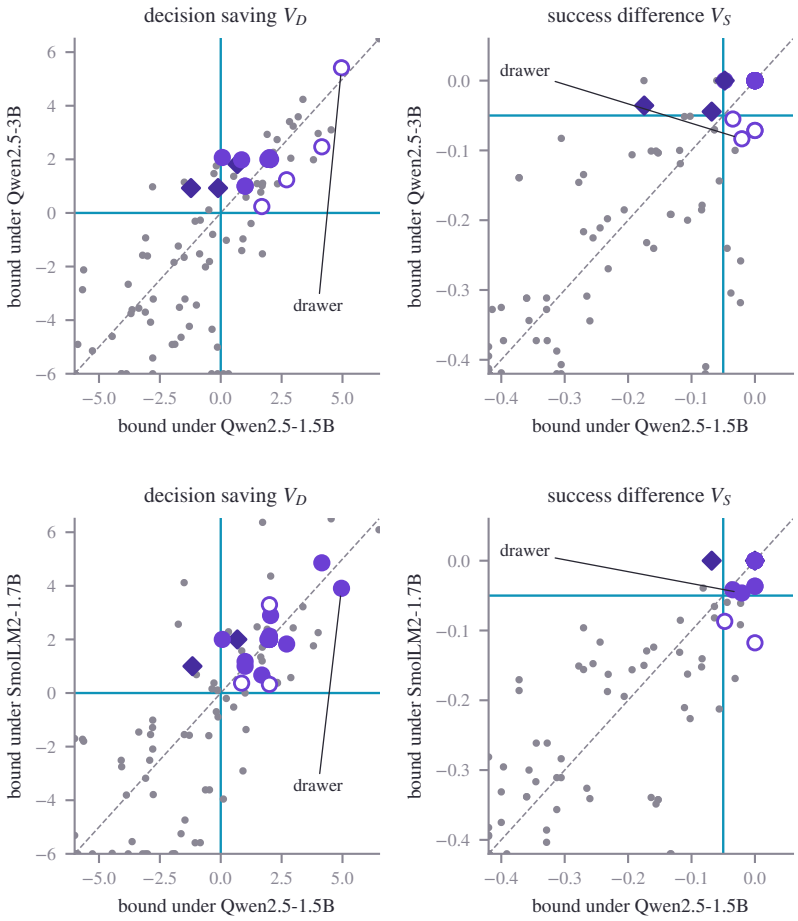


Figure 6: Both axes of the value certificate for the candidates of Figure 2, Qwen2.5-3B above and SmolLM2-1.7B below, each against Qwen2.5-1.5B. Marks carry the same meaning as in the body, and bounds outside an axis range sit on its edge.

Both value axes across policies. Figure 6 adds the success axis to Figure 2. Against Qwen2.5-1.5B, success bounds move by a median of 0.048 under Qwen2.5-3B and 0.045 under SmolLM2. Such shifts match the margin $\varepsilon_S = 0.05$ itself, so a candidate near the margin can pass for one policy and fail for another. Saving bounds move by medians of 1.14 and 1.31 decisions, whereas effect bounds move by only 0.02 and 0.03.

Certificate ablation. Figure 7 plots Table 2 with the individual seeds. Every seed of both variants without the value certificate stays far below the primitive policy, while PRA's seeds stay close to it. No single seed therefore carries the pooled pattern.

Table 5: Removing one certificate under each frozen policy on the ALFWorld development split. Each variant reports its paired success difference with the one-sided 95% lower bound and its fidelity.

Frozen policy	w/o value		w/o effect		**PRA (ours)**	
	ΔSucc. (LCB)	Fid.	ΔSucc. (LCB)	Fid.	ΔSucc. (LCB)	Fid.
Qwen2.5-1.5B	–0.310 (–0.357)	0.918	–0.029 (–0.048)	0.735	–0.002 (–0.014)	0.872
Qwen2.5-3B	–0.286 (–0.326)	0.894	+0.007 (–0.007)	0.815	–0.002 (–0.014)	0.983
SmolLM2-1.7B	–0.264 (–0.317)	0.898	–0.017 (–0.057)	0.789	–0.002 (–0.021)	0.826
SkillRL-7B	+0.024 (+0.000)	0.975	+0.002 (–0.024)	0.353	+0.019 (+0.005)	0.974

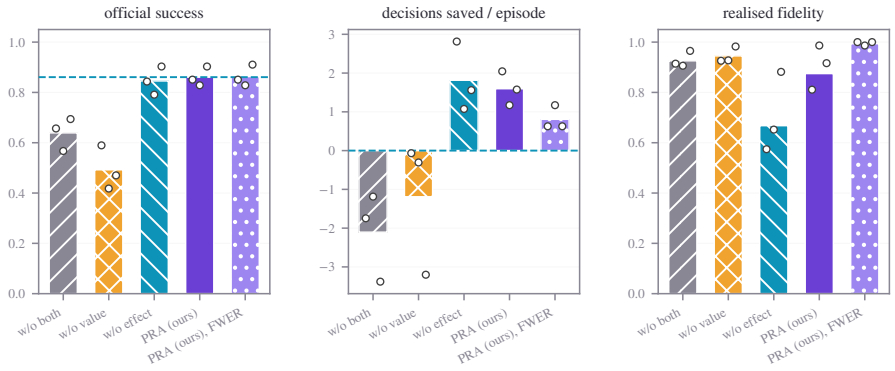


Figure 7: Values of Table 2 with the individual seeds shown. Hatched bars drop a certificate. Grey drops both, gold the value certificate and teal the effect certificate. Indigo bars are PRA, dotted at the family-wise level. Bars are means over three seeds and open markers the seeds. A dashed line in the left panel marks the primitive policy.

Table 5 repeats that ablation under every frozen policy on the development split. For the three cloned policies, removing the value certificate loses 0.26 to 0.31 success at fidelities near 0.9. Removing the effect certificate instead drops fidelity to between 0.74 and 0.82. Under the released agent the variant without the value certificate keeps success. Candidates for that agent come from its own successful episodes, which it already executes well. Proposing from expert traces instead shows what that protection rests on. On seed 41 construction statistics alone then admit abstractions that cost the same agent 0.207 success, and PRA's certified saving falls from 1.04 to 0.27 decisions. Proposing from the deploying policy therefore yields more certifiable actions, while only certification keeps the task under every policy.

Table 6: Development split, ALFWorld valid-seen. Method selection used this split only, and the held-out split in Table 1 carries the confirmatory claim. Both uncertified rows deploy PRA's own proposals with no certificate, so they are ablations of our method and not ports of published systems. Bold follows Table 1.

Arm	#Adm.	Success ↑	ΔSucc. (LCB) ↑	ΔDec. (LCB) ↑	Fidelity ↑
Flat	–	**0.895**	+0.000 (+0.000)	+0.00 (+0.00)	–
Handcrafted	11.7	0.876	–0.019 (–0.038)	+1.73 (+1.22)	0.738
PRA (ours)	7.0	0.893	–0.002 (–0.014)	+2.31 (+1.53)	**0.872**
PRA proposals, uncertified, all†	40.0	0.469	–0.426 (–0.498)	+4.58 (+3.45)	0.255
PRA proposals, uncertified, top-B†	8.0	0.452	–0.443 (–0.555)	+3.29 (+1.87)	0.238

Development split and policy swaps. Table 6 reports the split on which we selected methods. PRA saves 2.31 decisions at a success of 0.893 against Flat's 0.895. Handcrafted stays within the margin on this split but not on the held-out one. Both uncertified rows in Table 6 deploy PRA's own proposal pool without a certificate, so they are ablations of our method and not ports of published systems. On the held-out split they reach success 0.363 and 0.346 against Flat's 0.861 while removing 4.03 and 2.90 decisions. Compression without admission therefore loses the task rather than trading it for decisions.

Table 7 repeats the development comparison under Qwen2.5-3B. Every uncertified catalogue again falls outside the margin, and PRA keeps success at 0.862 against Flat's 0.864.

Table 8 deploys each Qwen policy with the catalogue certified for the other. Neither swap improves both saving and success. Under the larger policy's catalogue the smaller policy keeps its success but saves only 0.76 decisions instead of 2.31. Under the smaller policy's catalogue the larger policy saves 2.04 decisions instead of 0.76. Success difference then falls to -0.017 with a lower bound of

Table 7: Certifying against a different frozen policy: the pipeline re-run end to end with Qwen2.5-3B on the development split, 140 tasks × three seeds. Bold follows Table 1.

Arm	#Adm.	Success ↑	ΔSucc. (LCB) ↑	ΔDec. (LCB) ↑	Fidelity ↑
Flat	–	**0.864**	+0.000 (+0.000)	+0.00 (+0.00)	–
Handcrafted†	11.0	0.836	–0.029 (–0.062)	+1.45 (+0.50)	0.686
PRA proposals, uncertified, all†	40.0	0.469	–0.395 (–0.450)	+4.97 (+3.19)	0.155
PRA proposals, uncertified, top-B†	8.0	0.390	–0.474 (–0.524)	+3.20 (+1.16)	0.121
PRA (ours)	5.7	0.862	–0.002 (–0.014)	+0.76 (+0.55)	**0.983**

Table 8: Deploying each Qwen policy with the catalogue certified for the other, on the development split over three seeds. Parentheses report one-sided 95% lower bounds on the paired difference against the same policy's Flat run. (own) marks the catalogue certified for the deployed policy.

Deployed policy	Catalogue certified for	ΔSucc. (LCB) ↑	ΔDec. (LCB) ↑
Qwen2.5-1.5B	Qwen2.5-1.5B (own)	–0.002 (–0.014)	+2.31 (+1.53)
Qwen2.5-1.5B	Qwen2.5-3B	–0.005 (–0.014)	+0.76 (+0.59)
Qwen2.5-3B	Qwen2.5-3B (own)	–0.002 (–0.014)	+0.76 (+0.55)
Qwen2.5-3B	Qwen2.5-1.5B	–0.017 (–0.045)	+2.04 (+1.20)

-0.045 that nearly reaches the margin. A borrowed catalogue therefore buys decisions with success that the larger policy's own certification refused to spend when it rejected the drawer candidate.

Table 9: SkillRL-SkillsOnly inserts the released skill bank into the frozen policy's prompt through the original retrieval mechanism. Unlike full SkillRL, the policy is not trained with the bank. SkillRL-SkillsOnly needs a 2048-token context, so it is paired against the same policy at that context without the bank. A 768-token Flat is listed for scale.

Split	Arm	Success ↑	Decisions ↓	ΔSucc. (LCB)	ΔDec. (LCB)
valid-seen	Flat, 768 tokens	0.895	19.85	–	–
valid-seen	Flat, 2048 tokens	0.888	19.98	–	–
valid-seen	SkillRL-SkillsOnly, 2048 tokens	0.795	22.23	–0.093 (–0.131)	–2.24 (–3.62)
valid-unseen	Flat, 768 tokens	0.861	21.20	–	–
valid-unseen	Flat, 2048 tokens	0.861	21.20	–	–
valid-unseen	SkillRL-SkillsOnly, 2048 tokens	0.766	24.21	–0.095 (–0.164)	–3.01 (–4.81)

Table 9 transfers SkillRL's released skill bank onto our frozen policy without retraining it. Retrieved skills lengthen episodes and cost 0.093 success on valid-seen and 0.095 on valid-unseen. A skill bank written for a policy trained with it therefore does not transfer to a frozen one, and PRA instead certifies each action against the deploying policy.

E HELD-OUT CHECKS

Commit outcomes. Of 576 commits under the cloned policy on the held-out ALFWorld split, 504 ran to completion and every one realised the effect it declared. Unconditional fidelity of 0.875 is therefore a completion rate, and fidelity conditional on completion is 1.000. All 72 shortfalls are aborts, in which the binder emitted a command the environment did not accept and control returned to the policy. No admitted abstraction finished and left the wrong post-state. Under the released agent 18 of 284 completed commits did. Under the cloned policy, an admitted action on the evaluated commits thus either realised its declared effect or handed the decision back. We observed this execution pattern, and it does not show that an abort is free of partial side effects or cost in general. Figure 8 counts completed against made commits for every arm and adds savings in tokens and environment steps. Units of the SaS BPE port save more tokens but complete 479 of 2,460 commits on ALFWorld and none of 3,990 on WebShop. PRA completes 504 of 576 and 2,006 of 2,100.

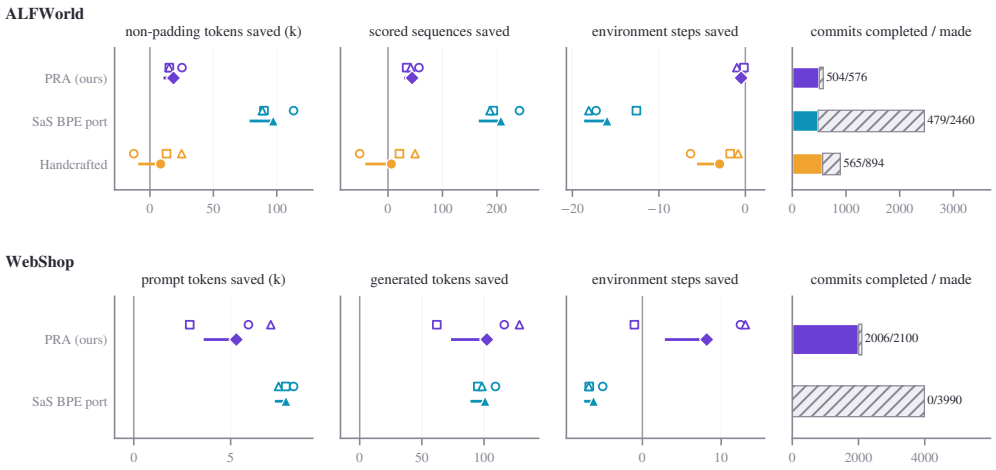


Figure 8: What admission saves besides decisions, paired on the same held-out episodes. Values are Flat minus arm, so a positive value is a saving. Hollow marks are per-seed means and the filled mark the pooled mean with a one-sided 95% lower bound. Each interface meters the policy differently, so each row carries its own units.

Catalogue stability. Recomputing admission offline over a grid of both frozen thresholds from the stored certification bounds measures how the composition of the catalogue moves, not how its deployed outcome moves (Appendix H). Lowering τ_F to 0.60 or 0.50 admits one more abstraction on two of three seeds, and raising it to 0.90 admits one fewer on two seeds. Admission therefore moves only at the margin of the catalogue, and Figure 5 shows that most candidates sit well away from the effect threshold.

Where the admitted abstractions come from. Table 10 lists every admitted abstraction with its bounds. Seeds share most of their object-processing and navigation sequences, and two of three admit the drawer candidate of Section 1. On WebShop every seed admits the same purchase entry. Certified catalogues therefore stay legible and mostly stable across seeds.

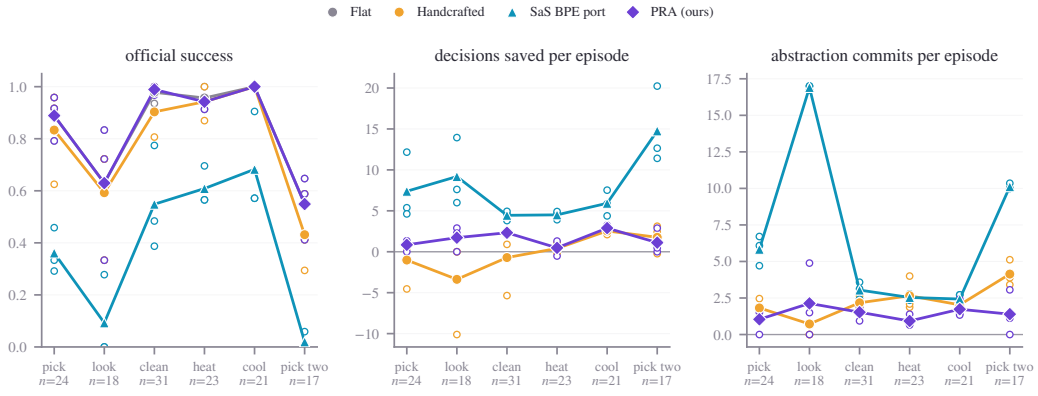


Figure 9: Held-out ALFWorld outcomes by task type, a descriptive split made after the split was opened. Lines are arms pooled over seeds, hollow marks the three seeds, and n counts tasks per type. PRA's saving concentrates on the tasks its admitted abstractions serve, and the SaS BPE port saves most on the task types where it fails.

Figure 9 splits the held-out episodes by task type. PRA's saving concentrates on cleaning and cooling tasks, which its admitted abstractions serve. SaS saves most on look and pick-two tasks, where its success is lowest.

Table 10: Every admitted abstraction per seed, with its lower bounds and firing games.

Seed	Lifted operator sequence	L	$F(z)$	$V_D(z)$	$V_S(z)$	Games
ALFWorld, frozen catalogue deployed on the held-out split						
41	go_to(ENTRY[drawer]) → open(SAME[0,0])	2	0.778	+4.96	−0.021	97
41	take_from(ENTRY, ENTRY) → go_to(TYPE[sinkbasin]) → clean_with(SAME[0,0], SAME[1,0]), guard clean	3	1.000	+0.06	+0.000	31
41	take_from(ENTRY[task], ENTRY) → go_to(TYPE[sinkbasin]) → clean_with(SAME[0,0], SAME[1,0]), guard clean	3	1.000	+1.94	+0.000	31
41	take_from(ENTRY[task], ENTRY) → go_to(TYPE[fridge]) → cool_with(SAME[0,0], SAME[1,0]), guard cool	3	1.000	+2.00	+0.000	44
41	take_from(ENTRY[task], ENTRY) → go_to(TYPE[sinkbasin]), guard clean	2	1.000	+1.00	+0.000	31
42	close(ENTRY[microwave]) → go_to(TYPE[garbagecan]) → go_to(TYPE[fridge]) → open(SAME[2,0])	4	1.000	+2.71	+0.000	51
42	take_from(ENTRY[task], ENTRY) → go_to(TYPE[sinkbasin]) → clean_with(SAME[0,0], SAME[1,0]), guard clean	3	1.000	+2.00	+0.000	33
42	take_from(ENTRY, ENTRY) → go_to(TYPE[sinkbasin]) → clean_with(SAME[0,0], SAME[1,0]), guard clean	3	1.000	+2.00	+0.000	33
42	take_from(ENTRY[task], ENTRY) → go_to(TYPE[fridge]) → cool_with(SAME[0,0], SAME[1,0]), guard cool	3	1.000	+2.00	+0.000	44
42	take_from(ENTRY, ENTRY) → go_to(TYPE[fridge]) → cool_with(SAME[0,0], SAME[1,0]), guard cool	3	0.933	+2.00	+0.000	45
42	close(ENTRY[microwave]) → go_to(TYPE[garbagecan]) → go_to(TYPE[fridge])	3	1.000	+1.69	+0.000	51
42	take_from(ENTRY[task], ENTRY) → go_to(TYPE[sinkbasin]), guard clean	2	1.000	+1.00	+0.000	33
42	take_from(ENTRY, ENTRY) → go_to(TYPE[sinkbasin]), guard clean	2	1.000	+1.00	+0.000	33
43	go_to(ENTRY[drawer]) → open(SAME[0,0])	2	0.808	+4.15	−0.035	86
43	close(ENTRY[microwave]) → go_to(TYPE[garbagecan]) → go_to(TYPE[fridge]) → open(SAME[2,0])	4	1.000	+0.86	−0.048	63
43	take_from(ENTRY[task], ENTRY) → go_to(TYPE[sinkbasin]) → clean_with(SAME[0,0], SAME[1,0]), guard clean	3	1.000	+2.00	+0.000	31
43	take_from(ENTRY, ENTRY) → go_to(TYPE[sinkbasin]) → clean_with(SAME[0,0], SAME[1,0]), guard clean	3	1.000	+2.00	+0.000	31
43	take_from(ENTRY[task], ENTRY) → go_to(TYPE[fridge]) → cool_with(SAME[0,0], SAME[1,0]), guard cool	3	1.000	+2.00	+0.000	42
43	take_from(ENTRY, ENTRY) → go_to(TYPE[fridge]) → cool_with(SAME[0,0], SAME[1,0]), guard cool	3	0.917	+2.05	+0.000	43
43	take_from(ENTRY[task], ENTRY) → go_to(TYPE[sinkbasin]), guard clean	2	1.000	+1.00	+0.000	31
43	take_from(ENTRY, ENTRY) → go_to(TYPE[sinkbasin]), guard clean	2	1.000	+1.00	+0.000	31
WebShop, frozen catalogue deployed on the held-out split						
41	click_option → click_buy	2	1.000	+15.45	+0.037	82
41	click_option → click_option → click_buy	3	0.829	+13.96	−0.024	82
42	click_option → click_buy	2	1.000	+14.09	+0.010	105
42	click_option → click_option	2	0.897	+4.59	−0.048	105
43	click_option → click_buy	2	1.000	+15.13	+0.022	89
43	click_option → click_option → click_buy	3	0.831	+14.04	−0.022	89
43	search → click_product	2	1.000	+0.01	−0.017	120

F COMPONENT CHECKS

Table 11: Component checks on the development split, reported at the seed counts run. Removing task guards from proposal leaves fewer certifiable abstractions and less saving. Changing the base policy leaves the pipeline intact in both directions of capacity.

Variant	Policy	Seeds	Admitted	ΔDec. vs Flat	Fidelity
PRA (ours)	Qwen2.5-1.5B	3	7.0	+2.31	0.872
PRA w/o task guard	Qwen2.5-1.5B	2	2.5	+1.96	0.875
PRA (ours), smaller policy	Qwen2.5-0.5B	1	5.0	+0.95	1.000
PRA (ours), larger policy	Qwen2.5-3B	3	5.7	+0.76	0.983

Table 11 reports two variants on the development split at the seed counts shown. Neither enters a held-out claim.

Guards enter at proposal, so removing them changes which templates exist rather than which survive certification. Without a task guard a template must hold across every task phase, and few do. Certification then admits 2.5 abstractions per seed against 7.0 for the full method, and the saving falls from 2.31 to 1.96 decisions at the same fidelity. Guards therefore widen the certifiable pool, and certification keeps what they add faithful.

G SETTINGS

Table 12: Recent hierarchical and skill-based agents under their released protocols. Both retrain the policy, whereas PRA leaves it fixed and decides which actions it may launch. PRA is therefore not a row here. Table 1 instead passes SkillRL's released agents through PRA.

			ALFWorld	WebShop	
Method	Backbone	Policy training	Success ↑	Success ↑	Score ↑
HiPER (Peng et al., 2026)	Qwen2.5-1.5B	Hierarchical RL	0.891	0.695	0.875
SkillRL (Xia et al., 2026)	Qwen2.5-7B	Skill-augmented GRPO	0.859	0.781	0.883

Recent agent systems. Table 12 reports two recent agent systems under their released protocols for scale. HiPER is trained with its released training script on Qwen2.5-1.5B for 150 epochs, on one H200 for ALFWorld and on four for WebShop. Global PPO mini-batch size and every other setting follow the released script. SkillRL is evaluated from its released Qwen2.5-7B checkpoints with its released evaluation script on two H200s. Both use their released evaluation protocols, ALFWorld valid-seen and a 1,000-product WebShop subset with templated goals. For HiPER on WebShop, prompts are capped at 2,048 tokens by retaining the most recent context when necessary.

Released agents as frozen policies. We serve the released SkillRL checkpoints with vLLM in bfloat16. Prompts use SkillRL's own templates and its `<action>` parsing, with the two most recent steps of history. SkillRL's skill bank enters through template retrieval of six skills. Decoding keeps SkillRL's validation temperature of 0.4 with a seed fixed by run, episode and step. We make each decision once per episode, step and prompt. Paired runs therefore share their random numbers until an abstraction changes the page. Our WebShop setup keeps the full product corpus and human goals. Flat success of the released agent is therefore lower there than under its own 1,000-product protocol in Table 12.

Table 13 lists every value fixed before the held-out splits were opened. Frozen values cover proposal limits, the two admission thresholds and the selection level. Bootstrap size, minimum firing count, decoding and horizon are frozen as well. Values are identical across seeds and across policies. Only the frozen policy and its decoding change between the panels of Section 5.2 and the rows of Section 5.4.

Table 13: Every setting used, fixed before the held-out splits were opened. Admission thresholds are shared across environments and were not re-tuned on the second one.

	ALFWorld	WebShop
Policy		
Base model	Qwen2.5-1.5B-Instruct, revision `989aa798`	
Adaptation	LoRA r=16, α=32, dropout 0.05, all attention and MLP projections	
Optimiser	AdamW, learning rate 2×10^{-4}	
Batch $\times$ accumulation	8×8	4×8
Epochs	1	2
Context length	768	1536
Action selection	score all admissible	generate one action
Released policy	SkillRL-7B, temperature 0.4, seeded per step	
Candidate proposal		
Abstraction length L	2–6	2–5
Proposal source	512 policy episodes	human trajectories
Candidate pool cap	40	24
Per-entry quota	3	3
Per-operator quota	12	–
Minimum distinct games	30	–
Minimum occurrences	–	4
Minimum constructible games	15	–
Certification		
Effect-fidelity threshold τ_F	0.75	0.75
Success margin ε_S	0.05	0.05
Selection level α_{sel}	0.05	0.05
Family-wise level α_0	0.05	0.05
Bootstrap resamples	10,000	10,000
Minimum commit games $n_{\min}$	20	20
Commit-game cap	160	–
Certification games	256	120
Evaluation		
Horizon	50 steps	30 steps
Seeds	41, 42, 43	41, 42, 43
Development episodes	140	120
Held-out episodes	134	200

H THRESHOLD SENSITIVITY

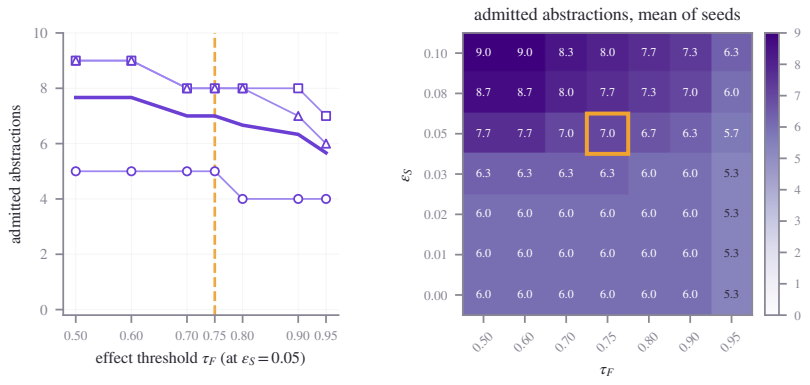


Figure 10: Admission recomputed offline over both frozen thresholds from the stored certification bounds, with no environment or policy call. Left, admitted abstractions against τ_F at the frozen ε_S, one thin line per seed and the mean in bold. Right, the mean over seeds on the full grid, the frozen setting outlined. Stability here is of catalogue composition, not of deployed outcome.

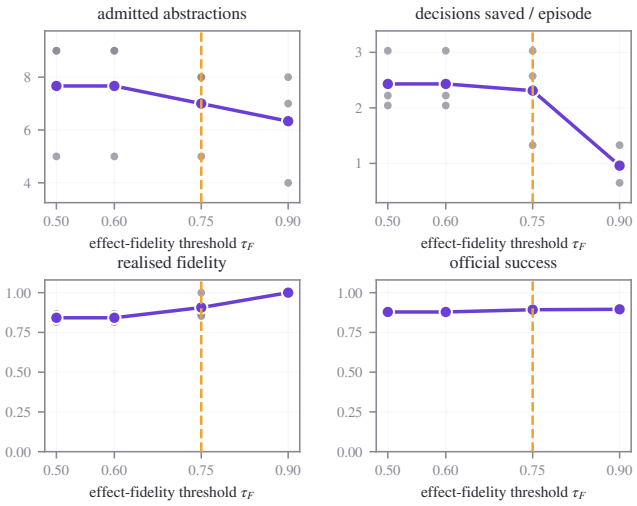


Figure 11: Deployed outcome of the catalogue admitted at each effect-fidelity threshold τ_F, on the development split. Values come from deployment, not from offline recomputation. Dark markers give the mean over three seeds, pale markers the seeds, and the dashed line the frozen setting. At $\tau_F = 0.90$ the catalogue shrinks, so saving falls while realised fidelity rises.

Sensitivity splits into two questions. Figure 10 recomputes admission offline from the stored certification bounds, so it reports how the composition of the catalogue moves with τ_F and ε_S. Figure 11 instead deploys the catalogue admitted at each τ_F and measures what it does on the split, which offline recomputation cannot answer. Both place $\tau_F = 0.75$ on a shallow slope rather than a cliff. Lowering τ_F to 0.50 admits one more abstraction on two seeds, adding 0.12 decisions of saving at 0.014 lower success and lower fidelity. Raising τ_F to 0.90 keeps success and makes every commit faithful but gives up more than half the saving.

I CERTIFICATION BUDGET

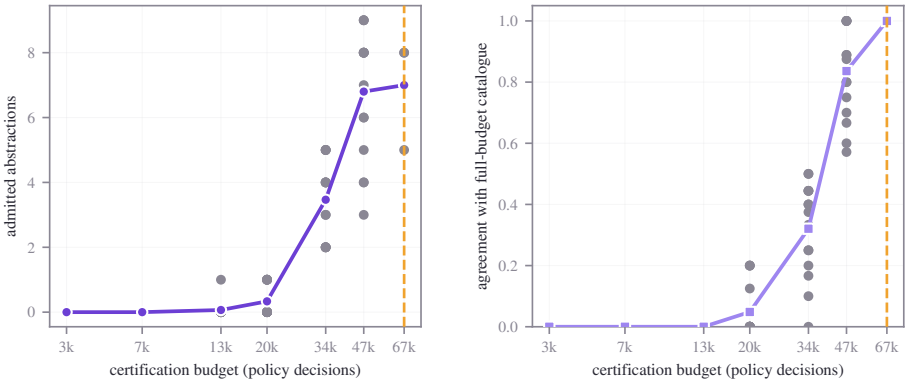


Figure 12: How much certification evidence the catalogue needs, re-running the frozen rule on subsampled calibration games. Grey points are individual subsamples over three seeds, the line their mean, and the dashed line the budget we spent. Each budget scales the admission cost of Table 14, proposal included, by the share of games kept. Admission nearly saturates at that budget. Deployment on an intermediate catalogue cannot be replayed offline.

Certification spends policy calls, so the yield curve asks how many of them the final catalogue actually needs. Figure 12 subsamples the recorded calibration games and re-runs the frozen rule on each

subsample. We report the number of admitted abstractions and the agreement with the full-budget catalogue. Nothing is re-executed in the environment, so the curve describes admission stability and not deployed performance. Below about a third of the budget the rule admits almost nothing. Admission then rises steeply to 3.5 abstractions per seed at half the evidence and to 6.8 at seven tenths. At that point the catalogue already agrees with the full-budget one at a Jaccard index of 0.84, so our budget sits just past the point where the catalogue stabilises.

Example 1: two frequent candidates that PRA rejects

Both candidates recur often enough in the policy's own successful episodes to enter the proposal pool, so a frequency criterion alone would promote them. Each comes from one failing region of Figure 5. Columns give the one-sided 95% lower bounds averaged over seeds, and red marks a bound that misses its threshold.

Lifted operator sequence	$F(z)$	$V_D(z)$	$V_S(z)$	Verdict
admission threshold	≥ 0.75	> 0	≥ -0.05	
close(ENTRY[microwave]) → go_to(TYPE[garbagecan]) → go_to(TYPE[fridge]) → open(SAME[2,0]) → close(SAME[2,0])	1.000	−1.15	−0.183	reject: value
close(ENTRY) → go_to(TYPE[garbagecan]) → go_to(TYPE[fridge]) → open(SAME[2,0]) → close(SAME[2,0])	0.445	+2.44	−0.281	reject: effect, value

One candidate executes as declared on every commit but its mean saving bound is negative. Another has a positive saving bound, yet its effect bound is only 0.445. Both also fail the success margin, so promoting them would have cost the policy tasks.

Example 1 shows two rejected candidates whose bounds sit far from every threshold, so each rejection follows from what the candidate does for this policy.

J COST BREAKDOWN

Table 14: Cost of admission on ALFWorld for the cloned Qwen2.5-1.5B policy, averaged over three seeds. Base-policy training is shared by every arm and is reported in tokens rather than charged to admission. Break-even divides the admission cost by the held-out decision saving.

	Policy decisions
Proposal episodes	10,636
Certification	56,834
Admission cost	**67,470**
Gating experiment, run and then excluded	72,999
Certification environment steps	115,535
Base-policy training, shared by all arms	6,197,630 tokens
Proposal pool	40 abstractions
Held-out saving	1.60 decisions / episode
Break-even	42,247 episodes
Held-out evaluation	134 episodes

Table 14 itemises admission for the cloned policy, which used 115,535 certification environment steps and 11.2 wall hours per seed over a 40-abstraction proposal pool. Only the wall hours include the excluded gating experiment. Proposal episodes account for 10,636 decisions and certification for 56,834. Admission therefore costs about 1,687 decisions per proposed abstraction, and the excluded gating experiment adds 72,999. Deployment saves 1.60 decisions and 1.85×10^4 non-padding tokens per episode on the held-out split, so break-even arrives after 42,247 episodes. At the larger development saving of 2.31 decisions the same admission cost repays after 29,214 episodes. No ordering of these accounts brings break-even within two orders of magnitude of either benchmark.

K ACTIVATION DETAILS

Each per-commit value sample is the number of continuation steps executed, signed by whether the commit realised its declared effect. A ridge predictor is fitted on one block of held-out train games and corrected on a disjoint block. A one-sided split-conformal residual correction (Lei et al., 2018; Romano et al., 2019) bounds a single future sample. For the drawer candidate its quantile lies between 1.86 and 1.91 and exceeds the sample range of $[-1, 1]$, so that gate never withholds the candidate. Other admitted abstractions have residual quantiles near zero. Bounding the conditional mean within score bands is non-degenerate and is what the reported contextual gate uses. Withholding admitted actions through the contextual gate gives no benefit, changing paired success by 0.0000 at a cost of 0.114 decisions.

L WEBSHOP INSTANTIATION

Two scope limits bound the evidence in the main text. Both environments are text interfaces with enumerable or on-screen actions, and we make no claim about unbounded-text action spaces. Absolute WebShop success is lower than published numbers for every arm because our setup keeps the full product corpus and human goals (Yao et al., 2022), so each panel supports a within-environment comparison and not a benchmark-level claim.

Our admission rule is unchanged across environments, and only the representation is instantiated per environment. We fixed the WebShop instantiation recorded below before any WebShop certification or evaluation run. No effect semantics could therefore be revised in response to a WebShop number.

Table 15: Declared effect per typed operator. Every predicate is read from the page's own rendered controls, which the environment returns to any agent, and from the observation the policy is already conditioned on. Unlike the text-adventure interface, this is not an oracle enumeration of valid actions.

Operator	Declared effect
`search`	the page comes to list products
`click_product`	the page comes to offer a purchase control
`click_nav`	the page state or observation changes
`click_info`	an information tab remains available and the observation changes
`click_option`	the purchase control remains available
`click_buy`	the episode terminates

WebShop's action space is `search[query]` and `click[target]`. Each raw target maps to exactly one of six typed operators (Table 15). Search queries become `search`, ASIN-shaped targets `click_product` and navigation controls `click_nav`. Information tabs become `click_info`, the purchase button `click_buy`, and every other target `click_option`. Only `search`, `click_product` and `click_option` carry a scene-specific argument. Constant targets make abstractions over the other three transferable across episodes. Our typing parses every action in the 1,571 public human trajectories used for construction.

A commit is faithful when every emitted step realises the effect its operator declares and the continuation runs to completion. Some signals are available only to the evaluator or the task generator. Such signals include the scalar reward with its sub-scores and the goal's target product and gold attributes. Product fields not rendered on the page belong to them too. None of them enters an effect, fidelity or construction predicate. Task success enters only through the success non-inferiority axis. We estimate that axis on certification goals disjoint from the held-out split, exactly as in the text-adventure instantiation.

Our split is the official one by human-goal index, used without altering its semantics. Held-out sessions form the official test range and development sessions the official evaluation range. Policy training, candidate construction and certification take disjoint sub-ranges of the official training range.

On WebShop both policies emit one action per call, whereas the cloned text-adventure policy scores every admissible command per call. Absolute compute is therefore not commensurable across the two environments, and we compare arms only within an environment. No handcrafted arm is constructed for WebShop: the text-adventure handcrafted control was an expert selection from an

existing proposal pool, and inventing a WebShop analogue after seeing the environment would be a new design choice made for table symmetry rather than a control.

M WEBSHOP RESULTS

Development on the official WebShop evaluation range preceded any held-out session. Over three seeds of 120 paired episodes each, Flat reaches 0.117 success and 0.267 score at 21.17 decisions. PRA reaches 0.111 success and 0.353 score at 11.21 decisions. Certification admits two or three abstractions per seed. All three seeds admit the same entry, which selects a product option and then purchases.

Example 2: one WebShop session under three arms

Held-out session 0 under the cloned policy on seed 41, with the query `clip-in hair extension`. Plain actions are policy decisions. Bracketed actions are committed without consulting the policy, by the [SaS unit] or by PRA's [certified entry]. *kit* abbreviates the option `big kizzy brown kizzy clips - no sew extension clips kit`.

Arm	Trajectory	Dec.	Bought
Flat	search, product, kit, features, prev, description, prev, brown, features, prev, kit, description, prev, buy now	14	✓
SaS BPE port	(search, [product, black, black, description, back to search]) ×5, stopped by the 30-step horizon	5	✗
PRA (ours)	search, product, kit, [black, buy now]	**3**	✓

Both abstraction arms commit blindly, and only their catalogues differ. SaS commits a unit that was never tested against this policy, and every commit returns the session to search before any purchase. Once the policy has chosen an option, PRA commits a certified entry that selects one more option and then buys. PRA therefore leaves the policy 3 decisions instead of 14, and the purchase still completes.

Rejections are individually interpretable, and each axis rejects something the other would keep. `click_option` → `click_option` realises its declared effect with lower bounds of 0.977 and 0.979 on seeds 41 and 43 and saves about 7.1 decisions there. Both seeds still reject it because success-difference lower bounds of −0.073 and −0.079 fall outside the non-inferiority margin. `search` → `click_product` → `click_option` saves 1.4 decisions and passes the value axis on two seeds. An effect bound of 0.608 then fails the effect axis. Three consecutive navigation clicks fail the value axis and have no estimable effect bound. A byte-pair vocabulary learned on the same traces instead recovers whole demonstrated purchase sequences and commits them open-loop. On the held-out range it commits about twice as often as PRA, and its success falls to 0.017.

Decision saving on WebShop reads as policy-specific. Most of it comes from committing to purchase once the policy has reached a buyable page, where the primitive policy tends to keep browsing. Ending an otherwise wandering trajectory both shortens the episode and raises the score. Such saving is a property of this behaviour-cloned policy rather than a general compression rate. Because the released SkillRL agent already buys when it should, certification rejects the same entry for it (Section 5.4).

Example 2 shows this pattern on one held-out session, chosen for illustration. Table 1 reports the averages over all sessions.

N CROSS-FAMILY CERTIFICATION

Section 5.4 summarises the policies, and Table 16 reports the family comparison in full. To separate family from capacity we repeat construction, certification and deployment on ALFWorld for SmolLM2-1.7B-Instruct. SmolLM2 matches Qwen2.5-1.5B-Instruct in parameter count but differs in pretraining corpus, tokenizer and vendor. Every threshold, split and seed is the one the Qwen runs used, and only the frozen policy changes. Prompts tokenize 3% longer under the new tokenizer and truncate at the same rate under the unchanged budget, so the shared budget favours neither policy.

Table 16: Certifying against a policy outside the Qwen family. Both panels run the pipeline end to end on the development split with 140 tasks × three seeds and share every threshold and split. An uncertified row deploys PRA's whole proposal pool, and bold follows Table 1.

Arm	#Adm.	Success ↑	ΔSucc. (LCB) ↑	ΔDec. (LCB) ↑	Fidelity ↑
Panel A: Qwen2.5-1.5B-Instruct					
Flat	–	**0.895**	+0.000 (+0.000)	+0.00 (+0.00)	–
PRA proposals, uncertified†	40.0	0.469	–0.426 (–0.498)	+4.58 (+3.45)	0.255
PRA (ours)	7.0	0.893	–0.002 (–0.014)	+2.31 (+1.53)	**0.872**
Panel B: SmolLM2-1.7B-Instruct					
Flat	–	**0.824**	+0.000 (+0.000)	+0.00 (+0.00)	–
PRA proposals, uncertified†	40.0	0.448	–0.376 (–0.464)	+6.00 (+3.80)	0.259
PRA (ours)	6.0	0.821	–0.002 (–0.021)	+2.83 (+2.06)	**0.826**

Both admission conditions hold for the new policy. PRA removes 2.83 decisions per episode with a lower bound of $+2.06$, at a success difference of -0.002 whose lower bound of -0.021 stays inside the -0.05 margin. Removing certificates again costs success under SmolLM2. SmolLM2's uncertified proposals fall to 0.448 success at a fidelity of 0.259. Without the value certificate PRA loses 0.264 success. SmolLM2 saves 2.83 decisions against 2.31 for Qwen, because the weaker policy spends more decisions on the sequences the catalogue replaces.

Admitted sets remain policy-specific at their margin. Pooled over seeds SmolLM2 admits all nine abstractions the Qwen policy admits and one cooling sequence more. Figure 6 places every candidate that both policies proposed by its bounds under each policy, and the marks scatter off the diagonal in both directions. What transfers across families is the test, not the catalogue it returns.

O SCOPE

Admission certifies lifted operator sequences run by a deterministic binder, so a behaviour whose next action depends on an observation the binder cannot resolve is not represented faithfully. Such a behaviour can still be proposed as a lifted template, and blind execution then fails the effect certificate. Richer binders widen what can be proposed without changing the certification rule.

A certified catalogue is a statement about the frozen policy it was certified against. Section 5.4 separates admission by capacity, by model family and by training on four policies of up to 7B parameters. Our evidence therefore does not cover policies much larger than those.

Admission is a per-candidate selection rule backed by bootstrap lower bounds on certification games. Neither the rule nor the paired held-out evaluation extends to deployment distributions other than the ones measured here.