R^2 FLOW: RECURSIVE SELF-IMPROVEMENT VIA RECURSIVE SKILL EVOLUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

LLM-based agents can improve themselves across tasks by reusing and revising the skills they orchestrate into executable procedures. Flow-based training fits this loop: it samples procedures in proportion to reward, and the flow through each skill credits it for the next library revision. Three obstacles stand in the way of making this self-improvement reliable: flow training suffers strategy collapse over tree-structured histories; nonnegative flow-based credit rewards frequent use as if it were benefit; and library edits rest on the task reward the policy optimizes. We introduce **R^2 Flow**, a recursive self-improvement framework that alternates policy learning, independent verification, and versioned skill-library updates on a shared-state orchestration graph. The graph merges histories that differ only in the order of independent steps, allowing flow training to pool evidence across equivalent executions. A flow-share readout of the trained flow, invariant to the backward policy, and a separate signed utility rank which skills to change, verifier evidence decides whether an edit is warranted, and a residual-variance plateau sets when to update. Committed edits reshape the graph the next policy learns on, realizing recursive skill evolution. Across question answering, mathematical reasoning, interactive decision making, and code generation, R^2 Flow improves task accuracy and library-edit precision over heuristic orchestration, reinforcement learning, and skill-evolution baselines, and transfers across executors. Code is available at `https://anonymous.4open.science/r/r2flow-DD00`.

1 INTRODUCTION

In recent years, LLM-based agentic systems have come to orchestrate reusable skills into executable procedures (Yao et al., 2022b; Wang et al., 2023; Yue et al., 2026), shifting the unit of automation from a single answer to an entire procedure. A central goal is therefore not only to complete a task but to improve the procedure used later (Fang et al., 2025; Ren et al., 2026; Chen et al., 2026b). This makes self-improvement a recursive loop: each phase trains the policy on a versioned library and updates that library from what it learned, and the next phase repeats this on the new library. Without human review, such a loop must admit each update on evidence outside the training reward.

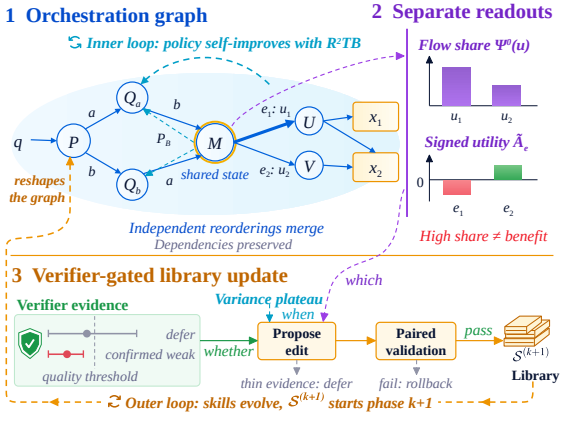


Figure 1: R^2 Flow trains a flow on shared states (inner loop), reads share and signed utility separately, and gates library edits on verified evidence (outer loop).

Task orchestration organizes primitive actions and reusable skills into structured trajectories on an orchestration graph (Figure 1, panel 1), where each node is an execution state and each edge is one orchestration event, which makes complex tasks more controllable and composable (Zhuge et al., 2024; Zhang et al., 2025; Yun et al., 2026; Tieu et al., 2026). Orchestration has accordingly evolved from static human-authored pipelines to graphs generated or edited at run time (Zhang et al., 2025; 2026d; Yue et al., 2026), and further to policies trained directly from execution feedback (Zhang

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

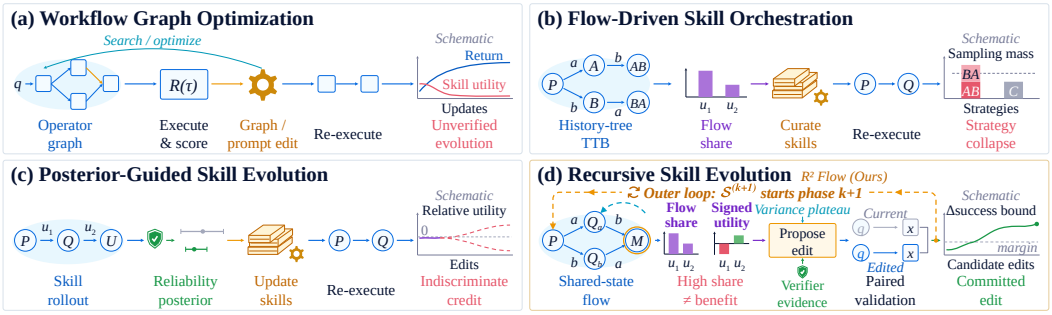


Figure 2: Four orchestration paradigms. (a) Workflow graph optimization hides a harmful skill in the return. (b) Flow-driven orchestration learns AB and BA twice on a history tree, so evidence never pools. (c) Posterior-guided evolution scores reliability, not signed utility, so success reads as benefit. (d) R^2 Flow pools evidence on shared states, separates share from utility, and commits verified edits.

et al., 2026e; Zhang, 2026b). However, these policies are still trained on tree-structured histories, where the same steps taken in a different order are never recognised as one procedure.

To address these issues, *graph-based orchestration* (Figure 2a) retrieves prebuilt skills or searches and edits workflow graphs at run time (Zhang et al., 2025; Zhuge et al., 2024; Yun et al., 2026; Li et al., 2026a; Tieu et al., 2026). *Step-level credit assignment* (Figure 2b) converts a terminal reward into finer-grained signals through process reward models, hindsight advantage estimation, and GFlowNet balance objectives that sample in proportion to reward (Bengio et al., 2021; Malkin et al., 2022; Zhang et al., 2026e; Tao et al., 2026; Tan et al., 2026). *Dynamic skill libraries* (Figure 2c) grow capability by curating skills with LLM-as-judge prompting, execution feedback, or posterior beliefs over skill reliability (Wang et al., 2023; Wu et al., 2026; Tang et al., 2026; Ouyang et al., 2026).

However, joining these lines of work into one loop faces three challenges. **(i) Strategy collapse.** When a state is the full interaction history, every state has exactly one parent, so the orchestration graph is a tree and the backward policy is trivially one (Malkin et al., 2022; Fawkes & Hartford, 2026; Wang et al., 2026). Executions that differ only in the order of independent steps sit on different branches and never share evidence; the flow then spreads probability over reorderings of one procedure rather than across genuinely different strategies, so a procedure with more admissible orderings receives proportionally more sampling mass, and more weight in the next library edit (Bengio et al., 2021; Zhang et al., 2026e). **(ii) Indiscriminate credit assignment.** Flow shares and usage counts are non-negative by construction and reliability posteriors score whether a call succeeds, so none says whether a skill helps, and a frequently invoked harmful skill looks critical and survives into the next library (Zhang et al., 2026e; Wu et al., 2026; Li, 2026). **(iii) Unverified skill evolution.** Edits are admitted on the same task reward whose attribution is in question, and on point estimates that cannot separate insufficient evidence from confirmed failure, so a skill can be pruned before its evidence accumulates (Zhang et al., 2025; Wu et al., 2026; Zhang et al., 2026f; Zhang, 2026c;a).

We introduce **R^2 Flow** (Figure 2d), running this loop through three designs, each serving the library update. The first, the **Orchestration Graph**, replaces the history tree of trained orchestration policies: it merges histories differing only in the order of independent steps into one state, so their evidence pools at that state; when a merged state has several legal predecessors, the backward policy is a genuine choice among them. Built on the graph, we further propose **flow share**: a flow network is trained on the shared states with the balance objective R^{2}TB, and each skill's share of the flow is read from the trained policy's rollouts weighted by their residuals, a readout invariant to the backward policy. In contrast to credit that reads use as benefit, it is kept separate from a signed utility that says whether the skill helps. R^2 Flow then performs **verifier-gated skill-library updates**, replacing admission on the task reward: independent verifiers label skill invocations, edge flow picks which ones to verify, and an edit refines or prunes a skill only when verified evidence supports it, waiting otherwise. The R^{2}TB residual also times the loop: a plateau in its variance ends a phase and triggers the next library update. Together the designs close one recursive loop: each phase trains on the library the previous phase committed, and its readouts, posterior and residual decide the next commit.

We evaluate on twelve datasets across four task domains, six held out, reporting task accuracy, executor transfer, and library-edit precision against heuristic orchestration, reinforcement learning (Zhang et al., 2026d), and skill-evolution baselines (Ni et al., 2026; Yang et al., 2026).

2 Related Work

Self-Improving Agents. Self-improving agents rewrite scaffolding, meta-agent designs, or their own logic and code (Zelikman et al., 2023; Hu et al., 2025; Yin et al., 2025; Zhang et al., 2026c), in loops framed as bounded or open-ended self-modification (Chen et al., 2026b). Skill evolution co-trains the library with the policy (Xia et al., 2026; Zhang et al., 2026e), Bayesian curation adds finite-sample uncertainty, and verification screens edits (Guo et al., 2026; Wu et al., 2026; Chen et al., 2026a; Zhang et al., 2026a). Most admit an edit on task-level outcomes, so the edit is judged by the signal whose attribution is in question, where an optimiser exploits its evaluator (Thaman, 2026). R^2 Flow instead admits library edits on verifier labels outside the reward.

Agent Task Orchestration. LLM agents orchestrate tasks in loops that plan, act, and revise on execution feedback (Yao et al., 2022b; Yue et al., 2026). Existing work spans operator graphs generated by search or edited under structured feedback (Zhang et al., 2025; Zhuge et al., 2024; Zhang et al., 2026d; Li et al., 2026a; Tieu et al., 2026), reusable skill memory, and self-evolving architectures (Wang et al., 2023; Tang et al., 2026; Xu et al., 2026b; Bai et al., 2026; Lin et al., 2026). Trained orchestration policies define states over raw histories, so a procedure and its reorderings never share statistics. R^2 Flow instead trains its flow on a dependency-aware shared-state graph of executable histories, where reorderings meet at one state (as in reasoning DAGs; Yu et al., 2024).

3 Preliminaries

Definition 1: Orchestration State Graph. Task orchestration is a directed graph $\mathcal{G}$ whose vertices are interaction histories H_t and whose edges are orchestration events. The update $H_t = H_{t-1} \oplus (r_t, e_t, o_t^{\text{exec}})$ appends one record, so $\mathcal{G}$ is acyclic and every non-root history has one parent.

Definition 2: Orchestration Trajectory. A complete path from the initial history H_0 to a terminal state defines an orchestration trajectory:

$$\tau = \left\{(r_t,\ e_t,\ o_t^{\text{exec}})\right\}_{t=1}^{T} \ \Rightarrow\ y_q, \tag{1}$$

where r_t is the reasoning text, $e_t = (u_t, \text{args}_t)$ the event, and o_t^{exec} the execution feedback. It ends on acceptance or budget exhaustion; its terminal outcome x earns a reward $R(x) \in [0, 1]$.

Problem Statement. Given a task q and an environment $\mathcal{E}$ with a frozen executor $\mathcal{M}_{\text{exec}}$, we augment the graph with a non-negative edge flow (Bengio et al., 2021; 2023) that is conserved at every non-terminal state, with terminal condition $F(x) = R_\eta(x)$ for the tempered reward $R_\eta(x) = (R(x)+\epsilon)^\eta$. The forward and backward policies are the normalised edge flows, $P_F(e \mid s) = F(s \to e)/F(s)$ and $P_B((s, e) \mid s') = F(s \to e)/F(s')$, and a flow meeting the terminal condition is called *reward-matching*. We seek a state map σ that merges reorderings of independent steps, preserves legal events, successors and terminal reward, and induces a graph admitting a reward-matching flow with terminal marginal proportional to the tempered reward:

$$P_F(x \mid q) \ =\ \frac{R_\eta(x)}{Z^\star(q)}, \qquad Z^\star(q) = \sum_{x \in \mathcal{X}} R_\eta(x), \tag{2}$$

where $Z^\star(q)$ is the true partition function. The flow must also give a per-skill attribution well defined on that graph, to rank the edits turning the phase-k library $\mathcal{S}^{(k)}$ into $\mathcal{S}^{(k+1)}$ (Appendices A and D).

4 Methodology: R^2 Flow

In Figure 3, R^2 Flow nests two self-improvement loops: within a phase (inner loop), the policy improves on its own rollouts with R^{2}TB until the residual variance plateaus; across phases (outer loop), verified edits update the library, and with it the next phase's graph. The graph (Section 4.1) carries a flow trained with R^{2}TB (Section 4.2) whose readouts and residual steer the edits (Section 4.3).

4.1 Orchestration Graph and Task Modeling

R^2 Flow follows a Supervisor–Executor paradigm (Yao et al., 2022b): a trainable Supervisor π_θ interacts with a structured environment $\mathcal{E}$ whose states are **shared states** rather than raw histories.

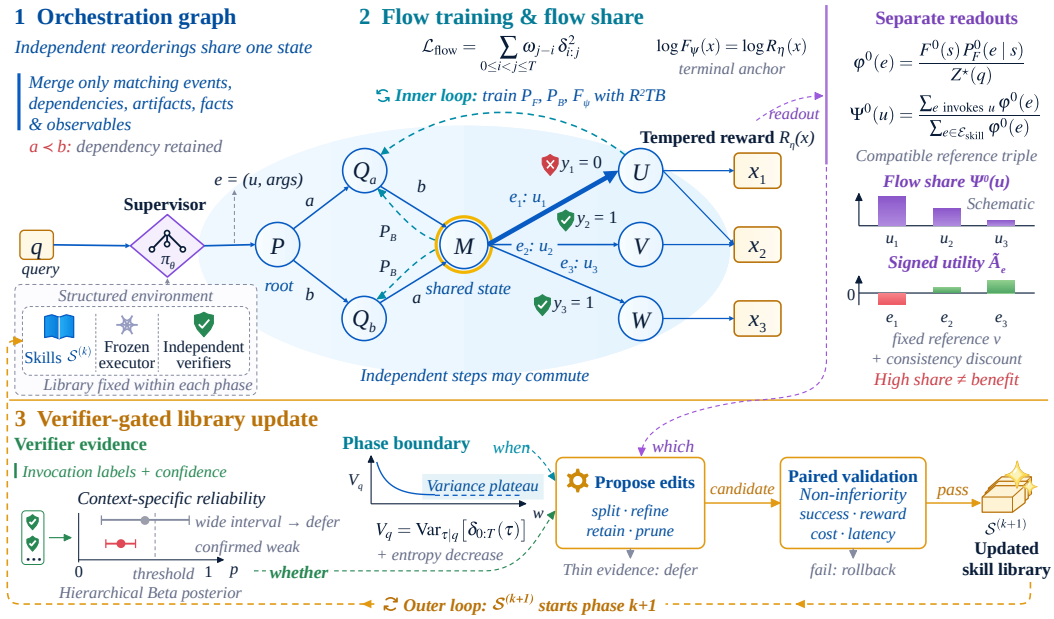


Figure 3: The R^2 Flow framework. (1) The Supervisor π_θ acts in a structured environment where histories differing only in the order of independent steps share one state. (2) P_F, P_B and F_ψ are trained with R^{2}TB; flow share $\Psi^0(u)$ and signed utility $\widetilde{A}_e$ are read separately. (3) Verifier evidence, the two readouts and the variance plateau decide whether, which and when to edit.

Structured Environment $\mathcal{E}$. It holds the skill library, a verifier suite, and the executor:

$$\mathcal{E} = (\mathcal{S},\ \mathcal{Y},\ \mathcal{M}_{\text{exec}}), \tag{3}$$

where $\mathcal{S}$ is the dynamic skill library, $\mathcal{Y}$ a suite of independent verifiers labelling single skill invocations, and $\mathcal{M}_{\text{exec}}$ the pluggable executor. The library changes at phase boundaries, so the action space is fixed within a phase and every trajectory in that phase is sampled against one library version.

Shared States and the Merge Rule. A raw history H_t (Definition 1) maps to its shared state by σ:

$$s_t = \sigma(H_t) = \big(E_t,\ \prec_t,\ \mathcal{A}_t,\ \mathcal{V}_t,\ \Omega_t\big), \tag{4}$$

where E_t is the set of committed events, $\prec_t$ their dependency order, $\mathcal{A}_t$ the canonicalised artifacts, $\mathcal{V}_t$ the verified facts, and Ω_t the environment observables. For example, independent checks A and B can run in either order, so $A \to B$ and $B \to A$ map to one state; if B consumes A's output, the dependency $A \prec B$ is retained and the orders stay distinct (legality conditions on σ in Appendix A).

In-Edges and Rank. A merged state has several in-edges, each from removing a $\prec$-maximal event:

$$\mathrm{In}(s') = \big\{(s, e) : s' = s \oplus e\big\}, \qquad \rho(s) = |E_s|, \tag{5}$$

where ρ is a state's rank; each edge commits an event, so ρ increases along paths and, with the step budget, the graph is a finite DAG. Each predecessor of s' is an in-edge for the backward policy.

Terminal States and Tempered Reward. The terminal state x carries the tempered reward $R_\eta(x)$ of Section 3, whose exponent η sets how sharply sampling favours high reward and whose $\epsilon > 0$ keeps it positive; R is a function of the terminal state, so histories merged into one state share one reward.

Proposition 1. *Merging reorderings yields a shared-state DAG; whenever a reachable state has multiple legal predecessors, the backward policy is a genuine choice among them.* (Empirically, the shared-states and backward-policy ablations and strategy counts, RQ4 in §5; formally, Appendix A.)

4.2 Flow Training and Flow Share

As shown in Figure 3, panel 2, over the shared-state graph from Section 4.1, R^2 Flow trains a flow network $(P_{F,\theta}, P_{B,\phi}, F_\psi)$ targeting the reward-matching flow of Section 3, whose forward policy

$P_{F,\theta}$ is the Supervisor π_θ; throughout, $P_F(e \mid s)$ denotes the one-step edge policy. Beyond mode coverage, the flow serves the edit step: $F(s)$ is shared by every path through s, so where paths merge, the reference flow F^0 gives the readout of Section 4.3, alongside the signed utility below.

Forward Policy P_F and Backward Policy P_B. An event is a skill identifier with structured arguments, so the forward policy factorises, while the backward policy is normalised over in-edges:

$$P_F(e \mid s) = P_F(u \mid s)\, P_F(\text{args} \mid u, s), \qquad P_B\big((s,e) \mid s'\big) = \frac{\exp f_\phi(s,e,s')}{\sum_{(\bar{s},\bar{e})\in \mathrm{In}(s')} \exp f_\phi(\bar{s},\bar{e},s')}, \quad (6)$$

where f_ϕ scores an in-edge from the hindsight state. Each in-edge removes a different $\prec$-maximal event, so P_B spreads training credit over the skills that reach s'. On a history tree every state has one in-edge, so $P_B \equiv 1$ and this credit is never split (Appendix A).

Recursive-Residual Trajectory Balance (R^{2}TB). Given P_F, P_B, a log-flow head $\log F_\psi$ and the reasoning text r_t, a sub-trajectory from s_i to s_j has the balance residual (SubTB; Madan et al., 2023):

$$\delta_{i:j} = \ell(s_i) + \sum_{t=i+1}^{j} \log P_F(e_t \mid s_{t-1}, r_t) - \ell(s_j) - \sum_{t=i+1}^{j} \log P_B\big((s_{t-1}, e_t) \mid s_t\big). \quad (7)$$

The training objective is the weighted squared residual, with the terminal flow anchored to the reward:

$$\mathcal{L}_{\text{flow}} = \sum_{0\le i<j\le T} \omega_{j-i}\, \delta_{i:j}^2, \qquad \ell(s) = \log F_\psi(s) + b_d \ \ (s \notin \mathcal{X}), \quad \ell(x) = \log R_\eta(x), \quad (8)$$

where $F_\psi(s_0) = Z_\theta(q)$ is the learned partition function, ω_{j-i} decays geometrically in $j-i$ (as in SubTB) and is normalised per trajectory, and the per-domain bias b_d absorbs a shared offset of the terminal residuals (Appendix B). Here $j = i+1$ recovers detailed balance and $(i,j) = (0,T)$ trajectory balance, so one loss interpolates between local bootstrapping and trajectory variance, and the target terminal law holds at zero loss (proposition D.3 and corollary D.5). R^{2}TB is sub-trajectory balance on the shared-state graph with the per-domain bias b_d and the reasoning-conditioned forward term, and its residual variance sets the phase boundary of Section 4.3.

Flow Share and Signed Utility. Flow share measures how much of the reward-matching flow passes through each skill: at exact balance, a compatible reference triple (F^0, P_F^0, P_B^0) makes the normalised edge flow the probability of traversing that edge, and a skill's share sums them over its edges:

$$\varphi^0(e) = \frac{F^0(s)\, P_F^0(e \mid s)}{Z^\star(q)}, \qquad \Psi^0(u) = \frac{\sum_{e \text{ invokes } u} \varphi^0(e)}{\sum_{e\in\mathcal{E}_{\text{skill}}} \varphi^0(e)}, \quad (9)$$

where F^0 and P_F^0 are defined by the uniform reference backward policy P_B^0 (Appendix D), which also fixes the edge share $\varphi^0(e)$ used to allocate verification. The denominator sums over skill edges, so the shares of all skills sum to one. Every path to a terminal commits the same events, so $\Psi^0(u)$ is the share of u among skill calls at terminals drawn in proportion to R_η, for any backward policy (Corollary D.9). It is estimated per query from trained-policy rollouts, each weighted by $e^{-\delta_{0:T}}$ normalised within the query, and averaged over queries; the normalisation cancels $\log Z_\theta(q)$ and b_d. Flow share is non-negative by construction and measures how much flow a skill carries; to measure whether it helps, following step-level and counterfactual credit (Zhang et al., 2026b; Lu et al., 2026; Li et al., 2026b), we add a signed utility $\widetilde{A}_e$ contrasted with a fixed reference policy ν:

$$\widetilde{A}_e = \Big(\widehat{\mathbb{E}}[R_\eta \mid s, e] - \mathbb{E}_{e'\sim\nu}\big[\widehat{\mathbb{E}}[R_\eta \mid s, e']\big]\Big) \exp\!\big(-|\delta_e|/\tau_c\big), \quad (10)$$

where $\widehat{\mathbb{E}}[R_\eta \mid s, e]$ estimates the expected tempered reward after taking e at s under a fixed continuation policy (eq. (B.7)), $\nu(\cdot \mid s)$ is uniform over the legal events at s, $\delta_e = \delta_{t-1:t}$ is the single-edge balance violation, and $\tau_c > 0$ scales a discount that weights each estimate by how well the flow balances on that edge. A positive $\widetilde{A}_e$ means the event beats an average legal alternative at s. In Figure 3, skill u_1 carries the most flow while its edge e_1 has negative signed utility.

Proposition 2. *At zero R^{2}TB loss the Supervisor samples terminals in proportion to the tempered reward, and flow share is conserved, independent of the backward policy, consistently estimated from the rollouts of any trained flow, and separate from whether a skill helps.* (Empirically, the reference-readout and utility ablations, RQ4 in §5; formally, Appendices A, B and D.)

4.3 Recursive Skill Evolution with Verifier-Gated Updates

Section 4.2 provides two readouts, the share $\Psi^0(u)$ and the signed utility $\widetilde{A}_e$; both are functions of the reward the library is judged by, so an edit's evidence must come from the verifiers of Section 4.1 rather than that reward. This section (Figure 3, panel 3) combines them with verifier evidence once per phase: verifier labels answer *whether*, the readouts answer *which*, and the residual answers *when*.

Each phase applies one fixed update, repeated until the compute budget is spent. The phase state X_k holds the library, graph, policy and flow parameters and verifier posterior counts, and a log $\mathcal{H}_k^{\mathrm{imp}}$ records each phase's edit decisions and validation outcomes. Under fixed configuration λ, the loop is the recursion $\Xi_{k+1} = \mathfrak{R}_\lambda(\Xi_k; \zeta_k)$ on $\Xi_k = (X_k, \mathcal{H}_k^{\mathrm{imp}})$, where $\mathfrak{R}_\lambda$ runs training, diagnostics, verification and the library operator Φ, and ζ_k is the phase randomness (Proposition C.2).

Whether: Verified Evidence and Credible Bounds. Each invocation carries a context z, and each verified one an independent-verifier outcome y_e with confidence c_e. The verified invocations of skill u in context z, $\mathcal{E}^{\mathrm{ver}}_{u,z}$, update a hierarchical Beta posterior over its reliability in z:

$$\alpha^{\mathrm{ver}}_{u,z} = \kappa_u \mu_u + \sum_{e \in \mathcal{E}^{\mathrm{ver}}_{u,z}} c_e y_e, \qquad \beta^{\mathrm{ver}}_{u,z} = \kappa_u (1-\mu_u) + \sum_{e \in \mathcal{E}^{\mathrm{ver}}_{u,z}} c_e (1-y_e), \tag{11}$$

where μ_u is the skill-level reliability and κ_u the shrinkage strength, pooling sparse contexts toward the skill. Verifier calls are budgeted and go to the invocations with the largest reference edge flow $\varphi^0(e)$ (Remark C.1). Decisions then act on the posterior's lower and upper credible bounds, $\mathrm{LCB}_{u,z}$ and $\mathrm{UCB}_{u,z}$, rather than point estimates. The interval separates thin evidence from confirmed weakness: a wide interval from too few verified records triggers *defer* to collect more observations, whereas $\mathrm{UCB}_{u,z}$ below a quality threshold triggers *refine* or *prune*.

Which: Flow Share and Signed Utility. Both readouts of Section 4.2 enter the library operator together with the credible bounds and the invocation counts:

$$\mathcal{S}^{(k+1)} = \Phi\big(\mathcal{S}^{(k)};\ \{\Psi^0(u), n_u^{\mathrm{call}}, \mathrm{LCB}_{u,z}, \mathrm{UCB}_{u,z}\}_{u \in \mathcal{S}^{(k)}},\ \{\widetilde{A}_e\}_e\big), \tag{12}$$

where n_u^{call} is the empirical invocation count of skill u. The operator applies *defer*, *split*, *refine*, *retain* and *prune* to single skills, *consolidate* to skill pairs, and *generate* to context clusters, in that order (rules in Appendix C). Flow share and signed utility rank the candidate edits. A skill is refined when it is sound overall and confirmed weak in some context, split when its contexts diverge, and pruned when it is confirmed weak in every context, has non-positive signed utility, and every context it covers is covered by another skill. Every edit is versioned and is accepted once paired rollouts are non-inferior on success, tempered reward, token cost, and latency. $\mathcal{S}^{(k+1)}$ changes the graph's edge set (Section 4.1), so the next phase retrains the flow on it and recomputes the readouts and V_q.

When: Residual Variance Plateau. For a fixed query, $\log Z_\theta(q)$ shifts every full-trajectory residual $\delta_{0:T}$ by the same constant, so their variance is free of it (log-partition variance; Zhang et al., 2023):

$$V_q = \mathrm{Var}_{\tau|q}\big[\delta_{0:T}(\tau)\big]. \tag{13}$$

V_q is therefore comparable across training steps on one library version: it falls while the policy learns on the current library, and when it levels off the phase ends and the next verifier-gated proposals are made. Writing $\bar{V}_w$ for V_q aggregated over held-out queries at step w, a phase boundary is declared when the fitted slope of $\bar{V}_w$ is flat within a tolerance, its relative decrease over a recent window is non-negative but small, and the normalised skill entropy has fallen (eq. (C.9)).

Proposition 3. *Every committed skill-library edit is backed by independent verifier evidence and passes paired validation, the readouts of Section 4.2 rank the candidates, and one fixed update maps each phase state to the next.* (Empirically, RQ4 and RQ6 in §5; formally, Appendix C.)

5 Experiments

We test whether R^2 Flow outperforms the baselines of Table 1 in distribution (**RQ1**) and out of distribution (**RQ2**) and transfers across executors (**RQ3**), what each component (**RQ4**) and R^{2}TB (**RQ5**) contribute, and whether it keeps improving with verified edits (**RQ6**).

		Baseline		SFT	GRPO†	AFlow	Agent+RL	Skill evolution		Ours
Dataset	**Metric**	**Qwen3.5**	**v4-flash**	**Qwen3.5**	**Qwen3.5**	**Qwen3.5**	**FlowSteer**	**Trace2Skill**	**SkillOpt**	**R² Flow (Δ ↑)**
(a) In-Distribution (IID) benchmarks										
HotpotQA	Ans EM	57.97 ±0.86	71.41 ±1.05	63.59 ±0.43	69.53 ±0.96	86.41 ±0.70	88.91 ±0.35	88.12 ±0.35	89.53 ±0.43	**91.09** ±0.70 (+33.1)
	Ans F1	73.58 ±0.82	81.92 ±0.43	76.86 ±0.84	81.22 ±0.29	88.97 ±0.18	90.30 ±0.13	90.65 ±0.89	92.47 ±0.10	**93.02** ±0.29 (+19.4)
TriviaQA	Ans EM	44.84 ±0.43	70.62 ±1.18	46.25 ±0.86	47.81 ±0.86	89.53 ±0.70	90.31 ±0.43	89.53 ±0.43	92.50 ±1.18	**95.00** ±0.70 (+50.2)
	Ans F1	53.19 ±0.14	80.75 ±0.99	59.28 ±0.21	58.52 ±0.21	92.50 ±0.86	93.02 ±1.37	91.78 ±0.73	93.71 ±1.25	**96.97** ±1.01 (+43.8)
AIME 2026	Acc.	51.33 ±1.83	48.67 ±1.83	32.00 ±2.98	36.00 ±2.79	53.33 ±2.36	64.00 ±1.49	66.67 ±0.00	67.33 ±2.79	**76.00** ±1.49 (+24.7)
HealthBench	Rubric	33.84 ±0.68	53.43 ±0.89	31.61 ±1.42	36.83 ±0.39	49.32 ±0.57	52.44 ±0.49	53.39 ±0.06	54.53 ±0.73	**61.89** ±0.44 (+28.0)
MBPP+	Pass@1	78.75 ±0.35	84.69 ±0.43	77.97 ±0.35	81.88 ±0.86	85.00 ±1.16	87.50 ±1.24	89.53 ±0.70	89.22 ±0.35	**91.25** ±0.86 (+12.5)
ALFWorld	SR	46.56 ±0.70	60.47 ±0.70	39.53 ±0.70	50.47 ±0.70	74.06 ±0.35	80.31 ±1.02	85.00 ±0.35	84.53 ±0.86	**88.75** ±0.43 (+42.2)
Avg. (IID)	Ans EM	51.41 ±0.48	71.02 ±0.79	54.92 ±0.48	58.67 ±0.64	87.97 ±0.49	89.61 ±0.28	88.83 ±0.28	91.02 ±0.63	**93.05** ±0.49 (+41.6)
	Ans F1	63.38 ±0.42	81.34 ±0.54	68.07 ±0.43	69.87 ±0.18	90.73 ±0.44	91.66 ±0.69	91.22 ±0.58	93.09 ±0.63	**95.00** ±0.53 (+31.6)
	Acc.	52.62 ±0.53	61.81 ±0.55	45.28 ±0.85	51.29 ±0.76	65.43 ±0.68	71.06 ±0.56	73.65 ±0.20	73.90 ±0.76	**79.47** ±0.46 (+26.9)
(b) Out-of-Distribution (OOD) benchmarks										
MuSiQue-Ans	Ans EM	39.06 ±0.55	49.69 ±0.43	38.91 ±0.65	40.62 ±0.55	77.66 ±0.70	79.53 ±1.28	80.94 ±0.70	81.41 ±0.86	**85.31** ±0.35 (+46.2)
	Ans F1	47.07 ±0.53	58.88 ±0.89	50.59 ±1.16	50.35 ±0.60	83.16 ±0.72	85.63 ±0.34	85.02 ±0.61	84.49 ±0.63	**88.33** ±0.97 (+41.3)
NQ-Open	Ans EM	23.28 ±0.65	38.12 ±1.16	25.94 ±0.86	23.91 ±1.05	76.88 ±0.89	79.84 ±0.35	81.88 ±0.35	82.50 ±0.43	**86.25** ±0.70 (+63.0)
	Ans F1	33.29 ±0.81	49.91 ±0.43	37.79 ±0.86	32.59 ±0.98	80.62 ±0.84	84.21 ±0.09	85.28 ±0.35	86.87 ±1.08	**90.16** ±1.25 (+56.9)
MATH-Hard	Acc.	88.44 ±1.16	91.88 ±0.70	88.12 ±0.35	87.50 ±0.00	91.41 ±0.00	93.59 ±0.65	94.84 ±0.70	92.19 ±0.00	**95.94** ±0.86 (+7.5)
GPQA	Acc.	61.09 ±0.35	74.06 ±1.28	68.75 ±1.10	66.88 ±0.70	74.06 ±1.02	83.91 ±0.43	81.88 ±0.65	81.88 ±0.86	**84.84** ±1.18 (+23.8)
SWE-bench	Resolved	16.41 ±0.78	34.84 ±0.70	15.78 ±0.35	18.91 ±0.86	25.16 ±0.86	41.25 ±0.86	40.62 ±1.10	42.97 ±0.78	**44.22** ±1.18 (+27.8)
WebShop	SR	33.28 ±0.43	65.31 ±1.05	32.97 ±1.02	35.62 ±0.43	60.16 ±0.00	79.06 ±1.02	85.78 ±0.86	82.50 ±0.43	**89.53** ±0.89 (+56.2)
Avg. (OOD)	Ans EM	31.17 ±0.43	43.91 ±0.62	32.42 ±0.54	32.27 ±0.59	77.27 ±0.57	79.69 ±0.66	81.41 ±0.39	81.95 ±0.48	**85.78** ±0.39 (+54.6)
	Ans F1	40.18 ±0.48	54.39 ±0.49	44.19 ±0.72	41.47 ±0.57	81.89 ±0.55	84.92 ±0.18	85.15 ±0.35	85.68 ±0.63	**89.25** ±0.79 (+49.1)
	Acc.	49.80 ±0.38	66.52 ±0.48	51.41 ±0.39	52.23 ±0.30	62.70 ±0.33	74.45 ±0.39	75.78 ±0.42	74.88 ±0.31	**78.63** ±0.52 (+28.8)

Table 1: Main results, five runs, mean ± standard deviation. All methods share one frozen Qwen3.5-9B executor; †GRPO trains the Supervisor backbone; Δ ↑ is the gain over Qwen3.5. Averages pool exact match and answer F1 over the question-answering sets and the primary metric over the rest.

Setup. Six in-distribution (IID) datasets train the Supervisor and six out-of-distribution (OOD) datasets are held out (Table 1; Yang et al., 2018; Joshi et al., 2017; Arora et al., 2025; Liu et al., 2023; Shridhar et al., 2020; Trivedi et al., 2022; Kwiatkowski et al., 2019; Hendrycks et al., 2021; Rein et al., 2023; Jimenez et al., 2024; Yao et al., 2022a), with exact match primary for QA. Baselines are the Supervisor backbone Qwen3.5-9B (Team, 2026) and DeepSeek-V4-Flash (Xu et al., 2026a) answering directly, SFT and GRPO (Shao et al., 2024) on the backbone, AFlow (Zhang et al., 2025), FlowSteer (Zhang et al., 2026d), Trace2Skill (Ni et al., 2026) and SkillOpt (Yang et al., 2026); all share the frozen Qwen3.5-9B executor, tools, retrieval corpus and inference budget.

In distribution (RQ1). R^2 Flow leads every IID dataset and metric (Table 1), 2.03 EM and 5.57 accuracy points above SkillOpt, the strongest baseline, with every average gain significant at $p < 10^{-3}$ (Figure 4b). SFT and GRPO† on the backbone fall below Qwen3.5 in accuracy, whereas the R^2 Flow architecture adds 13.71 IID points and training 18.07 more (Table 2). Tuning the backbone on task reward also trades one task for another: GRPO† raises the IID exact-match average by 7.26 points but lowers AIME 2026 from 51.33 to 36.00, while R^2 Flow raises both.

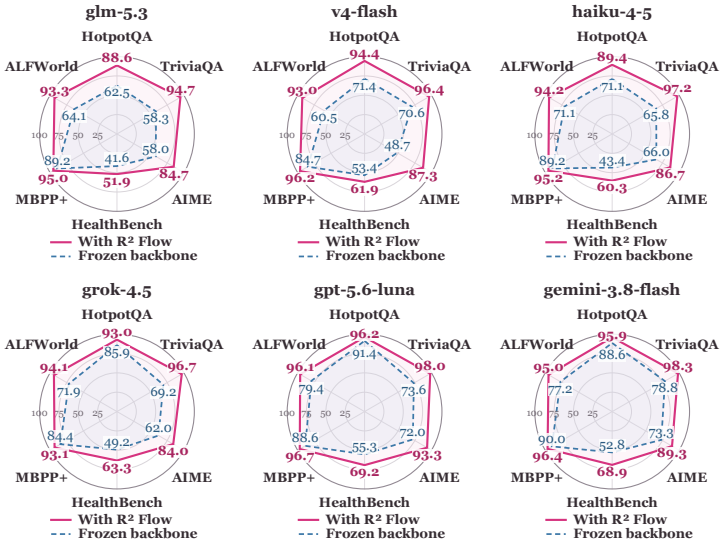
Out of distribution (RQ2). R^2 Flow again leads every OOD row, 3.83 EM and 2.85 accuracy points above the strongest baselines, with larger gains over Qwen3.5 than in distribution and its narrowest margins, about one point, on GPQA and MATH-Hard. Verification matters more here: edits admitted on reward-derived labels cost 5.00 OOD points against 3.34 IID (Table 2), as most of them raise the training reward while lowering the held-out verified score (RQ6). With a 9B Supervisor and executor, R^2 Flow also beats DeepSeek-V4-Flash answering directly on all twelve datasets.

Across executors (RQ3). With the trained Supervisor fixed and six other frozen executors (Figure 4a,d), R^2 Flow raises every backbone on every IID dataset and OOD domain, by 15.70 to 24.64 points on average, and narrows the IID spread across executors from 14.5 to 6.9 points. Weaker executors gain more (Pearson $r = -0.97$) and all gain more out of distribution, consistent with the orchestration and verified skills compensating for the weaker executors.

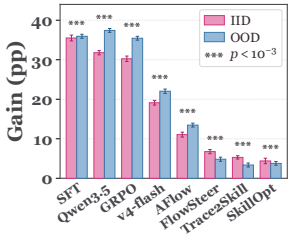
Component ablations (RQ4). All 108 ablated cells of Table 2 fall below the full model. Shared states matter most, 7.31 IID and 5.99 OOD points: on a history tree, reorderings of one procedure

Variant	IID						OOD					
	HotpotQA	TriviaQA	AIME	HealthB.	MBPP+	ALFWorld	MuSiQue	NQ-Open	MATH	GPQA	SWE	WebShop
	Ans EM	Ans EM	Acc.	Rubric	Pass@1	SR	Ans EM	Ans EM	Acc.	Acc.	Resolved	SR
Qwen3.5-9B (frozen)	57.97	44.84	51.33	33.84	78.75	46.56	39.06	23.28	88.44	61.09	16.41	33.28
R^2 Flow architecture, untrained	77.03	77.81	54.00	39.53	83.75	63.44	68.28	61.09	89.84	65.94	22.97	55.31
– Shared states (history tree)	84.06	91.09	62.00	53.75	88.12	81.09	77.19	83.12	93.28	79.06	38.12	79.38
– Learned P_B (uniform P_B)	86.56	91.88	67.33	57.19	88.59	85.31	81.25	84.06	94.84	83.91	40.31	84.84
– Reference readout (read off the learned flow)	88.75	93.59	69.33	59.84	89.53	87.19	83.91	84.69	95.31	84.06	42.50	85.94
– Signed utility (ranked by share alone)	87.34	91.25	70.00	58.28	90.78	84.69	80.78	82.34	94.69	82.97	40.16	82.81
– Verifier labels (reward-derived labels)	87.19	91.25	73.33	57.66	90.94	83.59	78.91	80.78	95.00	82.66	38.59	80.16
– Credible bounds (posterior mean)	89.06	93.91	72.00	58.91	89.84	86.88	81.09	85.31	93.59	82.81	40.47	85.00
– Paired non-inferiority check	88.12	93.28	69.33	59.69	89.69	86.25	83.12	83.44	95.62	83.28	41.56	84.53
– Plateau trigger (fixed phase length)	90.16	93.12	74.67	59.84	90.00	86.72	83.44	83.44	95.31	83.44	42.34	85.94
– Retained phase state (reset each phase)	86.56	89.06	68.00	58.12	89.38	82.97	78.59	78.91	94.69	82.19	39.69	81.41
R^2 Flow (Full)	**91.09**	**95.00**	**76.00**	**61.89**	**91.25**	**88.75**	**85.31**	**86.25**	**95.94**	**84.84**	**44.22**	**89.53**

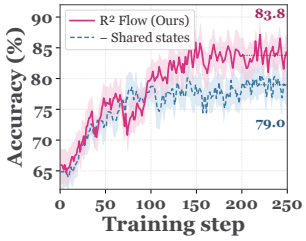
Table 2: Component ablations, five-run mean, protocol of §5. Each row removes one component of R^2 Flow, with everything else fixed. The frozen backbone and R^2 Flow (Full) rows are reproduced from Table 1. The untrained row runs the R^2 Flow architecture with an untrained Supervisor. P_B is the backward policy, and the flow share is read from the compatible reference flow; the signed utility says whether a skill helps; verifier labels, credible bounds and the paired non-inferiority check gate each library edit; a plateau in the residual variance ends a phase, and the flow and verifier posterior carry into the next. Row blocks follow §4.1–4.3: graph, flow training and readouts, skill evolution.



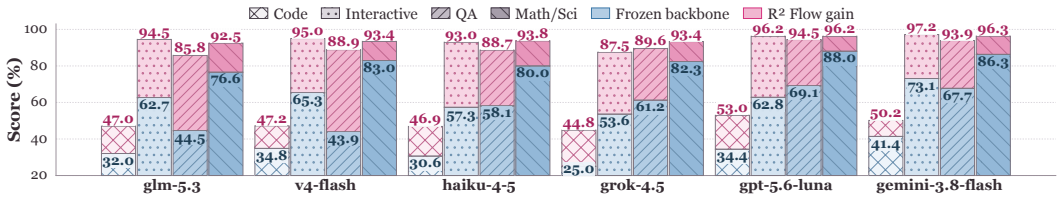
(a) IID scores on each frozen executor



(b) Significance of the gains



(c) Training curve

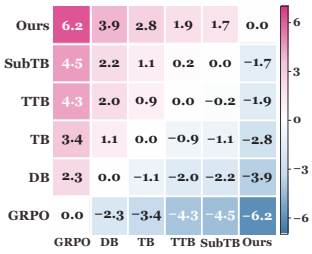


(d) OOD scores aggregated by task domain

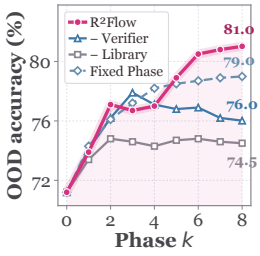
Figure 4: Executor transfer, significance and training dynamics, five-run mean. **(a)** IID scores of six frozen executors answering directly (dashed) and under R^2 Flow with the same trained Supervisor (solid); the radius is linear from 0 to 100. **(b)** Gain of R^2 Flow over each baseline of Table 1, averaged over the six IID or six OOD datasets on each primary metric, with 95% confidence intervals; *** marks $p < 10^{-3}$ (Welch's t-test, five runs). **(c)** Training accuracy of R^2 Flow and of − Shared states over 250 steps; shaded: spread across runs; numbers: mean over the last 50 steps. **(d)** OOD scores averaged per domain: QA (MuSiQue, NQ-Open), Math/Sci (MATH, GPQA), Code (SWE-bench), Interactive (WebShop); each bar stacks the frozen score and the R^2 Flow gain.

Objective	IID score						Learner, per step		Whole step	
	HotpotQA Ans EM	TriviaQA Ans EM	AIME Acc.	HealthB. Rubric	MBPP+ Pass@1	ALFWorld SR	Tokens	GPU s	GPU s	Wall-clock s
GRPO	84.22	92.66	64.67	53.93	86.56	79.06	2,510,218	252	2,406	788
DB	87.03	93.12	65.33	56.93	88.28	82.50	3,121,649	441	2,595	834
TB	88.59	93.91	70.67	57.52	89.69	85.16	3,296,311	389	2,542	822
TTB	88.12	94.06	70.00	58.47	90.16	85.31	3,296,346	390	2,544	823
SubTB	89.38	93.91	69.33	59.24	90.94	87.66	3,121,927	444	2,598	837
R²TB (Ours)	**91.09**	**95.00**	**76.00**	**61.89**	**91.25**	**88.75**	3,121,184	433	2,587	832

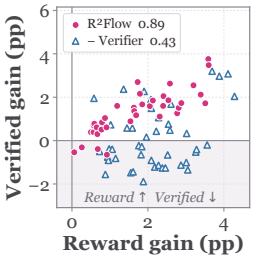
(a) IID scores and training cost per step



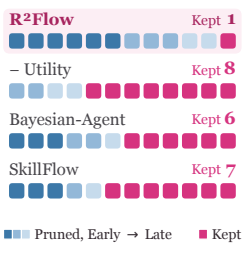
(b) Pairwise OOD differences



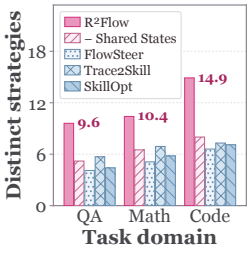
(c) Phase-to-phase gain



(d) Reward vs. verified



(e) Fate of harmful skills



(f) Distinct strategies

Figure 5: Training objectives and recursive self-improvement, five-run mean. **(a)** IID scores and per-step cost of each Supervisor objective (TB, Malkin et al., 2022; DB, Bengio et al., 2023; SubTB without b_d, Madan et al., 2023; TTB, Zhang et al., 2026e). **(b)** Row minus column objective in mean OOD score. **(c)** Mean OOD score per phase; − Verifier uses reward-derived labels, − Library freezes the library, and Fixed Phase replaces the plateau trigger. **(d)** Training-reward gain against verified gain per committed edit; shaded: held-out verified score falls. **(e)** Phase at which each of twelve injected harmful skills is pruned. **(f)** Distinct strategies per domain, reorderings counted once.

no longer pool evidence, so the flow spreads over them and distinct strategies per domain fall from 9.6–14.9 to 5.2–8.0, near the baselines (Figure 5f), and training accuracy falls behind (Figure 4c). For credit, a uniform P_B, which splits credit evenly over the skills reaching a merged state, costs 4.52 IID points, and ranking by share alone, which reads frequent use as benefit, 3.61. Replacing credible bounds by posterior means, which cannot tell thin evidence from confirmed weakness, costs 2.23 IID and 2.97 OOD points, and resetting the phase state, which discards the flow and verifier posterior carried across phases, about five points on both splits. Reading the shares off the learned flow, whose internal flows depend on P_B, costs 2.63 IID and 1.61 OOD points.

Training objectives (RQ5). R²TB leads every IID dataset (Figure 5a) and beats every other objective on OOD by 1.69–6.17 points (Figure 5b), for 7.5% more GPU time than GRPO. SubTB, which balances the same sub-trajectories without the per-domain bias b_d that absorbs a shared offset of each domain's terminal residuals, comes closest. TB and DB, which balance only whole trajectories or single edges, trail it, and GRPO, without a balance residual, is lowest.

Phase-to-phase improvement (RQ6). The mean OOD score rises from 71.2 to 81.02 over eight phases and dips only once (Figure 5c), as each phase trains on the library the previous one committed and 33 of the 37 committed edits raise the held-out verified score (precision 0.89, Figure 5d). With reward-derived labels, 25 of 44 edits raise the training reward but lower the verified score, so accuracy peaks at phase 3 and then declines; a frozen library stalls below 75, and fixed-length phases end at 78.99. Signed utility lets R² Flow prune eleven of twelve injected harmful skills (Figure 5e), six by phase 2 and the rest as evidence accumulates, whereas share-only ranking keeps eight, Bayesian-Agent (Wu et al., 2026) six and SkillFlow (Zhang et al., 2026e) seven.

6 Conclusion

We presented R² Flow, a recursive self-improvement framework that admits library edits on verifier evidence outside the task reward. On twelve datasets it beats every baseline in and out of distribution and improves over eight phases without human intervention, at 0.89 held-out edit precision.

Ethics Statement

This work follows the ICLR Code of Ethics. We report our results without fabrication, falsification or intentional misrepresentation, and Section 5 and Appendix E give the implementation and evaluation details needed to verify them. All twelve datasets are public benchmarks, used for their intended research purposes and under their licenses, and we cite the work and resources we build on. The study involves no human participants and collects no personal information. HealthBench serves only as a benchmark, and R^2 Flow is not intended to give medical advice. Because R^2 Flow revises its own skill library, it commits an edit only on independent verifier evidence and after paired validation.

Reproducibility Statement

Section 5 and Appendix E specify the twelve datasets with their fixed evaluation draws and seed, the metrics, the baselines, the frozen Qwen3.5-9B executor and the validation query set used for phase control. Appendix B gives the R^2TB objective with its weighting and tempering constants, Appendix C gives the verifier-gated evolution rules, the phase trigger and the paired validation, and Appendices A–D state the assumptions and give the proofs behind Propositions 1–3. Every reported number is a mean over five runs. The complete training code, the model weights and the datasets are available at `https://anonymous.4open.science/r/r2flow-DD00`.

AI Use Statement

In this work, we used generative AI tools to improve the readability of the text, to assist the literature survey and format the references, to refine the formal statements and proofs in Appendices A–D, to draft the text that interprets our experimental results, to give feedback on the design of the method and the experiments, and to prepare figures. We have not used generative AI tools to generate data or to produce the reported results. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

References

Rahul K Arora, Jason Wei, Rebecca Soskin Hicks, Preston Bowman, Joaquin Quiñonero-Candela, Foivos Tsimpourlas, Michael Sharman, Meghan Shah, Andrea Vallone, Alex Beutel, et al. Healthbench: Evaluating large language models towards improved human health. *arXiv preprint arXiv:2505.08775*, 2025.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Tong Bai, Zhenglin Wan, Pengfei Zhou, Xingrui Yu, Yang You, and Ivor W Tsang. Skilldag: Self-evolving typed skill graphs for llm skill selection at scale. *arXiv preprint arXiv:2606.03056*, 2026.

Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. *Advances in neural information processing systems*, 34:27381–27394, 2021.

Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J Hu, Mo Tiwari, and Emmanuel Bengio. Gflownet foundations. *Journal of Machine Learning Research*, 24(210):1–55, 2023.

Pier Giovanni Bissiri, Chris C Holmes, and Stephen G Walker. A general framework for updating belief distributions. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 78 (5):1103–1130, 2016.

Kunfeng Chen, Qihuang Zhong, Juhua Liu, and Bo Du. Skillcat: Contrastive assessment and topology-aware skill self-evolution for llm agents. *arXiv preprint arXiv:2606.13317*, 2026a.

Mingguang Chen, Licheng Wang, and Bo Qu. Recursive self-improvement in ai: From bounded self-refinement to autonomous research loops. *arXiv preprint arXiv:2607.07663*, 2026b.

Jinyuan Fang, Yanwen Peng, Xi Zhang, Yingxu Wang, Xinhao Yi, Guibin Zhang, Yi Xu, Bin Wu, Siwei Liu, Zihao Li, et al. A comprehensive survey of self-evolving ai agents: A new paradigm bridging foundation models and lifelong agentic systems. *arXiv preprint arXiv:2508.07407*, 2025.

Jake Fawkes and Jason Hartford. f-trajectory balance: A loss family for tuning gflownets, generative models, and llms with off-and on-policy data. *arXiv preprint arXiv:2605.15417*, 2026.

Zhongxin Guo, Danrui Qi, Hanwen Gu, Peng Cheng, and Yongqiang Xiong. Skill-disco: Distilling and compiling agent traces into reusable procedural skills. *arXiv preprint arXiv:2606.26669*, 2026.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, volume 2025, pp. 21344–21377, 2025.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? In *International Conference on Learning Representations*, volume 2024, pp. 54107–54157, 2024.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, 2017.

Leslie Kish. Survey sampling. 1965.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 6086–6096, 2019.

Ao Li, Shangpeng Yang, Fahao Chen, Tianheng Xu, Peng Li, and Zhou Su. Graphflow: A graph-based workflow management for efficient llm-agent serving. *arXiv preprint arXiv:2605.22566*, 2026a.

Lihong Li, Thomas J Walsh, and Michael L Littman. Towards a unified theory of state abstraction for mdps. *AI&M*, 1(2):3, 2006.

Yubo Li. Dynamic agent skills: A lifecycle survey and taxonomy of evolving skill libraries. *arXiv preprint arXiv:2607.10113*, 2026.

Zhongyi Li, Wan Tian, Jinju Chen, Huiming Zhang, Yang Liu, Yikun Ban, and Fuzhen Zhuang. Counterfactual credit policy optimization for multi-agent collaboration. *arXiv preprint arXiv:2603.21563*, 2026b.

Huawei Lin, Peng Li, Jie Song, Fuxin Jiang, and Tieying Zhang. Muse-autoskill: Self-evolving agents via skill creation, memory, management, and evaluation. *arXiv preprint arXiv:2605.27366*, 2026.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in neural information processing systems*, 36:21558–21572, 2023.

Mingyu Lu, Yushan Huang, Chris Lin, and Su-In Lee. Agents that matter: Optimizing multi-agent llms via removal-based attribution. *arXiv preprint arXiv:2605.27621*, 2026.

Kanika Madan, Jarrid Rector-Brooks, Maksym Korablyov, Emmanuel Bengio, Moksh Jain, Andrei Cristian Nica, Tom Bosc, Yoshua Bengio, and Nikolay Malkin. Learning gflownets from partial episodes for improved convergence and stability. In *International Conference on Machine Learning*, pp. 23467–23483. PMLR, 2023.

Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in gflownets. *Advances in Neural Information Processing Systems*, 35: 5955–5967, 2022.

Antoni Mazurkiewicz. Trace theory. In *Advanced course on Petri nets*, pp. 278–324. Springer, 1986.

Jingwei Ni, Yihao Liu, Xinpeng Liu, Yutao Sun, Mengyu Zhou, Pengyu Cheng, Dexin Wang, Erchao Zhao, Xiaoxi Jiang, and Guanjun Jiang. Trace2skill: Distill trajectory-local lessons into transferable agent skills. *arXiv preprint arXiv:2603.25158*, 2026.

Siru Ouyang, Jun Yan, Yanfei Chen, Rujun Han, Zifeng Wang, Bhavana Dalvi Mishra, Rui Meng, Chun-Liang Li, Yizhu Jiao, Kaiwen Zha, et al. Skillos: Learning skill curation for self-evolving agents. *arXiv preprint arXiv:2605.06614*, 2026.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Zhe Ren, Yimeng Chen, Dandan Guo, Guowei Rong, Tonghui Li, RB Xiong, Qingfeng Lan, Wenyi Wang, Li Nanbo, Yibo Yang, et al. Self-improvements in modern agentic systems: A survey. *arXiv preprint arXiv:2607.13104*, 2026.

Donald J Schuirmann. A comparison of the two one-sided tests procedure and the power approach for assessing the equivalence of average bioavailability. *Journal of pharmacokinetics and biopharmaceutics*, 15(6):657–680, 1987.

Moshe Shaked and J George Shanthikumar. *Stochastic orders*. Springer, 2007.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.

Hui-Ze Tan, Xiao-Wen Yang, Hao Chen, Jie-Jing Shao, Yi Wen, Yuteng Shen, Weihong Luo, Xiku Du, Lan-Zhe Guo, and Yu-Feng Li. Hindsight credit assignment for long-horizon llm agents. *arXiv preprint arXiv:2603.08754*, 2026.

Liyan Tang, Cyrus Rashtchian, Chun-Sung Ferng, Andrew Tomkins, Da-Cheng Juan, and Tu Vu. Wikiskill: Compiling agent experience into persistent knowledge for skill evolution. *arXiv preprint arXiv:2608.27454*, 2026.

Leitian Tao, Baolin Peng, Wenlin Yao, Tao Ge, Hao Cheng, Mike Hang Wang, Jianfeng Gao, and Sharon Li. Trace: Turn-level reward assignment via credit estimation for long-horizon agents. *arXiv preprint arXiv:2607.13988*, 2026.

Qwen Team. Qwen3. 5-omni technical report. *arXiv preprint arXiv:2604.15804*, 2026.

Kunvar Thaman. Reward hacking benchmark: measuring exploits in llm agents with tool use. *arXiv preprint arXiv:2605.02964*, 2026.

Katherine Tieu, Dongqi Fu, Yinglong Xia, Hong Li, Hong Yan, and Jingrui He. Inference-time graph engineering for multi-agent llm workflows. *arXiv preprint arXiv:2609.05774*, 2026.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. ♪ musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

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. *arXiv preprint arXiv:2305.16291*, 2023.

Xi Wang, Wenbo Lu, and Shengjie Wang. Rooted absorbed prefix trajectory balance with submodular replay for gflownet training. *arXiv preprint arXiv:2603.00454*, 2026.

Xiaojun Wu, Cehao Yang, Honghao Liu, Xueyuan Lin, Wenjie Zhang, Zhichao Shi, Xuhui Jiang, Chengjin Xu, Jia Li, and Jian Guo. Bayesian-agent: Posterior-guided skill evolution for llm agent harnesses. *arXiv preprint arXiv:2606.08348*, 2026.

Peng Xia, Jianwen Chen, Hanyang Wang, Jiaqi Liu, Kaide Zeng, Yu Wang, Siwei Han, Yiyang Zhou, Xujiang Zhao, Haifeng Chen, et al. Skillrl: Evolving agents via recursive skill-augmented reinforcement learning. *arXiv preprint arXiv:2602.08234*, 2026.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026a.

Ruiyao Xu, Tiankai Yang, and Wei-Chieh Huang. Hyperskill: Self-evolving llm agents via hypergraph-structured skill memory. *arXiv preprint arXiv:2608.16114*, 2026b.

Yifan Yang, Ziyang Gong, Weiquan Huang, Qihao Yang, Ziwei Zhou, Zisu Huang, Yan Li, Xuemei Gao, Qi Dai, Bei Liu, et al. Skillopt: Executive strategy for self-evolving agent skills. *arXiv preprint arXiv:2605.23904*, 2026.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022a.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022b.

Xunjian Yin, Xinyi Wang, Liangming Pan, Li Lin, Xiaojun Wan, and William Yang Wang. Gödel agent: A self-referential agent framework for recursively self-improvement. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27890–27913, 2025.

Fangxu Yu, Lai Jiang, Haoqiang Kang, Shibo Hao, and Lianhui Qin. Flow of reasoning: Training llms for divergent reasoning with minimal examples. *arXiv preprint arXiv:2406.05673*, 2024.

Ling Yue, Kushal Raj Bhandari, Ching-Yun Ko, Dhaval Patel, Shuxin Lin, Nianjun Zhou, Jianxi Gao, Pin-Yu Chen, and Shaowu Pan. From static templates to dynamic runtime graphs: a survey of workflow optimization for llm agents. *arXiv preprint arXiv:2603.22386*, 2026.

Sukwon Yun, Jie Peng, Pingzhi Li, Wendong Fan, Jie Chen, James Y Zou, Guohao Li, and Tianlong Chen. Graph-of-agents: A graph-based framework for multi-agent llm collaboration. In *International Conference on Learning Representations*, volume 2026, pp. 19745–19760, 2026.

Eric Zelikman, Eliana Lorch, Lester Mackey, and Adam Tauman Kalai. Self-taught optimizer (stop): Recursively self-improving code generation. *arXiv preprint arXiv:2310.02304*, 2023.

Chenchen Zhang. From reasoning to agentic: Credit assignment in reinforcement learning for large language models. *arXiv preprint arXiv:2604.09459*, 2026a.

Chenchen Zhang. Reinforcement learning for llm-based multi-agent systems through orchestration traces. *arXiv preprint arXiv:2605.02801*, 2026b.

David W Zhang, Corrado Rainone, Markus Peschl, and Roberto Bondesan. Robust scheduling with gflownets. *arXiv preprint arXiv:2302.05446*, 2023.

Haiyue Zhang. Credit without ground truth: Auditing step-level credit assignment in llm agents against executed replay. *arXiv preprint arXiv:2608.19760*, 2026c.

Hanrong Zhang, Shicheng Fan, Henry Peng Zou, Yankai Chen, Zhenting Wang, Jiayu Zhou, Chengze Li, Wei-Chieh Huang, Yifei Yao, Kening Zheng, et al. Coevoskills: Self-evolving agent skills via co-evolutionary verification. *arXiv preprint arXiv:2604.01687*, 2026a.

Huan Zhang, Mingju Chen, Dongxu Zhou, Can Lv, Heng Chang, Sen Cui, Faguo Wu, and Shiji Zhou. Trca: Transition-wise rubric credit assignment for long-horizon llm agents. *arXiv preprint arXiv:2608.16156*, 2026b.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin gödel machine: open-ended evolution of self-improving agents. In *International Conference on Learning Representations*, volume 2026, pp. 104223–104294, 2026c.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, et al. Aflow: Automating agentic workflow generation. In *International Conference on Learning Representations*, volume 2025, pp. 34040–34077, 2025.

Mingda Zhang, Wenjin Liu, Tiesunlong Shen, Qika Lin, Rui Mao, Erik Cambria, Xiaoying Tang, and Haoran Luo. Flowsteer: Towards agents designing agentic workflows via reinforced progressive canvas editing. *arXiv preprint arXiv:2602.01664*, 2026d.

Mingda Zhang, Tiesunlong Shen, Haoran Luo, Wenjin Liu, Zikai Xiao, Erik Cambria, and Xiaoying Tang. Skillflow: Flow-driven recursive skill evolution for agentic orchestration. *arXiv preprint arXiv:2605.14089*, 2026e.

Xing Zhang, Yanwei Cui, Guanghui Wang, Ziyuan Li, Wei Qiu, Bing Zhu, and Peiyang He. Library drift: Diagnosing and fixing a silent failure mode in self-evolving llm skill libraries. *arXiv preprint arXiv:2605.19576*, 2026f.

Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. Language agents as optimizable graphs. *arXiv preprint arXiv:2402.16823*, 2024.

A Formal Foundations of the Orchestration Graph

A.1 Quotient Orchestration Graph

Raw histories. An execution history evolves by appending atomic records,

$$H_t \;=\; H_{t-1} \oplus (r_t, e_t, o_t^{\text{exec}}), \tag{A.1}$$

where r_t is internal reasoning text, e_t the executed orchestration event (a skill or tool invocation with structured arguments), and o_t^{exec} the environment feedback. Given a query q and a retrieved skill subset $\mathcal{S}_{\text{ret}} \subseteq \mathcal{S}^{(k)}$ together with a task-conditioned prompt state ω_q, the root history is $H_0 = q \oplus \mathcal{S}_{\text{ret}} \oplus \omega_q$, to which eq. (A.1) appends the first atomic record.

Remark A.1 (Canonicalisation creates informative backward structure). If nodes are identified with histories, every node except the root has exactly one parent, namely H_t with its last record deleted. The resulting object is a prefix tree, not a confluent DAG, and the backward transition probability is degenerate, $P_B(H_{t-1} \mid H_t) \equiv 1$. Canonicalisation merges reorderings of independent events into shared states, turning the prefix tree of raw histories into a DAG with multiple valid incoming edges; the resulting backward transition therefore carries information about how a state was reached.

Definition A.1 (Execution trace and dependency order)**.** Let E_t be the set of atomic event instances committed after t steps. The direct dependency relation records when an event consumes an artifact, fact, or environment change produced by another event; $\prec_t$ denotes its transitive closure, so it is a partial order. Repeated calls with the same skill and arguments are distinct event instances, while their provenance is retained in the canonical state. Let $\mathcal{A}_t$ be the canonicalised artifacts with provenance, $\mathcal{V}_t$ the verified facts, and Ω_t the environment observables required for future decisions. The abstraction map, whose values are the shared states of Section 4.1, is

$$s_t \;=\; \sigma(H_t) \;=\; \big(E_t,\ \prec_t,\ \mathcal{A}_t,\ \mathcal{V}_t,\ \Omega_t\big). \tag{A.2}$$

Two histories are identified iff they are distinct linear extensions of the same dependency order with identical artifacts, facts, and observables; this is the Mazurkiewicz trace equivalence of concurrency theory (Mazurkiewicz, 1986) (trace in the sense of a partial-order equivalence class of executions, not an orchestration trace in the sense used by Zhang (2026b)).

Assumption A.1 (Quotient validity)**.** For all H, H' with $\sigma(H) = \sigma(H') = s$:

(a) *action-mask consistency*: $\mathcal{E}(H) = \mathcal{E}(H')$, the legal event sets coincide;

(b) *successor consistency*: $\mathbb{P}\big(\sigma(H \oplus e) = s' \mid H, e\big) = \mathbb{P}\big(\sigma(H' \oplus e) = s' \mid H', e\big)$ for all legal e;

(c) *terminal-reward consistency*: if s is terminal then $R(H) = R(H')$, i.e. the reward is measurable with respect to the terminal *node* s shared by H and H'.

Assumption A.2 (Markov policy and flow)**.** The backward policy and state flow read only the abstract state, and the forward policy reads it together with the current reasoning text r_t: the first two are functions of (s, q, k) and the forward policy of (s, q, k, r_t), none of the discarded history (appendix C.3). Reasoning text r_t influences future behaviour only through the canonicalised content it commits to the artifacts $\mathcal{A}_t$ or the verified facts $\mathcal{V}_t$.

Assumption A.3 (Finite event horizon and branching)**.** There is a finite horizon $T < \infty$, and every nonterminal abstract state has a finite nonempty set of legal events. Terminal states have no outgoing events. The horizon and event set are fixed within a library version; a phase update may change them only at a phase boundary declared by the trigger of eq. (C.9).

Assumption A.4 (Canonicalised transitions)**.** After canonicalisation, the successor state is a deterministic function of (s, e). Tools with stochastic canonicalised output are modelled by the event/environment split $s \to (s, e) \to s'$ into an event step and an environment step.

Definition A.2 (Edges, in-edges, rank)**.** An edge $s \xrightarrow{e} s'$ exists iff e is legal at s and σ-appending e yields s'. The in-edge set is $\mathrm{In}(s') = \{(s, e) : s \xrightarrow{e} s'\}$, and its predecessor-state projection is:

$$\mathrm{Pa}(s') \;=\; \big\{\, s : \exists e \text{ with } s \xrightarrow{e} s' \,\big\}. \tag{A.3}$$

Each edge commits one atomic event, so each in-edge deletes one $\prec$-maximal event whose removal leaves a legal predecessor. The rank is $\rho(s) = |E_s|$, the number of committed atomic events.

Proposition A.1 (Well-posedness)**.** *Every edge strictly increases ρ; hence the quotient graph is a finite DAG with a single source $s_0 = \sigma(H_0)$, and every complete trajectory visits states of strictly increasing rank ρ, the number of committed atomic events.*

Proof. Each edge commits at least one atomic event, so $\rho(s') \geq \rho(s)+1 > \rho(s)$. A strictly monotone integer potential forbids cycles. By Assumption A.3, every path has at most T events and every node has finite branching, so the reachable quotient graph is finite. □

Terminal states and tempered reward. Let $\mathcal{X}$ be the set of terminal nodes. By Assumption A.1(c) the reward is a function of the terminal node, and we temper and shift it as

$$R_\eta(x) \;=\; \big(R(x) + \epsilon\big)^\eta, \qquad R(x) \in [0,1],\ \epsilon > 0,\ \eta > 0. \tag{A.4}$$

Proposition A.2 (Representation multiplicity)**.** *Suppose an observable outcome o is represented by $k(o)$ distinct terminal nodes, each anchored to $R_\eta(o)$. Then at any exact balance solution*

$$P_F^{\mathrm{term}}(o \mid q) \;=\; \frac{k(o)\, R_\eta(o)}{Z^\star(q)}, \tag{A.5}$$

i.e. outcomes are over-sampled in proportion to how many ways they can be represented.

Proof. At the exact solution $P_F^{\mathrm{term}}(x \mid q) = R_\eta(x)/Z^\star(q)$ for each terminal node x (proposition A.3). Summing over the $k(o)$ nodes representing o gives eq. (A.5). □

Library versioning. A skill-library update changes the legal event set and therefore the graph itself. We write $\mathcal{G}^{(k)}$ for the graph of library version k and condition all learned objects on the version index, the policies $P_F(\cdot \mid s, q, k)$ and $P_B(\cdot \mid s', q, k)$, and the flow $F_\psi(s, q, k)$.

A.2 Forward and Backward Policies

Semantic action factorisation. An event is a pair $e = (u, \mathrm{args})$ of a skill/tool identifier $u \in \mathcal{S}^{(k)}$ and structured arguments, so that

$$P_F(e \mid s, q, k) \;=\; P_F(u \mid s, q, k)\, P_F(\mathrm{args} \mid u, s, q, k). \tag{A.6}$$

Remark A.2 (Token-length normalisation). Because each event is modelled directly as a skill identifier and structured arguments, eq. (A.6) defines a normalised probability over executable events without token-length correction, and a policy used consistently for sampling and for balance evaluation satisfies the balance identities encoded by the residual in eq. (B.1).

Backward policy. The backward policy is a distribution over *in-edges*, parameterised with hindsight features and normalised explicitly:

$$P_B\big((s,e) \mid s'\big) \;=\; \frac{\exp f_\phi(s, e, s')}{\displaystyle\sum_{(\bar{s},\bar{e}) \in \mathrm{In}(s')} \exp f_\phi(\bar{s}, \bar{e}, s')}. \tag{A.7}$$

Definition A.3 (Canonical backward policy)**.** $P_B^0\big((s,e) \mid s'\big) = 1/|\,\mathrm{In}(s')|$: the valid in-edges are weighted uniformly *locally*. This uniform weighting is local, so complete linearisations of a trace can have different probabilities under the canonical backward policy P_B^0.

Proposition A.3 (Two faces of P_B)**.** *Let $Z^\star(q) = \sum_{x \in \mathcal{X}} R_\eta(x)$ and suppose balance holds exactly.*

(i) (Invariance.) *The terminal marginal satisfies $P_F^{\mathrm{term}}(x \mid q) = R_\eta(x)/Z^\star(q)$ for every choice of normalised backward policy P_B with full support.*

(ii) (Dependence.) *The internal state flows $F(s)$ do depend on P_B.*

Proof. (i) Exact balance gives $P_F(\tau) = R_\eta(x_\tau) P_B(\tau \mid x_\tau)/Z^\star(q)$. Summing over trajectories reaching a fixed x and using $\sum_{\tau \to x} P_B(\tau \mid x) = 1$ yields $P_F^{\mathrm{term}}(x \mid q) = R_\eta(x)/Z^\star(q)$. (ii) Take a single terminal x reached by two length-two paths through distinct states $\hat{u}$ and $\hat{v}$. Then $F(\hat{u})/F(\hat{v}) = P_B(\hat{u}\text{-path} \mid x)/P_B(\hat{v}\text{-path} \mid x)$, which equals 9 for $P_B = (0.9, 0.1)$ and 1 for P_B^0, while the terminal marginal is identical in both cases. □

Remark A.3 (Consequence for credit). Terminal reward matching does not pin down internal flow: edge and state flows read off F depend on the backward policy, whereas the skill share Ψ^0 does not (corollary D.9). We keep training and readout separate: the flow is trained with the learned $P_{B,\phi}$, and the reference readout is defined by a compatible triple (F^0, P_F^0, P_B^0) and estimated from the trained policy's rollouts with the residual weights of corollary D.9.

B Balance Objectives and Flow Diagnostics

B.1 The R^{2}TB Objective

We learn a forward policy P_F, a backward policy P_B, and an explicit log-flow head $\log F_\psi(s)$. For $0 \le i < j \le T$ define the residual

$$\delta_{i:j} \;=\; \log F_\psi(s_i) + \sum_{t=i+1}^{j} \log P_F(e_t \mid s_{t-1}) - \log F_\psi(s_j) - \sum_{t=i+1}^{j} \log P_B\big((s_{t-1}, e_t) \mid s_t\big), \tag{B.1}$$

with the terminal anchor and source identification

$$\log F_\psi(x) \;=\; \log R_\eta(x) \quad \forall x \in \mathcal{X}, \qquad F_\psi(s_0) \;=\; Z_\theta(q). \tag{B.2}$$

The objective is

$$\mathcal{L}_{\text{flow}} \;=\; \sum_{0 \le i < j \le T} \omega_{j-i}\, \delta_{i:j}^2, \tag{B.3}$$

where $j = i + 1$ recovers detailed balance, $(i, j) = (0, T)$ recovers trajectory balance, and intermediate lengths interpolate between local bootstrapping and whole-trajectory variance. R^{2}TB is this objective with the domain offset of remark B.1 and the r_t-conditioned P_F of appendix C.3.

Remark B.1 (Domain bias). For a query of domain d the implementation adds one scalar b_d to $\log Z_\theta(q)$ and to every intermediate $\log F_\psi(s)$, and keeps the terminal anchor at $\log R_\eta(x)$. The offset cancels in every residual whose endpoints are both nonterminal and shifts every residual that ends at a terminal by b_d. Since the flow head is conditioned on q, b_d keeps the zero-loss solutions of proposition D.3 and absorbs the shared domain offset during training. It is updated toward a closed-form target: after each step it moves by 0.3 times the median, over the domain's trajectories, of the shift $c^\star = -\sum_i \omega_{T-i}\delta_{i:T} / \sum_i \omega_{T-i}$ that minimises each trajectory's loss, and it is carried to the next phase together with ψ. At zero loss every $c^\star$ vanishes, so b_d is stationary. The within-query variance V_q and the reference share $\Psi^0(u)$ do not depend on b_d. The implementation uses $\omega_{j-i} \propto 0.9^{\,j-i}$ normalised per trajectory, $\eta = 4$, and $\epsilon = 0.1$.

Remark B.2 (Which residual to report). $\log Z_\theta(q)$ enters $\delta_{0:T}$ as a constant shift for a fixed q, but $\log Z_\theta(q)/T$ does not when trajectory lengths vary. All diagnostics in appendix C.2 therefore use the raw residual $\delta_{0:T}$ of eq. (B.1), never a length-normalised variant.

B.2 Flow Diagnostics

Proposition B.1 (What the state flow is)**.** *Let $h_B(s \mid x)$ be the probability that backward sampling from terminal x under P_B passes through s. At exact balance,*

$$F(s) \;=\; \sum_{x \in \mathcal{X}} R_\eta(x)\, h_B(s \mid x), \qquad F(s_0) = Z^\star(q), \qquad F(x) = R_\eta(x). \tag{B.4}$$

Proof. Trajectory flow at the optimum is $F(\tau) = Z^\star P_F(\tau) = R_\eta(x_\tau) P_B(\tau \mid x_\tau)$. Summing over trajectories through s and grouping by terminal state gives eq. (B.4). □

Remark B.3 (The state flow as downstream reward mass). The flow through s that terminates at x equals both $F(s)P_F(x \mid s)$ and $R_\eta(x)h_B(s \mid x)$, so $P_F(x \mid s) = R_\eta(x)h_B(s \mid x)/F(s)$ and

$$\mathbb{E}_{P_F}\big[R_\eta \mid s\big] \;=\; \frac{1}{F(s)} \sum_x R_\eta(x)^2\, h_B(s \mid x). \tag{B.5}$$

The second moment appears, so $F(s) \not\propto \mathbb{E}[R_\eta \mid s]$ in general. F is a partition function over the downstream reward mass of s and $\log F$ a free energy.

Importance: normalised flow share. Define the normalized expected invocation mass routed through an edge and through a skill under the reward-matching reference law,

$$\varphi^0(e) \;=\; \frac{F^0(s)\,P_F^0(e \mid s)}{Z^\star(q)}, \qquad \Psi^0(u) \;=\; \frac{\sum_{e \text{ invoking } u} \varphi^0(e)}{\sum_{e \in \mathcal{E}_{\text{skill}}} \varphi^0(e)}. \tag{B.6}$$

Proposition B.2 (Conservation)**.** *If $\mathcal{C}$ is a cut, i.e. every complete trajectory visits exactly one state of $\mathcal{C}$, then $\sum_{s\in\mathcal{C}} F^0(s) = Z^\star(q)$. For every nonterminal state s, $\sum_{e\in\mathrm{Out}(s)} \varphi^0(e) = F^0(s)/Z^\star(q)$. If $\mathcal{E}_{\text{skill}}$ is the set of counted edges and every such edge invokes exactly one skill, then $\sum_u \Psi^0(u) = 1$ when the denominator of Ψ^0 is restricted to $\mathcal{E}_{\text{skill}}$.*

Proof. For each terminal-directed trajectory τ, let its reference trajectory mass be $m^0(\tau) = F^0(s_0)P_F^0(\tau)$. Exact balance and the terminal anchor give $\sum_\tau m^0(\tau) = Z^\star(q)$. Since the cut is visited exactly once, partitioning the trajectories by their unique cut state yields $\sum_{s\in\mathcal{C}} F^0(s) = \sum_\tau m^0(\tau) = Z^\star(q)$. For a fixed state, $F^0(s) = \sum_{e\in\mathrm{Out}(s)} F^0(s)P_F^0(e \mid s)$, so dividing by $Z^\star(q)$ gives $\sum_{e\in\mathrm{Out}(s)} \varphi^0(e) = F^0(s)/Z^\star(q)$. Finally, if each counted skill edge has one and only one skill label, the skill sums form a partition of $\sum_{e\in\mathcal{E}_{\text{skill}}} \varphi^0(e)$; after dividing by this same positive denominator, $\sum_u \Psi^0(u) = 1$, the normalisation of the skill share in eq. (B.6). □

Utility: a separate, signed quantity. For utility we compare an event against a *fixed* reference distribution ν, uniform over the legal events at s,

$$A_e^{\text{util}} \;=\; \widehat{\mathbb{E}}\big[R_\eta \mid s, e\big] - \mathbb{E}_{e'\sim\nu(\cdot|s)}\big[\widehat{\mathbb{E}}[R_\eta \mid s, e']\big], \tag{B.7}$$

and discount it by the edge-level balance violation $\delta_e = \delta_{t-1:t}$,

$$\widetilde{A}_e \;=\; A_e^{\text{util}}\,\exp\!\big(-|\delta_e|/\tau_c\big). \tag{B.8}$$

Here $\widehat{\mathbb{E}}[R_\eta \mid s, e]$ estimates $Q^{\pi_{\text{eval}}}(s, e)$ under a fixed continuation policy π_{eval}.

C Evidence, Gating, and Skill Evolution

C.1 Verifier-Gated Bayesian Calibration

Each invocation carries a context feature vector

$$z = \big[\text{context class},\ \text{failure mode},\ \text{token bucket},\ \text{horizon bucket}\big]. \tag{C.1}$$

Context features use only information available before the invocation; a failure mode observed afterwards enters later invocations as a lagged feature.

Evidence. Labels come from independent verifiers (unit tests, schema validation, assertions, execution status) with confidence $c_e^{\text{ver}} \in [0, 1]$. For each skill/context cell define the verifier-record index set, which collects the verifier records of skill u observed in context z,

$$\mathcal{E}_{u,z}^{\text{ver}} = \{e \in \mathcal{E}^{\text{ver}} :\ u(e) = u,\ z(e) = z\}. \tag{C.2}$$

Only these verifier records enter the gate posterior:

$$\alpha_{u,z}^{\text{ver}} = \alpha_{0,u} + \sum_{e\in\mathcal{E}_{u,z}^{\text{ver}}} c_e^{\text{ver}} y_e, \tag{C.3}$$

$$\beta_{u,z}^{\text{ver}} = \beta_{0,u} + \sum_{e\in\mathcal{E}_{u,z}^{\text{ver}}} c_e^{\text{ver}} (1 - y_e). \tag{C.4}$$

A soft label $\widetilde{y}_e \in [0, 1]$ derived from $\widetilde{A}_e$ enters a separate proposal score $q_{u,z}^{\text{soft}}$ that orders candidates and stays outside the gate posterior $(\alpha^{\text{ver}}, \beta^{\text{ver}})$ of eqs. (C.3) and (C.4).

Remark C.1 (Flow allocates the verification budget). $\varphi^0(e)$ decides *which* invocations are worth verifying; verifier confidence decides *how much* an observation counts as evidence. Keeping the two roles separate is what makes the posterior interpretable.

Remark C.2 (Effective sample size). With fractional weights, the quantity that behaves like a sample count is Kish's effective sample size (Kish, 1965),

$$N_{\text{eff}} = \frac{\left(\sum_e w_e\right)^2}{\sum_e w_e^2} \quad \text{(zero without records)}, \tag{C.5}$$

not $\sum_e w_e$: one weight of size N with the rest near zero has $\sum_e w_e = N$ but $N_{\text{eff}} \approx 1$.

Hierarchical prior and credible bounds. Contexts are pooled towards the skill-level reliability μ_u with strength κ_u, using the same prior parameters as the verifier gate, $\alpha_{0,u} = \kappa_u \mu_u$ and $\beta_{0,u} = \kappa_u(1-\mu_u)$:

$$p_{u,z} \;\sim\; \text{Beta}\big(\kappa_u\mu_u,\; \kappa_u(1-\mu_u)\big), \tag{C.6}$$

and decisions use Beta quantiles rather than a normal approximation:

$$\text{LCB}_{u,z} = Q_a(\alpha^{\text{ver}}_{u,z}, \beta^{\text{ver}}_{u,z}), \qquad \text{UCB}_{u,z} = Q_{1-a}(\alpha^{\text{ver}}_{u,z}, \beta^{\text{ver}}_{u,z}). \tag{C.7}$$

With fractional confidences these bounds are credible quantiles of a power posterior (proposition D.10). The − Credible bounds ablation of Table 2 replaces them by the posterior mean $\alpha^{\text{ver}}_{u,z}/(\alpha^{\text{ver}}_{u,z} + \beta^{\text{ver}}_{u,z})$.

C.2 Phase Transition and Skill Evolution

A partition-function-free stagnation statistic. For a fixed query q, fixed graph, policy, and trajectory law, the source log-flow $\log Z_\theta(q)$ shifts every complete-trajectory residual $\delta_{0:T}(\tau)$ by the same constant, hence the residual variance

$$V_q \;=\; \text{Var}_{\tau|q}\big[\delta_{0:T}(\tau)\big] \tag{C.8}$$

is invariant to that source-log-flow shift. It is estimated from independent rollouts of each query, and queries are aggregated with a random-effects model.

Trigger. Let $\bar{V}_w$ be V_q aggregated over the held-out validation query set (Appendix E) at training step w, b the slope of a linear fit to $\bar{V}_w$ over a window W, and $\text{CI}(b)$ its confidence interval at a fixed level. A phase boundary is declared iff all three hold:

$$\text{CI}(b) \subseteq [-\epsilon_b,\, \epsilon_b], \qquad \frac{\bar{V}_{w-W} - \bar{V}_w}{\max\{\bar{V}_{w-W},\, v_{\min}\}} \in [0, \gamma_{\text{var}}), \qquad \Delta H_{\text{norm}} < -h_0, \tag{C.9}$$

where $\epsilon_b > 0$ is the slope tolerance, $v_{\min} > 0$ a numerical floor, $\gamma_{\text{var}} > 0$ the plateau tolerance, $h_0 > 0$ a margin, and ΔH_{norm} the change over the window of the normalised skill entropy $H_{\text{norm}} = H(\mathcal{S} \mid q, z)/\log|\mathcal{S}_{\text{active}}|$, with $H(\mathcal{S} \mid q, z) = -\sum_u p_{u|q,z} \log p_{u|q,z}$ and $p_{u|q,z} \propto n^{\text{call}}_{u,q,z}$ computed from the held-out rollouts in that query/context cell. The first condition is an equivalence test that places the whole interval inside a tolerance band, the lower bound of zero in the second excludes divergence, and the normalised entropy keeps changes in library size from masquerading as saturation.

Evolution operator. The library is refined by

$$\mathcal{S}^{(k+1)} \;=\; \Phi\big(\mathcal{S}^{(k)};\; \Psi^0(u),\; n^{\text{call}}_u,\; \text{LCB}_{u,z},\; \text{UCB}_{u,z},\; \widetilde{A}_e\big), \tag{C.10}$$

whose actions live at three levels: single skills, skill pairs and context clusters.

Single-skill decisions, evaluated in this order for each skill u:

(1) **Defer / Explore**: $N_{\text{eff},u} < n_{\min}$. No structural change; allocate verification and exploration budget instead, since the effective sample size of u is below $n_{\min}$.

(2) **Split**: with sufficient support in each context, $\mathbb{P}\big(\max_z p_{u,z} - \min_z p_{u,z} > \theta_h \mid \mathcal{D}\big) > 1 - a$.

(3) **Refine**: globally sound but locally confirmed weak, $\text{LCB}^{\text{glob}}_u \geq \theta_{\text{mid}}$ and $\exists z:\ \text{UCB}_{u,z} < \theta_{\text{low}}$.

(4) **Retain**: confirmed reliable at the skill level, $\text{LCB}^{\text{glob}}_u \geq \theta_{\text{high}}$.

(5) **Prune**: $\max_z \text{UCB}_{u,z} < \theta_{\text{low}}$ and $\overline{\widetilde{A}}(u) \leq 0$ and u provides no unique coverage.

Pairwise: **Consolidate** is a redundancy test on a skill pair: near equivalence of invocation contexts, outputs and verifier outcomes, with no coverage loss on removal; downstream reward and cost are checked by the paired validation below. *Cluster-level*: **Generate** requires a context cluster with sufficient support, concentrated failure mass and no existing skill whose UCB is adequate there, and the new skill must be checkable by the verifiers. Here $N_{\mathrm{eff},u}$ is the Kish effective sample size (C.5) over the verifier records of skill u, and sufficient support means an effective sample size of at least $n_{\min}$ for the skill, context or cluster concerned; $\mathrm{LCB}_u^{\mathrm{glob}}$ is the lower credible bound of the skill-level posterior pooled over contexts; u provides unique coverage when some context it serves is served by no other skill; and $\widetilde{\widetilde{A}}(u)$ aggregates $\widetilde{A}_e$ over the invocations of u and serves for proposal ordering and as the utility veto of Prune. The thresholds satisfy $0 \le \theta_{\mathrm{low}} < \theta_{\mathrm{mid}} < \theta_{\mathrm{high}} \le 1$.

Versioning, rollback, hysteresis. Each application of Φ yields $\mathcal{G}^{(k+1)}$ and is accepted only after paired rollouts on the held-out validation query set with a non-inferiority margin on success, tempered terminal reward, token cost and latency, assessed by a one-sided test per metric against that margin (Schuirmann, 1987); otherwise it is rolled back. A cooldown prevents a skill from being re-edited within the cooldown window through Split–Refine–Prune.

Definition C.1 (Verifier-only acceptance gate)**.** Let $\mathcal{C}_k$ be the candidate edits proposed or ranked from flow diagnostics, signed utility, and retained records at phase k. Define $\mathrm{Acc}_k(d) = 1$ only if (i) every structural predicate for d is computed from the verifier posterior $(\alpha^{\mathrm{ver}}, \beta^{\mathrm{ver}})$ and its support threshold $n_{\min}$, (ii) the candidate passes the paired held-out non-inferiority check, and (iii) the version, cooldown and coverage predicates hold. The soft score q^{soft} and $\widetilde{A}_e$ rank $\mathcal{C}_k$, and Prune additionally requires the utility veto $\widetilde{\widetilde{A}}(u) \le 0$ for the pruned skill u.

Proposition C.1 (Gate and rollback invariants)**.** *If Acc_k is applied after candidate generation and before a library version is committed, then every committed edit has verifier-only structural eligibility and paired validation, while a failed validation or failed gate leaves the active library equal to $\mathcal{S}^{(k)}$.*

Proof. By definition, a candidate can enter the commit branch only when $\mathrm{Acc}_k(d) = 1$, which includes the verifier-only predicate and the paired non-inferiority check. If either check fails, the transition takes the defer or rollback branch, whose returned library is defined to be the previous version $\mathcal{S}^{(k)}$. Since q^{soft} and $\widetilde{A}_e$ cannot supply the verifier-only eligibility condition, changing them while holding the verifier posterior and validation outcomes fixed cannot turn an ineligible candidate into an eligible one. Reward-derived signals can therefore block an edit through the utility veto of Prune but never make one eligible for the commit branch. □

Proposition C.2 (Phase-state recursion)**.** *Let X_k contain the active library and graph version, trainable policy and flow parameters, retained verifier statistics, and all persistent bookkeeping read by the next transition. Under a fixed outer protocol λ and phase randomness ζ_k, the completed phase transition has the form*

$$X_{k+1} = \mathcal{U}_\lambda(X_k; \zeta_k), \qquad r_k = \mathcal{L}_\lambda(X_k, X_{k+1}; \zeta_k), \qquad \mathcal{H}_{k+1}^{\mathrm{imp}} = \mathcal{H}_k^{\mathrm{imp}} \frown [r_k]. \tag{C.11}$$

Consequently $\Xi_k = (X_k, \mathcal{H}_k^{\mathrm{imp}})$, which pairs X_k with the log of edit decisions and validation outcomes, obeys the well-defined recurrence $\Xi_{k+1} = \mathfrak{R}_\lambda(\Xi_k; \zeta_k)$.

Proof. The fixed protocol maps a completed phase state and its random input to the next state. The record map $\mathcal{L}_\lambda$ is total on completed transitions, so sequence concatenation defines the next audit log. Pairing the two updates gives $\mathfrak{R}_\lambda$. Because X_k contains every field the next transition reads, this map is a function of the phase state Ξ_k for fixed phase randomness ζ_k. □

C.3 Approximate Balance and Abstraction Error

Reasoning-conditioned events. The implemented residual scores $\log P_F(e_t \mid s_{t-1}, r_t)$ with the sampled reasoning text as context; free-text arguments of a skill call are treated the same way, so $P_F(\mathrm{args} \mid u, s)$ scores only the structured arguments. Sampling and scoring then use the same conditional policy, so the balance identities apply to it (remark A.2). When $|\delta_{0:T}| \le \varepsilon$ holds for every sampled reasoning text, the terminal-distribution bound of proposition C.3 applies to the marginal policy as well, because the marginal trajectory probability averages the conditional trajectory probabilities, each of which satisfies the residual bound.

Proposition C.3 (Residual to terminal distribution)**.** *Let $\widehat{Z} = F_\psi(s_0) = Z_\theta(q)$ be the learned source flow and $Z^\star = \sum_{x\in\mathcal{X}} R_\eta(x)$ the true reward partition function. On the finite reachable DAG, assume that the forward and backward policies have positive support on every legal edge and that $|\delta_{0:T}(\tau)| \le \varepsilon$ for every complete legal trajectory τ (hence for every terminal-support fibre). Then for every terminal x,*

$$e^{-\varepsilon} \;\le\; \frac{\widehat{Z}\, P_F^{\mathrm{term}}(x \mid q)}{R_\eta(x)} \;\le\; e^{\varepsilon}, \qquad e^{-\varepsilon} \;\le\; \frac{\widehat{Z}}{Z^\star} \;\le\; e^{\varepsilon}, \qquad e^{-2\varepsilon} \;\le\; \frac{Z^\star P_F^{\mathrm{term}}(x \mid q)}{R_\eta(x)} \;\le\; e^{2\varepsilon}. \tag{C.12}$$

Proof. $\delta_{0:T}(\tau) = \log\big[\widehat{Z} P_F(\tau)/(R_\eta(x_\tau) P_B(\tau \mid x_\tau))\big]$. Exponentiating the residual bound and summing over trajectories ending at x, while using $\sum_{\tau\to x} P_B(\tau \mid x) = 1$, gives the first inequality. Summing the first inequality after multiplying by $R_\eta(x)/\widehat{Z}$ over all terminal states gives $e^{-\varepsilon} \le \widehat{Z}/Z^\star \le e^{\varepsilon}$. Multiplying the first inequality by $Z^\star/\widehat{Z}$, which the second inequality bounds, yields the third inequality, with the exponent 2ε in place of ε. □

Proposition C.4 (Abstraction error propagation)**.** *Let σ be the abstraction map and π a common abstract policy. For a concrete history H with $\sigma(H) = s$, let $K^{\sigma,\pi}_{H,t}(\cdot \mid s)$ be the history-level one-step terminal-or-successor kernel, and let $\overline{K}_t(\cdot \mid s)$ be the adopted quotient kernel. Assume a common initial abstract distribution, terminal space, policy and termination convention, and*

$$\sup_{H:\,\sigma(H)=s} \mathrm{TV}\Big(K^{\sigma,\pi}_{H,t}(\cdot \mid s), \overline{K}_t(\cdot \mid s)\Big) \le \varepsilon_t \quad \textit{for every reachable } s \textit{ at decision step } t. \tag{C.13}$$

If terminated paths are padded by a common absorbing transition and all paths have at most T decision steps, then the terminal marginals of the history-level and quotient processes differ by at most $\min\{1, \sum_{t=0}^{T-1} \varepsilon_t\}$ in total variation. With $\varepsilon_t = 0$ at every decision step, the kernel bound of eq. (C.13) *is the model-irrelevance condition of Li et al. (2006), and the terminal marginals of the history-level and quotient processes then coincide.*

Proof. Project the history-level process to the abstract state. Conditional on a projected state, its one-step kernel is the mixture of the history-level kernels over the conditional distribution of concrete histories. Convexity of total variation and eq. (C.13) bound the distance between this mixture and $\overline{K}_t$ by ε_t. A stepwise hybrid argument, or induction using contraction of Markov kernels, then gives the sum of the per-step bounds; the outer minimum uses $\mathrm{TV} \le 1$. □

Remark C.3 (Reward measurability of merged terminals). Assumption A.1(c) makes $R_\eta(x)$ well defined on every merged terminal, so the terminal anchor of eq. (B.2) applies there; the abstraction map σ meets this terminal-reward consistency by keeping in s every feature of the history that the reward reads, which makes the reward σ-measurable.

D Proof Details

This section proves the flow identities used by the method text. The exact results characterise the zero-residual optimum, and proposition C.3 bounds the deviation when residuals are small. The first subsection derives the path factorisation and the balance identities, the second builds the compatible reference flow and the rollout estimator of the flow share, and the third treats the verifier posteriors behind the credible bounds and the acceptance gate of definition C.1.

D.1 Path factorisation and balance identities

Lemma D.1 (Path factorisation)**.** *Under Assumptions A.1 to A.4, let $\tau = (s_0, e_1, s_1, \ldots, e_T, s_T)$ be a legal trajectory ending at $x = s_T$. For any normalized forward policy P_F and backward policy P_B with positive support on the edges of τ,*

$$P_F(\tau) = \prod_{t=1}^{T} P_F(e_t \mid s_{t-1}), \qquad P_B(\tau \mid x) = \prod_{t=1}^{T} P_B((s_{t-1}, e_t) \mid s_t). \tag{D.1}$$

Proof. The Markov assumption makes the conditional law of the next event depend only on (s_{t-1}, q, k), and Assumption A.4 makes each successor a function of the current state and event, so the chain rule reduces to the first product. Running the normalized in-edge policy backward from the terminal $x = s_T$ gives the second product of eq. (D.1). □

Lemma D.2 (Residual additivity)**.** *For $0 \le i < \ell < j \le T$, the residual in eq.* (B.1) *satisfies*

$$\delta_{i:j} = \delta_{i:\ell} + \delta_{\ell:j}. \tag{D.2}$$

Consequently, zero residual on every sub-trajectory implies both detailed balance on every edge and trajectory balance on every complete trajectory of the DAG.

Proof. Expand the two terms on the right of eq. (D.2). The intermediate terms $\log F_\psi(s_\ell)$ and $-\log F_\psi(s_\ell)$ cancel, as do no other terms, leaving exactly the sums from $i+1$ through j. Choosing $j = i+1$ gives detailed balance on every edge, and choosing $(i, j) = (0, T)$ gives trajectory balance on every complete trajectory. The converse does not hold from trajectory balance alone: internal flows can change while preserving the endpoint product. □

Proposition D.3 (Sub-trajectory objective and exact balance)**.** *Let μ be the distribution over complete legal trajectories used for the population objective and write $\mathcal{J}_\mu = \mathbb{E}_{\tau\sim\mu}[\mathcal{L}_{\text{flow}}(\tau)]$. Assume $\omega_{j-i} > 0$ for every included pair and $\Pr_\mu(e \in \tau) > 0$ for every constrained edge. Then $\mathcal{J}_\mu = 0$ if and only if every included residual is zero on the support of μ. If all contiguous pairs are included and the support covers every legal edge, this is equivalent to the collection of detailed-balance identities on the DAG together with the terminal anchor of eq.* (B.2)*.*

Proof. The integrand is a finite sum of nonnegative terms. Its expectation is zero only if every term is zero μ-almost surely. The positive edge-visitation condition makes each constrained edge occur with positive probability, so each included residual is zero on that edge's support. The first implication of lemma D.2 gives detailed balance and trajectory identities. Conversely, those identities make every included residual $\delta_{i:j}$ in the objective of eq. (B.3) zero. □

Remark D.1 (From exact to approximate balance). A finite-sample optimizer leaves small nonzero residuals; when every complete-trajectory residual is at most ε in magnitude, proposition C.3 bounds the resulting deviation of the terminal law in eq. (C.12).

D.2 Compatible reference flows

Proposition D.4 (Backward reconstruction of a compatible triple)**.** *On a finite DAG in which every reachable node lies on a source-to-terminal path, with positive terminal rewards and a normalized full-support backward policy P_B^0 on every nonempty in-edge set, define the reference state and edge flows recursively*

$$\begin{aligned} F^0(x) &= R_\eta(x) && (x \in \mathcal{X}), \qquad f^0(s, e, s') = F^0(s')P_B^0((s, e) \mid s'), \\ F^0(s) &= \sum_{(s,e,s')\in \mathrm{Out}(s)} f^0(s, e, s') && (s \notin \mathcal{X}). \end{aligned} \tag{D.3}$$

Then $Z^\star = F^0(s_0) = \sum_x R_\eta(x)$ and $P_F^0(e \mid s) = f^0(s, e, s')/F^0(s)$ form a compatible flow triple (F^0, P_F^0, P_B^0). It is unique for the chosen P_B^0 and terminal anchor.

Proof. The recurrence is evaluated in decreasing rank. Terminal values are fixed by the terminal anchor. Since every reachable node lies on a terminal path and all terminal rewards and backward probabilities are positive, induction gives $F^0(s) > 0$ at every reachable node. Once all children of s have been assigned flow values, every outgoing edge mass is fixed by P_B^0, so their sum fixes $F^0(s)$ and division by it fixes the normalized forward probabilities. For each non-root node s', normalization of the backward policy P_B^0 gives $\sum_{(s,e)\in \mathrm{In}(s')} f^0(s, e, s') = F^0(s')$. Summing this incoming-flow identity over all non-root states and the outgoing-flow definition over all nonterminal states cancels the mass of every internal edge, leaving $F^0(s_0) = \sum_{x\in\mathcal{X}} F^0(x) = \sum_x R_\eta(x)$. The same induction over decreasing rank shows uniqueness. The construction also gives $F^0(s)P_F^0(e \mid s) = F^0(s')P_B^0((s, e) \mid s')$, which is the required compatibility identity. □

Corollary D.5 (Terminal law under an exact compatible triple)**.** *If a forward policy is generated by proposition D.4, then its terminal marginal is*

$$P_F^{0,\mathrm{term}}(x \mid q) = \frac{R_\eta(x)}{Z^\star}. \tag{D.4}$$

The terminal law is invariant to the choice of full-support P_B^0, while the internal state and edge flows generally are not, as in the two-path example of proposition A.3.

Proof. For every trajectory ending at x, compatibility gives $P_F^0(\tau) = R_\eta(x)P_B^0(\tau \mid x)/Z^\star$. Summing over trajectories and using normalization of the backward trajectory distribution gives eq. (D.4). Internal flows contain the factors $P_B^0(\tau \mid x)$ before they are summed, so two backward policies can induce different internal allocations while preserving the same terminal marginal. □

Remark D.2 (Reference readout versus learned training triple). Equation (B.6) is a flow diagnostic only when all of its factors come from one compatible triple; pairing a learned $P_{F,\theta}$ with a state flow built from another P_B^0 need not conserve flow. R^2 Flow therefore estimates the reference readout from the trained policy's rollouts with the residual weights of corollary D.9, which needs no reconstruction.

Lemma D.6 (Occupancy identity)**.** *For a compatible positive flow on the fixed rooted, terminal-complete DAG,*

$$\Pr_{P_F^0}(s \in \tau) = \frac{F^0(s)}{Z^\star}, \qquad \Pr_{P_F^0}((s, e, s') \in \tau) = \frac{F^0(s)P_F^0(e \mid s)}{Z^\star}. \tag{D.5}$$

Proof. The source is visited with probability one. In increasing rank, incoming-edge events to a fixed state are mutually exclusive, so its visitation probability satisfies

$$v(s') = \sum_{(s,e)\in \mathrm{In}(s')} v(s)P_F^0(e \mid s) = \frac{\sum_{(s,e)\in \mathrm{In}(s')} F^0(s)P_F^0(e \mid s)}{Z^\star} = \frac{F^0(s')}{Z^\star},$$

where the last equality is flow conservation. Multiplying $v(s)$ by the conditional edge probability gives the edge identity. Terminal completeness makes these visitation probabilities equal to sums over the complete trajectories containing the state or edge. □

Proposition D.7 (Flow-share normalization)**.** *Fix a finite compatible reference-flow triple (F^0, P_F^0, P_B^0) with source flow $F^0(s_0) = Z^\star > 0$. Let $\mathcal{E}_u$ be the set of counted edges labelled by skill u, assume that the sets $\{\mathcal{E}_u\}_u$ form a disjoint partition of $\mathcal{E}_{\mathrm{skill}}$, and define*

$$\varphi^0(e) = \frac{F^0(s)P_F^0(e \mid s)}{Z^\star} \quad \textit{for } e = (s \to s'). \tag{D.6}$$

For a complete trajectory τ, let

$$I_e(\tau) = \mathbf{1}\{e \in \tau\}, \qquad N_u(\tau) = \sum_{e\in\mathcal{E}_u} I_e(\tau), \qquad T_{\mathrm{skill}}(\tau) = \sum_{e\in\mathcal{E}_{\mathrm{skill}}} I_e(\tau). \tag{D.7}$$

Then

$$\sum_{e\in\mathcal{E}_u} \varphi^0(e) = \mathbb{E}_{\tau\sim P_F^0}[N_u(\tau)], \qquad M_{\mathrm{skill}} = \sum_{e\in\mathcal{E}_{\mathrm{skill}}} \varphi^0(e) = \mathbb{E}_{\tau\sim P_F^0}[T_{\mathrm{skill}}(\tau)]. \tag{D.8}$$

If $M_{\mathrm{skill}} > 0$, the normalized skill share is therefore

$$\Psi^0(u) = \frac{\mathbb{E}_{P_F^0}[N_u(\tau)]}{\mathbb{E}_{P_F^0}[T_{\mathrm{skill}}(\tau)]} = \frac{\sum_{e\in\mathcal{E}_u}\varphi^0(e)}{M_{\mathrm{skill}}}, \qquad \sum_u \Psi^0(u) = 1. \tag{D.9}$$

In general this is a ratio of expectations, not $\mathbb{E}[N_u(\tau)/T_{\mathrm{skill}}(\tau)]$.

Proof. Because the graph is acyclic, a complete trajectory traverses any fixed edge at most once. Compatible flow balance gives

$$F^0(s)P_F^0(e \mid s) = \sum_{\tau:\, e\in\tau} F^0(s_0)P_F^0(\tau). \tag{D.10}$$

By lemma D.6, dividing by $F^0(s_0) = Z^\star$ yields

$$\varphi^0(e) = \sum_\tau P_F^0(\tau) I_e(\tau) = \mathbb{E}_{\tau \sim P_F^0}[I_e(\tau)]. \tag{D.11}$$

All sums are finite, so linearity of expectation permits exchanging the edge sum and the trajectory expectation. This proves both identities in eq. (D.8). Since the skill-labelled edge sets form a partition, $T_{\text{skill}}(\tau) = \sum_u N_u(\tau)$ pointwise. Taking expectations and dividing by the common positive denominator M_{skill} of eq. (D.8) proves eq. (D.9). □

Corollary D.8 (Backward-policy independence under path-invariant counts)**.** *Suppose that for every terminal node x there are deterministic functions $n_u(x)$ and $t_{\text{skill}}(x)$ such that every complete trajectory ending at x satisfies*

$$N_u(\tau) = n_u(x), \qquad T_{\text{skill}}(\tau) = t_{\text{skill}}(x). \tag{D.12}$$

Then every full-support compatible backward policy P_B^0 induces the same normalized skill share:

$$\Psi^0(u) = \frac{\sum_x R_\eta(x) n_u(x)}{\sum_x R_\eta(x) t_{\text{skill}}(x)}, \tag{D.13}$$

provided the denominator of eq. (D.13) *is positive.*

Proof. Compatibility gives the terminal law $P_F^0(x) = R_\eta(x)/Z^\star$, independently of the chosen full-support backward policy. Under eq. (D.12), conditioning on x removes all remaining path dependence:

$$\mathbb{E}_{P_F^0}[N_u(\tau)] = \sum_x \frac{R_\eta(x)}{Z^\star} n_u(x), \qquad \mathbb{E}_{P_F^0}[T_{\text{skill}}(\tau)] = \sum_x \frac{R_\eta(x)}{Z^\star} t_{\text{skill}}(x). \tag{D.14}$$

Substitution into eq. (D.9) cancels $Z^\star$ and yields eq. (D.13). □

Corollary D.9 (Skill share on the shared-state graph and its estimator)**.** *Let every edge commit one atomic event labelled by one skill $u \in \mathcal{S}^{(k)}$ (definition A.2).*

(i) Every complete trajectory ending at x satisfies eq. (D.12)*, with $n_u(x)$ the number of events of E_x that invoke u and $t_{\text{skill}}(x) = \sum_u n_u(x)$, so $\Psi^0(u)$ is given by eq.* (D.13) *for every full-support compatible backward policy P_B^0, as in corollary D.8.*

(ii) Let $\tau_1, \dots, \tau_N$ be rollouts of query q drawn independently from a forward policy with positive support on every legal edge, let P_B be any normalized full-support backward policy, and let $w_i = \exp(-\delta_{0:T}(\tau_i))$ be computed from them. Then

$$\widehat{\Psi}_N^0(u) = \frac{\sum_i w_i \, n_u(x_{\tau_i})}{\sum_i w_i \, t_{\text{skill}}(x_{\tau_i})} \;\longrightarrow\; \Psi^0(u) \quad \textit{almost surely as } N \to \infty, \tag{D.15}$$

whatever the values of the source flow $F_\psi(s_0)$ and the domain bias b_d.

(iii) With unit weights and $|\delta_{0:T}| \le \varepsilon$ on every complete legal trajectory, the limit of $\widehat{\Psi}_N^0(u)$ in eq. (D.15) *lies in $[e^{-4\varepsilon}, e^{4\varepsilon}]\,\Psi^0(u)$.*

The same weights estimate the edge share, $\widehat{\varphi}_N^0(e) = \sum_i w_i \, h_{B^0}(e \mid x_{\tau_i}) / \sum_i w_i$, where $h_{B^0}(e \mid x)$ is the probability that a backward walk under P_B^0 from x traverses e.

Proof. (i) A state records its committed events, the source has none, and each edge adds one event, so a complete trajectory ending at x adds every event of E_x exactly once and its skill counts depend on x alone; corollary D.8 then applies. (ii) As in the proof of proposition C.3, $w_i = c_q \, R_\eta(x) P_B(\tau_i \mid x)/P_F(\tau_i)$ with $x = x_{\tau_i}$ and a constant $c_q > 0$ that collects $F_\psi(s_0)$ and b_d for the query. For any function g of the terminal, $\mathbb{E}_{P_F}[w \, g(x_\tau)] = c_q \sum_\tau R_\eta(x_\tau) P_B(\tau \mid x_\tau) g(x_\tau) = c_q \sum_x R_\eta(x) g(x)$, because $\sum_{\tau \to x} P_B(\tau \mid x) = 1$. Taking $g = n_u$ and $g = t_{\text{skill}}$, the strong law of large numbers applies to the numerator and the denominator, c_q cancels, and the ratio converges to eq. (D.13). When the forward term conditions on sampled reasoning text, $P_F(\tau_i)$ is the event probability given that text; the text is drawn from its own distribution, so summing over it leaves the same expectation. (iii) With

unit weights the limit is $\sum_x P_F^{\mathrm{term}}(x \mid q)\, n_u(x) / \sum_x P_F^{\mathrm{term}}(x \mid q)\, t_{\mathrm{skill}}(x)$. By proposition C.3, $P_F^{\mathrm{term}}(x \mid q)$ lies within a factor $e^{\pm 2\varepsilon}$ of $R_\eta(x)/Z^\star$, so the numerator and the denominator each lie within that factor of their targets and the ratio within $e^{\pm 4\varepsilon}$. For the edge share, take $g(x) = h_{B^0}(e \mid x)$ and $g \equiv 1$; the limit is $\sum_x R_\eta(x) h_{B^0}(e \mid x)/Z^\star = \varphi^0(e)$ by proposition B.1 applied to the reference triple and the compatibility identity $F^0(s)P_F^0(e \mid s) = F^0(s')P_B^0((s,e) \mid s')$. □

Remark D.3 (Precision of the share estimate). The Kish effective sample size (C.5) of $w_1, \ldots, w_N$ measures the precision of eq. (D.15). The log-weights have variance V_q, so if they are roughly Gaussian the effective sample size is about Ne^{-V_q} and the estimate sharpens as V_q falls.

D.3 Verifier posteriors and decision invariants

Proposition D.10 (Fractional verifier evidence as a power posterior)**.** *Let $p_{u,z}$ have prior $\mathrm{Beta}(\alpha_{0,u}, \beta_{0,u})$ with $\alpha_{0,u}, \beta_{0,u} > 0$. For binary verifier records $y_e \in \{0,1\}$ with confidence weights $c_e \in [0,1]$, define the generalized likelihood*

$$L(p) = \prod_{e \in \mathcal{E}_{u,z}^{\mathrm{ver}}} [p^{y_e}(1-p)^{1-y_e}]^{c_e}. \tag{D.16}$$

Then the generalized posterior (Bissiri et al., 2016) is $\mathrm{Beta}(\alpha_{u,z}^{\mathrm{ver}}, \beta_{u,z}^{\mathrm{ver}})$ with the confidence-weighted update of the gate posterior in eqs. (C.3) *and* (C.4)*.*

Proof. Multiplying the Beta prior density by eq. (D.16) collects the powers of p and $(1-p)$ as $p^{\alpha_{0,u}-1+\sum_{e \in \mathcal{E}_{u,z}^{\mathrm{ver}}} c_e y_e}$ and $(1-p)^{\beta_{0,u}-1+\sum_{e \in \mathcal{E}_{u,z}^{\mathrm{ver}}} c_e (1-y_e)}$. The normalizing constant is the Beta function, giving the stated generalized posterior. Because fractional powers are used, this is a power-posterior interpretation of the update in eqs. (C.3) and (C.4). □

Lemma D.11 (Beta density-ratio ordering)**.** *Let $P_{\alpha,\beta}$ denote a $\mathrm{Beta}(\alpha,\beta)$ random variable, where $\alpha, \beta > 0$. For every $c \geq 0$,*

$$P_{\alpha+c,\beta} \succeq_{\mathrm{FOSD}} P_{\alpha,\beta}, \qquad P_{\alpha,\beta+c} \preceq_{\mathrm{FOSD}} P_{\alpha,\beta}, \tag{D.17}$$

where $\succeq_{\mathrm{FOSD}}$ denotes first-order stochastic dominance (Shaked & Shanthikumar, 2007). The inequalities are strict on the interior $(0,1)$ when $c > 0$.

Proof. The case $c = 0$ is equality, so take $c > 0$. The two Beta densities satisfy

$$\frac{f_{\alpha+c,\beta}(p)}{f_{\alpha,\beta}(p)} = \frac{B(\alpha,\beta)}{B(\alpha+c,\beta)} p^c =: r_{\alpha,c}(p). \tag{D.18}$$

The ratio is continuous and strictly increasing. Since both densities integrate to one, it has a unique crossing $p_\star \in (0,1)$: the density difference is negative before $p_\star$ and positive after it. For $D(t) = F_{\alpha+c,\beta}(t) - F_{\alpha,\beta}(t)$, this gives $D(t) \leq 0$ for $t \leq p_\star$. For $t > p_\star$, using $\int_0^1 (f_{\alpha+c,\beta} - f_{\alpha,\beta}) = 0$ gives

$$D(t) = -\int_t^1 (f_{\alpha+c,\beta}(p) - f_{\alpha,\beta}(p))\, dp \leq 0. \tag{D.19}$$

Thus the first CDF is everywhere no larger, proving the first stochastic-order relation. Similarly,

$$\frac{f_{\alpha,\beta+c}(p)}{f_{\alpha,\beta}(p)} = \frac{B(\alpha,\beta)}{B(\alpha,\beta+c)} (1-p)^c \tag{D.20}$$

is strictly decreasing; reversing the preceding sign argument proves the second relation. Positivity of Beta densities on $(0,1)$ gives strict interior inequalities for $c > 0$. □

Corollary D.12 (Monotonicity of verifier-posterior quantiles)**.** *For $a \in (0,1)$, define*

$$Q_a(\alpha,\beta) = \inf\{p \in [0,1] : F_{\alpha,\beta}(p) \geq a\}. \tag{D.21}$$

Then, for every $c \geq 0$,

$$Q_a(\alpha+c,\beta) \geq Q_a(\alpha,\beta), \qquad Q_a(\alpha,\beta+c) \leq Q_a(\alpha,\beta). \tag{D.22}$$

Consequently, adding verifier success mass c cannot decrease a lower posterior quantile, and adding verifier failure mass c cannot increase an upper posterior quantile, provided the other shape parameter and prior hyperparameters are held fixed.

Proof. If $F_1(t) \leq F_0(t)$ for every t, the level a is reached by F_1 no earlier than by F_0, so $Q_a(F_1) \geq Q_a(F_0)$. Applying this implication to the CDF orderings in lemma D.11 proves the inequalities. Under the power-posterior update of proposition D.10, a verifier success of mass c changes (α, β) to $(\alpha + c, \beta)$ and a verifier failure of mass c changes it to $(\alpha, \beta + c)$. □

E FULL EXPERIMENTAL RESULTS

This appendix holds the protocol details and the per-row analysis behind Section 5.

Benchmarks and metrics. The IID datasets are HotpotQA (Yang et al., 2018), TriviaQA (Joshi et al., 2017), AIME 2026, HealthBench (Arora et al., 2025), MBPP+ (Austin et al., 2021; Liu et al., 2023) and ALFWorld (Shridhar et al., 2020); the OOD datasets are MuSiQue-Ans (Trivedi et al., 2022), NQ-Open (Kwiatkowski et al., 2019; Lee et al., 2019), MATH-Hard (Hendrycks et al., 2021), GPQA (Rein et al., 2023), SWE-bench (Jimenez et al., 2024) and WebShop (Yao et al., 2022a). Every source is pinned to a repository revision and an evaluation population, and each dataset is a fixed 128-record draw under one fixed seed; AIME 2026 is evaluated on all thirty released problems. HotpotQA, TriviaQA, MuSiQue-Ans and NQ-Open report exact match, the primary metric used in every table and figure, and answer F1. AIME 2026, MATH-Hard and GPQA report accuracy, HealthBench the native rubric mean, ALFWorld and WebShop the native success rate, SWE-bench the resolved rate, and MBPP+ pass@1 against base and extended tests. Every primary metric and its verifier version are fixed before any training or evaluation run.

Implementation. The Supervisor and the frozen executor are both Qwen3.5-9B. The phase trigger and paired validation of Appendix C use a separate validation query set, disjoint from the training queries and from every evaluation record; the six OOD datasets serve only for final evaluation. All methods share the same executor, tool set, retrieval corpus and inference budget. Verifier labels come from unit tests, schema validation, assertions and execution status (Appendix C).

Extended analysis. *Main results.* Training the backbone raises the IID exact-match average by at most 7.26 points (GRPO†) but lowers AIME 2026 from 51.33 to 32.00 (SFT) and 36.00 (GRPO†). Out of distribution, the margin of R^2 Flow over the best baseline is widest on MuSiQue-Ans (3.90), NQ-Open and WebShop (3.75 each), and its gain over Qwen3.5 ranges from 7.5 points on MATH-Hard to 63.0 on NQ-Open. With the 9B backbone, R^2 Flow exceeds DeepSeek-V4-Flash prompted directly on all twelve datasets, including SWE-bench (44.22 against 34.84). *Executor transfer.* On the six IID datasets the frozen averages range from 62.3 (glm-5.3) to 76.8 (gemini-3.8-flash) and those under R^2 Flow from 84.7 to 91.6; for every backbone the average gain on the OOD datasets exceeds that on the IID datasets. Aggregated by OOD domain, the gain is largest on question answering (25.3 to 45.0 points) and interactive tasks (24.1 to 35.6) and smallest on Math/Sci (8.2 to 15.9), where the frozen scores already lie between 76.6 and 88.0; code stays the lowest domain after orchestration (44.8 to 53.0) while still gaining 8.8 to 19.8 points. *Ablations.* The ablated cells of Table 2 fall 0.31 to 14.00 points below the full model. Returning states to full histories drops most on AIME 2026 (14.00), WebShop (10.15) and HealthBench (8.14). Reading the shares off the learned flow instead of the reference readout costs 2.63 IID and 1.61 OOD points, and ranking by share alone costs most on WebShop (6.72). Replacing verifier labels by reward-derived labels costs most on WebShop (9.37) and MuSiQue-Ans (6.40); the posterior mean in place of the credible bounds costs 2.23 IID and 2.97 OOD points, skipping the paired non-inferiority check 2.94 and 2.42, and a fixed phase length 1.58 and 2.03. *Training objectives.* The IID mean of R^{2}TB is 2.25 points above SubTB, 2.98 above TTB, 3.07 above TB, 5.13 above DB and 7.15 above GRPO, which is the lowest objective on every IID dataset. SubTB is within 1.09 points of R^{2}TB on TriviaQA, MBPP+ and ALFWorld but 6.67 behind on AIME 2026. Per step, R^{2}TB uses 433 learner GPU-seconds, 11 fewer than SubTB and 44 more than TB, at nearly the same learner token volume as SubTB (3,121,184 against 3,121,927); a whole step takes 2,587 GPU-seconds and 832 s of wall-clock, 7.5% and 5.6% more than GRPO. *Phases, edits and harmful skills.* With fixed-length phases OOD accuracy ends at 78.99; with reward-derived labels it peaks at 77.9 after phase 3; with the library frozen it stays between 73.4 and 74.8 after the first phase. Of the 44 edits committed on reward-derived labels, 25 raise the training reward while lowering the held-out verified score. Of the twelve injected harmful skills, R^2 Flow prunes six by phase 2, three in phases 3–5 and two in phases 6–8, and keeps one.

Seeds and variance. Every reported number is a mean over five runs; Table 1 also reports the standard deviation, and the per-run results are released with the code.