GROKRSI: FEW-SHOT AGENTIC RECURSIVE SELF-IMPROVEMENT THROUGH QUALIFIED TASK–HARNESS CO-EVOLUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Language-model agents can improve their behavior without updating model weights by modifying the external procedures that govern reasoning, memory, and tool use. However, under few-shot supervision, Harness evolution repeatedly learns from a small and largely fixed set of situations. Simply generating more tasks is not sufficient: unreliable self-generated supervision can be recursively amplified as both the curriculum and the agent evolve. We present GrokRSI, a framework for qualified Task–Harness co-evolution. The current executable Harness conditions subsequent task proposals through its public behavior, while qualified outcomes from those tasks determine the evidence available for constructing the next Harness. A qualification and evidence-routing protocol separates verified learning signals from provisional observations, and frozen promotion prevents validation and held-out outcomes from feeding back into subsequent updates. Across two few-shot benchmarks, the joint procedure yields higher held-out point estimates than protocol-matched Harness-only evolution; a complementary standard-scale study tests the same task-side strategy beyond the few-shot regime. Together with the mechanism ablations, these results provide evidence that adapting what an agent learns from together with how it acts is a promising direction for recursive improvement under scarce supervision.

1 INTRODUCTION

Language-model agents are increasingly improved by editing the external systems around a frozen base model: prompts, tools, memory, and control flow determine what the model can observe and do. Recent Harness and agent self-evolution methods show that these components can be adapted from execution feedback (Hu et al., 2024; Zhang et al., 2024; Huang et al., 2026b), but most methods keep the source of that feedback fixed or optimize only one procedural surface at a time. This creates a bottleneck in few-shot settings: repeated updates see only a small and potentially uninformative set of learning situations. Task and curriculum generation can add diversity, yet a plausible generated task does not guarantee a valid domain contract, a checker-aligned response, or a supported answer. If such feedback is admitted indiscriminately, recursive updates can amplify the same errors. The missing capability is therefore a controlled coupling between the evolving task source and the executable Harness.

We ask whether an agent can improve its executable Harness when the task distribution that supplies its feedback is itself scarce, while preventing self-generated feedback from contaminating later updates. We introduce GrokRSI, a Task–Harness co-evolution protocol with domain-agnostic interfaces and domain-specific qualification checks. The current Harness's public behavior conditions subsequent task proposals, while qualified outcomes from those tasks condition the search for complete Harness candidates. Qualification routes task validity, response–checker alignment, and answer provenance before feedback enters learning; frozen validation selects the active candidate, and a one-way held-out evaluation prevents write-back. Here recursive self-improvement means iterative revision of the external executable procedure and learning state while keeping the base-model weights fixed. Our single empirical claim is that, in few-shot settings, this joint evolution yields higher held-out point estimates than protocol-matched Harness-only evolution on fixed tasks. We test this claim on pooled AIME 2025–2026 and HMMT 2025–2026, with GPQA-Diamond as

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.

a complementary standard-scale study. The mathematical benchmarks instantiate these interfaces rather than define the scope of the method.

Contributions.

- **Task–Harness co-evolution.** We formulate recursive agent improvement as a coupled process in which public behavior of the current executable Harness determines subsequent learning situations, while qualified outcomes from those situations determine the evidence available for the next Harness update.
- **Qualified recursive learning.** We provide a protocol with domain-agnostic interfaces that separates verified learning evidence from provisional observations and prevents validation or held-out information from entering subsequent recursive updates.
- **Few-shot evidence.** Under matched solver, split, and evaluation protocols, we compare the coupled procedure with Harness-only evolution on AIME and HMMT and use mechanism ablations to examine the roles of task evolution and cross-round feedback; a complementary GPQA-Diamond study tests the task-side strategy at standard scale.

2 Related Work

Agent and Harness self-improvement. Self-Refine, Reflexion, and Voyager improve agent behavior through iterative feedback, retained reflections, or executable skills (Madaan et al., 2023; Shinn et al., 2023; Wang et al., 2024a). More recent systems optimize the agent procedure itself: ADAS searches agent designs, AFlow synthesizes workflows, MemoHarness adapts a complete Harness from experience, and RSEA uses held-out selection for recursive artifact updates (Hu et al., 2024; Zhang et al., 2024; Huang et al., 2026b; Nguyen et al., 2026). AWM similarly induces reusable workflows from successful trajectories (Wang et al., 2024b). These results establish that context assembly, tools, memory, and output handling can be revised without changing model weights. Our distinction is to evolve the source of learning situations together with the executable Harness, under a fixed held-out boundary.

Environment and task-side adaptation. EnvHarness changes static environments through composable Setup, Rule, and Link layers while preserving a common interface (Huang et al., 2026a). Related agent environments and tool interfaces make interaction substrates more realistic or controllable (Zhou et al., 2023; Yang et al., 2024; Liu et al., 2023; Qin et al., 2023; Yao et al., 2024). RLVE adapts the difficulty distribution of verifiable environments during reinforcement learning (Zeng et al., 2025). Our task-side component operates at a different boundary: it proposes benchmark-compatible task packages, conditions them on the active Harness's public behavior, and routes their evidence before it can affect recursive learning. The benchmark contract is therefore fixed at evaluation even though the learning curriculum changes.

Task generation, curriculum, and cognitive targeting. Curriculum learning studies how example selection and ordering affect learning (Bengio et al., 2009). Self-Instruct and STaR show how generated instructions or rationales can expand a limited seed set (Wang et al., 2023b; Zelikman et al., 2022). Bloom's taxonomy provides a language for distinguishing recall, analysis, and creation (Bloom et al., 1956; Anderson & Krathwohl, 2001); recent work operationalizes these cognitive depths for generated questions (Wang et al., 2026). Such methods do not by themselves guarantee that a generated task has a valid domain contract, an aligned checker, or evidence suitable for recursive updates. Our protocol separates cognitive targeting, which structures proposals, from qualification, which determines whether their feedback is learning-eligible.

Agent memory and positioning. Generative Agents, MemoryBank, and MemGPT study episodic storage, reflection, and hierarchical context management; Reflexion and Voyager retain verbal or executable experience across interactions (Park et al., 2023; Zhong et al., 2023; Packer et al., 2023; Shinn et al., 2023; Wang et al., 2024a). MemoHarness makes retrieval part of Harness adaptation, whereas our protocol distinguishes verified outcomes from provisional observations and exposes the resulting state to both task proposal and Harness construction. Thus prior work supplies the components—adaptive Harnesses, environments, curricula, and memory—but generally studies them as separate axes. GrokRSI connects those axes through a qualified Task–Harness state transition in the few-shot regime, rather than introducing another isolated memory or workflow module.

3 PROBLEM FORMULATION

Let $\mathcal{S}$, $\mathcal{V}$, and $\mathcal{T}$ denote role-disjoint seed, promotion-validation, and heldouttask sets. The seed set is deliberately small; $\mathcal{V}$ is used only to select an active Harness, and $\mathcal{T}$ is revealed only after learning and selection have stopped. An executable Harness$H = (c, \theta)$ consists of source code c and a structured configuration θ governing context assembly, tool use, generation, orchestration, memory, and output processing. The base language model and its weights remain fixed.

At round r, the learning state is $Z_r = (H_r, \mathcal{B}_r, \mathcal{O}_r, \mathcal{A}_r)$, where $\mathcal{B}_r$ is a verified experience bank, $\mathcal{O}_r$ is a provisional observation store, and $\mathcal{A}_r$ is a retentive candidate archive. Task evolution maps public evidence in Z_r to a curriculum of generated task packages; Harness evolution maps the qualified outcomes of that curriculum to complete candidates for H_{r+1}. We study whether repeatedly coupling these two updates yields higher held-out point estimates on $\mathcal{T}$ relative to Harness evolution over a fixed task set.

4 GROKRSI

GrokRSI implements recursive improvement as an alternating update of a task curriculum and an executable Harness. The two sides interact through a shared, qualified learning state rather than through model-weight updates: public behavior from the active Harness shapes the next task proposals, and verified outcomes from those proposals shape the next Harness search. Figure 1 shows the complete protocol: few-shot initialization feeds a feedback-conditioned task generator; generated proposals pass through qualification and evidence routing; and the resulting verified state supports distillation, candidate construction, frozen validation, and promotion. The final evaluation boundary freezes both the selected Harness and the verified bank and forbids write-back, so no held-out response can influence a later task or candidate.

The complete round-level procedure, including qualification, evidence routing, candidate retention, frozen promotion, and held-out isolation, is given in Algorithm 1 in Appendix A.

4.1 ROLE-DISJOINT INITIALIZATION

The initial state combines the small seed set $\mathcal{S}$, an executable Harness H_0, an empty verified bank, an empty provisional store, and an archive anchored by H_0. The validation panel $\mathcal{V}$ and held-out set $\mathcal{T}$ are fixed before recursive learning. Solver identity, base model, tool interface, and resource limits are held constant across rounds so that the evolving objects are the curriculum, accumulated learning state, and Harness source/configuration.

4.2 FEEDBACK-CONDITIONED TASK EVOLUTION

Task evolution converts sparse supervision into a curriculum that responds to the agent's observed behavior. At round r, the generator receives seed and parent lineage $\mathcal{L}_{r-1}$, public observations $\mathcal{O}^{\mathrm{pub}}_{r-1}$, and distilled patterns $\mathcal{P}_{r-1}$:

$$\mathcal{G}_r = G(\mathcal{S}, \mathcal{L}_{r-1}, \mathcal{O}^{\mathrm{pub}}_{r-1}, \mathcal{P}_{r-1}). \tag{1}$$

The public observations expose how the active Harness behaved without exposing private answers, validation episodes, or a direct Harness identifier. This is the Harness-to-task direction of co-evolution: changes in execution alter the observable context from which later tasks are proposed.

Each proposal is a canonical package containing a public task, a cognitive contract, a response schema, a checker, expected-answer support, and parent lineage. We condition proposals on Apply, Analyze, or Create targets, following the cognitive-depth operationalization of Wang et al. (2026) and the broader Bloom tradition (Bloom et al., 1956; Anderson & Krathwohl, 2001). These targets structure the intended cognitive operation; they do not certify task correctness.

4.3 QUALIFICATION AND EVIDENCE ROUTING

Generated supervision must satisfy both the task domain and the mechanism that turns a response into feedback. For a proposal τ, let q_{valid}, q_{align}, and q_{prov} denote domain validity, schema–checker

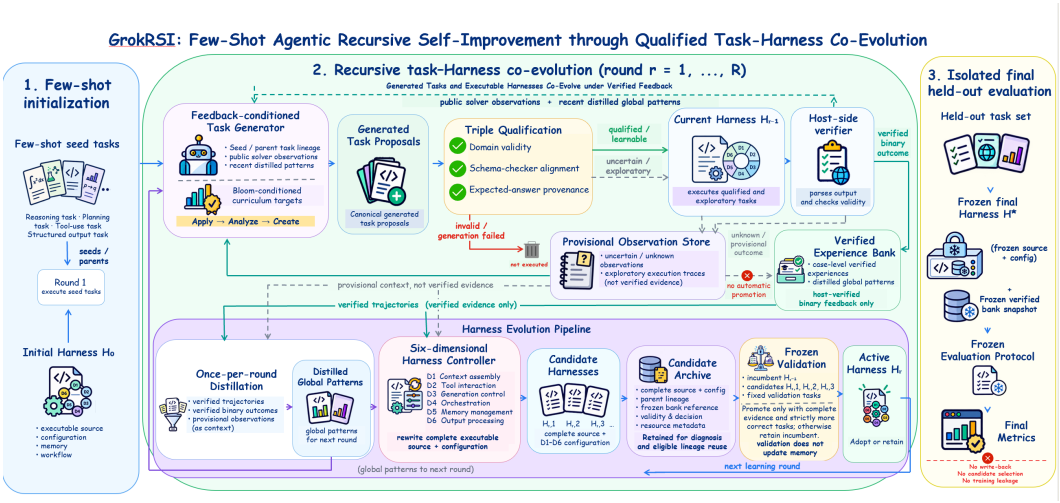


Figure 1: Overview of GrokRSI. Few-shot seed tasks and an initial executable Harness enter a recursive Task–Harness co-evolution loop. Public solver observations and distilled global patterns condition the next task proposals; each proposal is routed through domain validity, schema–checker alignment, and expected-answer provenance. The active Harness executes qualified and exploratory tasks. Outcomes with uncertain or unknown answer status remain in the provisional observation store rather than entering the verified experience bank; their execution traces and public observations remain available as explicitly marked context to the Harness controller. Verified trajectories and distilled patterns drive a six-dimensional Harness controller that constructs and retains complete candidates; frozen validation promotes an active Harness only with complete evidence and strict improvement. The final held-out evaluation uses one frozen Harness and verified-bank snapshot with no write-back. The domain-specific checks shown in the figure are interfaces that can be instantiated for any task domain. A biology interface-portability audit is included in Appendix Q.

alignment, and expected-answer provenance. Feedback-contract completeness is enforced within the schema–checker alignment predicate. The three-part learnability predicate is therefore

$$q(\tau) = q_{\text{valid}}(\tau) \wedge q_{\text{align}}(\tau) \wedge q_{\text{prov}}(\tau). \tag{2}$$

The predicates are domain interfaces. In our mathematical experiments, domain validity becomes mathematical validity and the checker evaluates the required answer format; another domain can supply different validators without changing the recursive protocol.

Qualified generated tasks are executed by the active Harness, and their outcomes enter verified memory only after host-side checking yields a complete binary result. Eligible but unresolved proposals may be executed for exploration, but their outcomes remain provisional even if a lower-level checker returns a value. Known-invalid proposals and generation failures are not executed. Fixed seed tasks follow host verification directly because they are not self-generated proposals. Consequently, parseable output alone never constitutes verified evidence.

4.4 Verified-state consolidation

Let E_{r-1} denote the case-level verified experiences available before round r, and let ΔE_r be the newly verified cases and trajectories produced at round r. After learning executions, the bank is updated and distilled once:

$$E_r = E_{r-1} \cup \Delta E_r, \qquad \mathcal{P}_r = D(E_r, \mathcal{P}_{r-1}; \mathcal{O}_r), \qquad \mathcal{B}_r = (E_r, \mathcal{P}_r). \tag{3}$$

The observation store can provide explicitly marked context to distillation and Harness construction, but it is never promoted automatically into E_r. Thus, the Harness controller can consult both channels: host-verified correct and incorrect outcomes provide binary feedback through the verified state, while executable tasks with uncertain or unknown answer status provide provisional feedback through the observation store. Their records remain available as reference context with the

status preserved; they are not collapsed into one evidence class. Known-invalid tasks and generation failures are excluded before execution and therefore do not supply Harness feedback. During execution, retrieval selects bounded, task-relevant, lineage-deduplicated successes and failures from $\mathcal{B}_r$. This consolidated state is the task-to-Harness direction of co-evolution: the curriculum changes which verified cases, traces, and abstractions are available when the next executable procedure is constructed.

4.5 Complete Harness evolution and retention

Building on the executable Harness abstraction (Huang et al., 2026b), the controller rewrites complete source and configuration rather than editing a prompt alone. The configuration spans six surfaces: context assembly, tool interaction, generation control, orchestration, memory management, and output processing. Candidate generation is expressed as

$$C_r = K(H_{r-1}, \mathcal{B}_r, \mathcal{O}_r, \mathcal{A}_{r-1}), \tag{4}$$

where the separate $\mathcal{B}_r$ and $\mathcal{O}_r$ arguments expose the verified learning state and auxiliary observation context to the controller, and every member of C_r is an executable source/configuration pair. Search may alter any subset of the six surfaces; co-evolution does not require all dimensions to change in every round.

Every proposal is appended to $\mathcal{A}_r$ with its source, configuration, parent lineage, frozen bank reference, validity, resource metadata, and validation decision. Rejected candidates are retained for analysis, while only valid active or designated anchor branches are eligible future parents. Retention separates the history of explored Harnesses from the single branch used for execution.

4.6 Frozen promotion and held-out evaluation

Promotion compares each candidate and the incumbent on the same fixed tasks, bank snapshot, model, tools, and resource limits. All candidates first receive a common evaluation pass; repeated per-task comparisons are then allocated to the selected finalist set. Let $s(H; \mathcal{V}, \mathcal{B}_r)$ count validation tasks with complete, correct binary evidence, and let H_r^+ be the highest-scoring member of $\{H_{r-1}\} \cup C_r$. The active branch changes only under strict improvement:

$$H_r^+ = \arg \max_{H \in \{H_{r-1}\} \cup C_r} s(H; \mathcal{V}, \mathcal{B}_r), \tag{5}$$

$$H_r = \begin{cases} H_r^+, & \text{if evidence is complete and } s(H_r^+) > s(H_{r-1}), \\ H_{r-1}, & \text{otherwise.} \end{cases} \tag{6}$$

Validation reads but does not update memory. After R rounds, the final pair $(H_\star, \mathcal{B}_\star) = (H_R, \mathcal{B}_R)$ is frozen and evaluated once on $\mathcal{T}$. Held-out tasks cannot update the bank, generate new tasks, construct candidates, or affect selection, thereby separating evidence used to improve the system from evidence used to measure it.

5 Experiments

Setup. We organize the evaluation into a primary few-shot study and a complementary standard-scale study. The primary study uses AIME (2025–2026) and HMMT (2025–2026), where four seed tasks provide the initial supervision and the remaining tasks are separated into promotion-validation and disjoint held-out roles. The AIME benchmark is formed by pooling the AIME 2025 and AIME 2026 task collections under one fixed split, yielding forty-four held-out tasks; HMMT uses its corresponding 2025–2026 pooled split. GPQA-Diamond is evaluated separately at standard task volume to test whether the task-generation component can improve an evolving Harness beyond the few-shot setting. In GPQA, the task side changes the public question presentation while preserving the original answer choices and correct-answer mapping. Its 120-question training pool comprises 80 persistent Core positions and 40 Expansion candidates; each round retains the 80 Core positions and selects 20 Expansion questions using weaknesses exposed by the preceding Harness. The 18 validation questions and 60 held-out questions remain isolated from this selection loop. The co-evolution protocol freezes candidate selection before held-out evaluation. All primary solver runs use Claude Haiku 4.5 with medium reasoning effort; task generation and Harness adaptation therefore share a

fixed solver identity within each benchmark. Pure-LLM controls are evaluated without an evolving Harness or experience bank using GPT-5.5, GPT-5.6-Terra, GPT-5.6-Sol, Claude Opus 5, Claude Sonnet 5, and Claude Haiku 4.5, all with medium reasoning effort. We report pass@5, avg@5, and cons@5; coverage and infrastructure failures are retained separately.

Datasets. The primary few-shot evaluation pools AIME 2025 and AIME 2026 into one benchmark with four seed tasks, twelve validation tasks, and forty-four held-out tasks; generated tasks are used only during learning. Generated task proposals are conditioned on explicit Bloom targets (Apply, Analyze, or Create) and represented with a cognitive contract before qualification, making the task curriculum structured rather than arbitrary. HMMT follows the same few-shot role separation over its 2025 and 2026 task collections. The complementary GPQA-Diamond protocol uses a 120-question training pool (80 Core and 40 Expansion), an isolated 18-question validation set, and a 60-question held-out set. At each round, the preceding Harness's public failure evidence determines which 20 Expansion questions join the fixed 80-question Core panel; accepted task variants can replace their source Core positions while preserving the original answer mapping. This provides a standard-scale test of task-guided Harness adaptation. For all benchmarks, held-out questions are evaluated only after the final Harness and its bank snapshot are frozen. Appendix Q provides a complementary interface-level audit by instantiating the same Bloom-conditioned task contract in four biology domains.

Baselines. The principal comparison is against MemoHarness with the same solver identity, repeat schedule, and held-out metrics. We additionally report pure-LLM controls without Harness evolution or experience retrieval to distinguish recursive system improvement from base-model strength. The AIME comparison isolates joint Task–Harness evolution from Harness-only evolution in the few-shot regime; GPQA-Diamond tests the corresponding task-guided question-presentation strategy against Harness-only evolution at standard scale.

Main results. Table 1 foregrounds the paper's primary few-shot question by placing the pooled AIME 2025–2026 and HMMT 2025–2026 evaluations before the complementary standard-scale GPQA-Diamond result. The MemoHarness rows are protocol-consistent baselines evaluated with the same solver identity, held-out split, repeat schedule, and metrics. On AIME, Ours has higher avg@5, pass@5, and cons@5 than MemoHarness by 3.64, 4.55, and 6.82 points, respectively; the consistency difference reflects 36 versus 33 tasks with a correct unique-mode answer. On HMMT, the corresponding point-estimate differences are 2.13, 8.51, and 4.25 points, and Ours also exceeds the same-model pure-LLM control by 2.55 avg@5 points and 8.51 pass@5 points. On the larger standard-scale GPQA-Diamond setting, Ours has higher pass@5 and avg@5 than MemoHarness by 5.00 and 2.00 points. The pure-LLM rows are model-strength references rather than Harness baselines; each pure-LLM control is reported as completing the full scheduled question set. For AIME, task-level paired statistics are reported in Appendix B. The point estimates favor Ours; paired uncertainty intervals and tests are reported in the appendix.

5.1 Ablation Study

The ablation study identifies which information paths contribute to held-out performance, without elevating implementation safeguards into separate claims. The AIME context ablation uses the same 4/12/44 split, six-round schedule, solver identity (Claude Haiku 4.5, medium), 44-task held-out set, five samples per task, and held-out instruction-retrieval protocol as the reference AIME run (88.64 pass@5, 72.73 avg@5, 81.82 cons@5). During ablation training and promotion, retrieved solver experience is disabled; the frozen bank is explicitly injected only at held-out evaluation.

The task-evolution-disabled arm is the complete Harness-only ablation, evaluated with the same solver, AIME held-out split, and five-repeat schedule. It removes generated task proposals while retaining the fixed task set, Harness candidate search, frozen validation, and held-out evaluation. Its held-out evaluation is complete; the promotion count is not part of this control's reporting contract. The cross-round-context arm retains task generation and Harness search but withholds public observations and recent distilled patterns from the next generator call. The Bloom-conditioned generation ablation removes the explicit cognitive target from task proposals while retaining the rest of the held-out protocol. It completes all 220 episodes and obtains 84.09 pass@5, 66.82 avg@5, and 75.00 cons@5. This is the Bloom ablation of the main experiment: the task generator is unchanged in its execution and qualification protocol, but receives no explicit Bloom cognitive target.

Table 1: Held-out accuracy on the primary few-shot benchmarks and the complementary standard-scale benchmark. AIME and HMMT pool the 2025 and 2026 competitions; GPQA-Diamond is reported separately. Values are percentages. The AIME pure-Haiku control reports the original unrepaired evaluation.

Benchmark	Method / model	Regime	pass@5	avg@5	cons@5	Completed	Correct
AIME	**Ours (Claude Haiku 4.5, medium)**	few-shot	**88.64**	**72.73**	**81.82**	220	160
AIME	MemoHarness (Claude Haiku 4.5, medium)	few-shot	84.09	69.09	75.00	220	152
AIME	Pure Claude Haiku 4.5 (medium)	few-shot	88.64	68.64	79.55	220	151
Additional model-strength references are reported in Appendix C.							
HMMT	**Ours (Claude Haiku 4.5, medium)**	few-shot	**78.72**	**52.34**	**59.57**	235	123
HMMT	MemoHarness (Claude Haiku 4.5, medium)	few-shot	70.21	50.21	55.32	235	118
HMMT	Pure Claude Haiku 4.5 (medium)	few-shot	70.21	49.79	57.45	235	117
Additional model-strength references are reported in Appendix C.							
GPQA-Diamond	**Ours (Claude Haiku 4.5, medium)**	standard-scale	**86.67**	**66.67**	**66.67**	300	200
GPQA-Diamond	MemoHarness (Claude Haiku 4.5, medium)	standard-scale	81.67	64.67	66.67	300	194
GPQA-Diamond	Pure Claude Haiku 4.5 (medium)	standard-scale	83.33	63.00	61.67	300	189
Additional model-strength references are reported in Appendix C.							

Table 2: Three-layer ablation evidence under matched splits, solver identity, repeat schedule, and resource limits. The task-evolution-disabled row is the direct Harness-only test of the main claim; the cross-round and Bloom rows are mechanism controls, each removing one information path while retaining the remaining protocol.

Arm	pass@5	avg@5	cons@5	N	Evidence
Ours	88.64	72.73	81.82	220/220	Joint evolution
Task evolution disabled	84.09	69.09	75.00	220/220	Fixed-task Harness control
Cross-round context removed	81.82	69.09	75.00	220/220	No task feedback
Bloom conditioning removed	84.09	66.82	75.00	220/220	No cognitive targeting

Cross-round co-evolution feedback. The cross-round-context arm tests whether the task side can respond to public evidence produced by the current Harness. It preserves the same task proposal budget, qualification checks, candidate search, validation panel, and held-out solver protocol, but removes the public execution observations and distilled patterns supplied to the next generator call. Its pass@5, avg@5, and cons@5 decrease by 6.82, 3.64, and 6.82 percentage points, respectively, relative to Ours. This directly tests the feedback link that makes task and Harness updates interdependent rather than two independent alternating procedures.

Cognitive targeting. The Bloom arm preserves the same six-round curriculum schedule and qualification contract but leaves the cognitive target unspecified during generation. Its larger avg@5 decrease (5.91 points) than the task-evolution-disabled control is consistent with a useful role for explicit cognitive targeting in shaping the difficulty and operation mix of generated tasks, rather than merely labeling them after the fact. Figure A12 reports the two pairwise mechanism comparisons; the corresponding changes from Ours are shown in Figure A11.

The complete table shows that removing task evolution, cross-round public context, or Bloom-conditioned targeting lowers at least one held-out metric relative to Ours; all 220 held-out episodes completed for these controls. The corresponding ablation effect-size plot and absolute metric profile are provided in Appendix L.

The ablation evidence is organized in three layers. The direct Harness-only comparison tests whether task evolution is necessary for the main few-shot gain. The cross-round comparison tests whether the task side must condition its next proposals on public behavior of the current Harness. The Bloom comparison tests whether explicit cognitive targeting adds a further structuring signal once the feedback loop is present. Their round-level routing records provide descriptive context for the observed mechanism differences.

Figures A12 and A11 report the pairwise mechanism effects on a common percentage-point scale. The direct task-evolution comparison supports the main claim; the cross-round result supports the

feedback-path interpretation; and the Bloom result supports an additional role for cognitive structuring. These mechanism results are tied to the evaluated AIME split and solver identity.

5.2 Task Curriculum and Harness Selection Dynamics

The task side is a primary component of the proposed method, rather than a preprocessing step for Harness search. Figure A10 shows how the generated curriculum changes across learning rounds: each round contains an explicit mixture of cognitive targets, and each proposal is routed independently through qualification and execution outcomes. This makes visible both the content of the generated curriculum and the evidence boundary that determines which proposals can support subsequent adaptation. These task-side dynamics are the empirical counterpart of the task-evolution branch in the method overview.

The joint trajectory in Figure A9 then places the task-side routing beside frozen Harness selection across the same rounds. It uses immutable AIME round records and keeps qualification-passed, uncertain/exploratory, and execution-not-completed outcomes separate. Across six rounds, the run produces eight generated proposals per round. The paired validation panel reports the strongest complete finalist comparison retained by the frozen selection procedure; promotion markers identify rounds in which the active Harness changes. The figure is descriptive evidence of the coupled process.

The Bloom-conditioned proposal mix is shown separately in Figure A10. Each round contains four adjacent bar positions ordered Apply, Analyze, Create, and generation failure; the stacked segments report the downstream route. This view confirms that the cognitive contract is an explicit task-generation variable while qualification outcomes remain an independent evidence-routing decision.

Appendix Figure A5 audits the executable object being searched: all three candidates are retained in each round, while only the promoted lineage changes the active Harness.

The detailed AIME trajectory, state map, Bloom routing, and qualification funnel are reported in Appendix K, including the aggregate routing view in Figure A13; these are descriptive views of the same audited proposal records.

5.3 Case Study

The supplementary AIME case study traces one complete qualification, execution, candidate-validation, and promotion chain; it is an auditable process example rather than a per-task causal estimate. The triptych in Appendix A8 adds round-wise routing, a public lineage, and aggregate route counts.

The example is a generated problem with an Analyze target. Before execution it is checked for domain validity, schema–checker alignment, and expected-answer provenance. The target specifies an intended cognitive operation but does not certify correctness. The public record exposes the task, target, parent, and route labels, while answers and checker certificates remain private. Crucially, an executable proposal with uncertain or unknown answer status stays provisional; host-verified correct and incorrect outcomes are complete binary feedback; known-invalid and failed proposals are routed away. Thus generated, executed, and verified are distinct states.

The selected example receives a complete host verdict and enters the verified round state, which is then available to consolidation and complete source/configuration Harness construction. The same round produces three candidates; on the shared 12-task frozen validation panel, the selected finalist scores 10 correct tasks versus 8 for the incumbent. Strict improvement, not the single generated task, triggers promotion; rejected candidates remain archived and only the promoted branch becomes active.

The example exposes the public/private evidence boundary and the point at which task-side feedback can influence Harness construction. Aggregate effects are evaluated at the benchmark level in the main results.

6 DISCUSSION

The experiments support a mechanism-level interpretation of Task–Harness co-evolution. The task side is not merely a source of additional prompts: its public interface exposes the current Harness's observable behavior, and its qualified outcomes define which evidence can be used when constructing later Harness candidates. The round-wise records and mechanism comparisons are consistent with this division of roles. In particular, the task-evolution and cross-round-feedback controls address different questions: the former tests whether the learning situations themselves must adapt, while the latter tests whether task proposals must respond to the current Harness. The Bloom control examines whether an explicit cognitive target provides additional structure for the task-generation branch.

The protocol also clarifies what recursive improvement means in this setting. The base model remains fixed; improvement occurs through the executable Harness, its external learning state, and the qualified task distribution that supplies new evidence. Qualification is therefore a control boundary rather than a claim that every generated task is correct. Tasks with unresolved answer support can remain provisional and still expose public execution context, while only complete host-verified outcomes enter the verified learning state. This separation makes the feedback path explicit and keeps promotion validation and held-out evaluation read-only.

The standard-scale GPQA-Diamond study provides a complementary check of the task-side interface beyond the small mathematical benchmarks. It uses the same answer-preserving task transformation principle at a larger data scale, while the biology audit illustrates how the interface can be instantiated with domain-specific contracts. Together, these studies characterize the portability of the protocol interfaces across task specifications.

7 LIMITATIONS

Our empirical evidence is based on predefined benchmark partitions, a fixed solver identity, and a small number of few-shot benchmark families. The paired analyses quantify uncertainty for the reported fixed-split comparisons, but do not establish statistical generality over unseen partitions, models, or task distributions. The standard-scale study remains within question answering, and the biology audit is an interface audit rather than an additional held-out performance benchmark.

The qualification layer depends on the validators and response contracts available in a target domain. Exact-integer checkers make the current protocol particularly concrete, whereas interactive, open-ended, or multi-part outputs may require richer checker types and domain-specific evidence rules. Longer recursive horizons may also introduce failure modes that are not visible in the six-round evaluations. Extending the method across solvers, Harness families, domains, and independently sampled task partitions is an important direction for future work.

8 CONCLUSION

We introduced GrokRSI, which jointly evolves a generated task curriculum and an executable Harness from few-shot supervision. Public Harness observations shape later task proposals, qualified outcomes shape complete Harness search, and frozen validation separates selection from read-only held-out evaluation. Across the completed evaluations, the coupled process has higher held-out point estimates than Harness-only evolution with the same solver, while the mechanism ablations show the roles of cross-round feedback and cognitive targeting in the observed behavior. The paired analyses quantify uncertainty around these fixed-split differences. The biology audit further instantiates the same task-package and Bloom interfaces outside the mathematical examples. Together, the results support qualified task-side adaptation as a mechanism for recursive Harness improvement under scarce supervision. Extending the protocol to interactive or open-ended environments requires validators and feedback contracts appropriate to those domains.

REFERENCES

Lorin W. Anderson and David R. Krathwohl (eds.). *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*. Longman, 2001.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 41–48, 2009. doi: 10.1145/1553374.1553380. URL https://doi.org/10.1145/1553374.1553380.

Benjamin S. Bloom, Max D. Engelhart, Edward J. Furst, Walker H. Hill, and David R. Krathwohl (eds.). *Taxonomy of Educational Objectives: The Classification of Educational Goals, Handbook I: Cognitive Domain*. Longmans, Green, 1956.

Chrisantha Fernando, Dylan Banarse, Henryk Michalewski, Simon Osindero, and Tim Rocktäschel. Promptbreeder: Self-referential self-improvement via prompt evolution, 2023. URL https://arxiv.org/abs/2309.16797.

Qingyan Guo, Rui Wang, Junliang Guo, Bei Li, Kaitao Song, Xu Tan, Guoqing Liu, Jiang Bian, and Yujiu Yang. Evoprompt: Connecting llms with evolutionary algorithms yields powerful prompt optimizers, 2023. URL https://arxiv.org/abs/2309.08532.

Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuxiao Cheng, Ceyao Zhang, Zili Wang, Steven Zhang, Jialong Wang, Zhaoyang Ran, Xin Xiao, et al. Metagpt: Meta programming for multi-agent collaborative framework, 2023. URL https://arxiv.org/abs/2308.00352.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems, 2024. URL https://arxiv.org/abs/2408.08435.

Chengsong Huang, Zifeng Wang, Rujun Han, Jun Yan, Yanfei Chen, Zoey CuiZhu, Ke Jiang, Peng Xia, Han Yu, Yufan Zhuang, Yifei Ming, Jiaqi Pan, Bhavana Dalvi Mishra, Jiaxin Huang, Burak Gokturk, Tomas Pfister, and Chen-Yu Lee. Envharness: Awakening static worlds for agent learning, 2026a. URL https://arxiv.org/abs/2608.19880.

Yue Huang, Wenjie Wang, Han Bao, Yuchen Ma, Xiaonan Luo, Yi Nian, Haomin Zhuang, Zheyuan Liu, Yue Zhao, and Xiangliang Zhang. Memoharness: Agent harnesses that learn from experience, 2026b. URL https://arxiv.org/abs/2607.14159.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav R. Vardhan, Saiful Haq, Ashutosh Sharma, et al. Dspy: Compiling declarative language model calls into self-improving pipelines. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.03714.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step, 2023. URL https://arxiv.org/abs/2305.20050.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, et al. Agentbench: Evaluating llms as agents, 2023. URL https://arxiv.org/abs/2308.03688.

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models, 2023. URL https://arxiv.org/abs/2310.12931.

Aman Madaan, Niket Tandon, Peter Clark, et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36, 2023. URL https://arxiv.org/abs/2303.17651.

Michael Nguyen, Quoc Nguyen, and Paul Vuong. Recursive self-evolving agents via held-out selection, 2026. URL https://arxiv.org/abs/2606.28374.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. MemGPT: Towards LLMs as operating systems, 2023. URL https://arxiv.org/abs/2310.08560.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, 2023. doi: 10.1145/3586183.3606763. URL https://doi.org/10.1145/3586183.3606763.

Reid Pryzant, Dan Iter, Jerry Li, Yin Tat Lee, Chenguang Zhu, and Michael Zeng. Automatic prompt optimization with "gradient descent" and beam search, 2023. URL https://arxiv.org/abs/2305.03495.

Yujia Qin, Shi Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. Toolllm: Facilitating large language models to master 16000+ real-world apis, 2023. URL https://arxiv.org/abs/2307.16789.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36, 2023. URL https://arxiv.org/abs/2302.04761.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2023. URL https://arxiv.org/abs/2303.11366.

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. *Transactions on Machine Learning Research*, 2024a. URL https://arxiv.org/abs/2305.16291.

Peiyi Wang, Lei Li, Zhihong Shao, R. X. Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations, 2023a. URL https://arxiv.org/abs/2312.08935.

Xiaolong Wang, Zhe Zhao, Song Lai, Chaoli Zhang, Zijie Geng, Yu Tong, Ye Wei, and Qingsong Wen. From memorization to creation: Evaluating the cognitive depth of llm-generated educational questions. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2026. doi: 10.1145/3770854.3785686.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, pp. 13484–13508, 2023b. doi: 10.18653/v1/2023.acl-long.754. URL https://aclanthology.org/2023.acl-long.754/.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory, 2024b. URL https://arxiv.org/abs/2409.07429.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversation, 2023. URL https://arxiv.org/abs/2308.08155.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers, 2023. URL https://arxiv.org/abs/2309.03409.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering, 2024. URL https://arxiv.org/abs/2405.15793.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models, 2023a. URL https://arxiv.org/abs/2305.10601.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023b. URL https://arxiv.org/abs/2210.03629.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. tau-bench: A benchmark for tool-agent-user interaction in real-world domains, 2024. URL https://arxiv.org/abs/2406.12045.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models, 2023. URL https://arxiv.org/abs/2309.12284.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models, 2024. URL https://arxiv.org/abs/2401.10020.

Mert Yuksekgonul, Federico Bianchi, Marco Boehm, et al. Textgrad: Automatic "differentiation" via text, 2024. URL https://arxiv.org/abs/2406.07496.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35, 2022. URL https://arxiv.org/abs/2203.14465.

Zhiyuan Zeng, Hamish Ivison, Yiping Wang, Lifan Yuan, Shuyue Stella Li, Zhuorui Ye, Siting Li, Jacqueline He, Runlong Zhou, Tong Chen, Chenyang Zhao, Yulia Tsvetkov, Simon Shaolei Du, Natasha Jaques, Hao Peng, Pang Wei Koh, and Hannaneh Hajishirzi. Rlve: Scaling up reinforcement learning for language models with adaptive verifiable environments, 2025. URL https://arxiv.org/abs/2511.07317.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. AFlow: Automating agentic workflow generation, 2024. URL https://arxiv.org/abs/2410.10762.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. MemoryBank: Enhancing large language models with long-term memory, 2023. URL https://arxiv.org/abs/2305.10250.

Pei Zhou, Jay Pujara, Xiang Ren, Xinyun Chen, Heng-Tze Cheng, Quoc V. Le, Ed H. Chi, et al. Self-discover: Large language models self-compose reasoning structures, 2024. URL https://arxiv.org/abs/2402.03620.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents, 2023. URL https://arxiv.org/abs/2307.13854.

APPENDIX CONTENTS

A Algorithm and Implementation Details 14

B Main-result Visualization 14

C Additional Model-Strength Controls 14

D Case-Study Evidence Chain 14

E Examples of Generated Tasks 14

F Harness Evolution Structure 14

G Paired AIME Gains 14

H Supplementary HMMT Paired Analysis 16

I Supplementary Task-side Audits 16

J AIME Case Study 17

K Supplementary Round-wise Dynamics 17

L Supplementary Ablation Visualization 17

M Evaluation Boundaries 18

N Dataset and Metric Details 19

O Baseline Contracts 20

P Implementation and Reproducibility Details 20

Q Biology Interface Portability Audit 21

R Statement on AI-Assisted Tools 26

A Algorithm and Implementation Details

Table A1 summarizes the symbols used in the method. Algorithm 1 gives the complete round-level procedure. The implementation records qualification decisions, bank snapshots, candidate lineage, validation outcomes, and the role of every task used by the protocol.

B Main-result Visualization

Figure A1 provides a compact visual comparison of the three retained main-table rows on all benchmarks; the complete model-strength reference table follows below.

C Additional Model-Strength Controls

The main table retains Ours, MemoHarness, and the same-model Pure Claude Haiku reference because these rows directly support the Task–Harness claim. The remaining pure-LLM runs are model-strength references, not protocol-matched Harness baselines. They use medium reasoning effort and complete the full scheduled question set.

D Case-Study Evidence Chain

Figure A2 expands the case study in the main text into its auditable information path. The visualization includes only public task attributes, qualification outcomes, verified round-state changes, candidate search facts, and the frozen validation decision.

E Examples of Generated Tasks

The following examples make the task side of the method concrete. We expose only the public problem statement and public design metadata; private answers, checker certificates, and run-specific identifiers are omitted. The examples also illustrate that task generation is coupled to an explicit cognitive target and response contract rather than being unconstrained paraphrasing. Figure A3 gives the parent–child lineage, while Figure A4 prints the corresponding public task statements.

The lineage view also clarifies the routing distinction used throughout the method. Tasks with complete host-verifiable evidence can contribute binary feedback to the verified learning state. An executable task whose answer provenance or other evidence remains uncertain or unknown can still provide provisional context to the Harness controller, with that status preserved. Known-invalid proposals and generation failures are not executed and are absent from this table. The mathematical content is specific to the benchmarks used here; the qualification and evidence-routing interfaces are domain-independent. Table A3 summarizes the recorded resource units for the audited AIME run.

The learning-stage receipts cover generation, qualification, and controller support calls. The reported 184 logical calls and token totals provide the recorded accounting for the learning-stage protocol; held-out adaptation and solving stages are reported separately in the table.

F Harness Evolution Structure

Figure A5 audits the retained candidate archive and the source/configuration dimensions changed at each promotion.

G Paired AIME Gains

Figure A6 shows the task-level distribution and uncertainty underlying the paired AIME statistics in the main text, while Table A4 reports the corresponding point differences, intervals, and paired tests.

Table A1: Notation used by the proposed method.

Symbol	Meaning
$\mathcal{S}$	Fixed seed task set
$\mathcal{V}$	Frozen promotion-validation task set
$\mathcal{T}$	Isolated heldout task set
$H_r = (c_r, \theta_r)$	Active executable Harness at round r
$\mathcal{G}_r$	Generated task proposals at round r
$\mathcal{B}_r$	Verified case experience and pattern bank
$\mathcal{O}_r$	Provisional observation store
$\mathcal{A}_r$	Append-only candidate archive
$\mathcal{P}_r$	Distilled global patterns
C_r	Complete Harness candidates proposed at round r

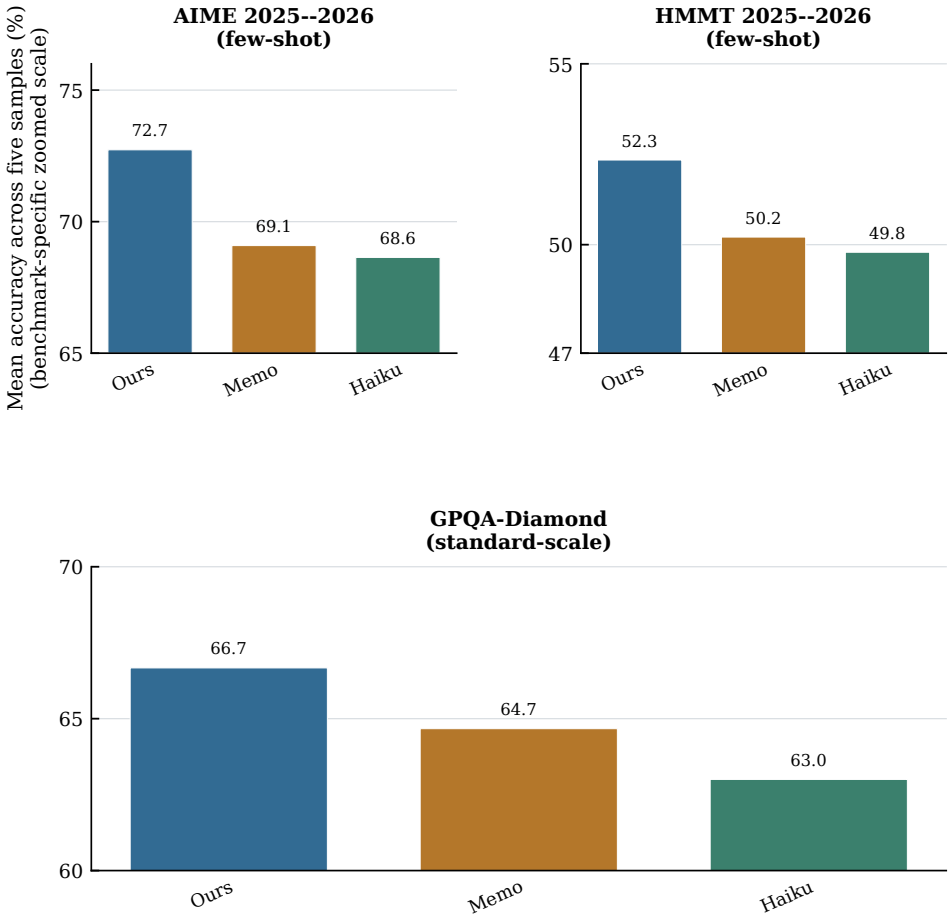


Figure A1: Held-out avg@5 under the respective evaluation protocols on completed benchmarks. The panels report pooled AIME 2025–2026, pooled HMMT 2025–2026, and standard-scale GPQA-Diamond, respectively. Ours, MemoHarness, and the same- model Claude Haiku 4.5 control are shown. The additional pure-LLM controls are reported in Table A2 but are omitted from this compact visualization. Each panel uses a benchmark-specific zoomed y-axis; values are percentages read from the tick labels.

Table A2: Additional pure-LLM model-strength references omitted from the main table.

Benchmark	Model	pass@5	avg@5	cons@5
AIME	GPT-5.5	97.73	87.73	95.45
AIME	GPT-5.6-Terra	90.91	74.55	84.09
AIME	GPT-5.6-Sol	93.18	80.91	88.64
AIME	Claude Opus 5	97.73	76.82	90.91
AIME	Claude Sonnet 5	95.45	69.55	77.27
HMMT	GPT-5.5	95.74	77.45	91.49
HMMT	GPT-5.6-Terra	89.36	72.77	80.85
HMMT	GPT-5.6-Sol	91.49	74.04	80.85
HMMT	Claude Opus 5	100.00	74.04	82.98
HMMT	Claude Sonnet 5	91.49	68.09	78.72
GPQA-Diamond	GPT-5.5	96.67	82.33	86.67
GPQA-Diamond	GPT-5.6-Terra	93.33	81.00	83.33
GPQA-Diamond	GPT-5.6-Sol	95.00	86.00	88.33
GPQA-Diamond	Claude Opus 5	91.67	74.00	78.33
GPQA-Diamond	Claude Sonnet 5	88.33	72.00	76.67

Algorithm 1 Verified-experience-mediated Task–Harness co-evolution.

Require: Seed tasks $\mathcal{S}$, validation panel $\mathcal{V}$, heldout set $\mathcal{T}$, initial Harness H_0

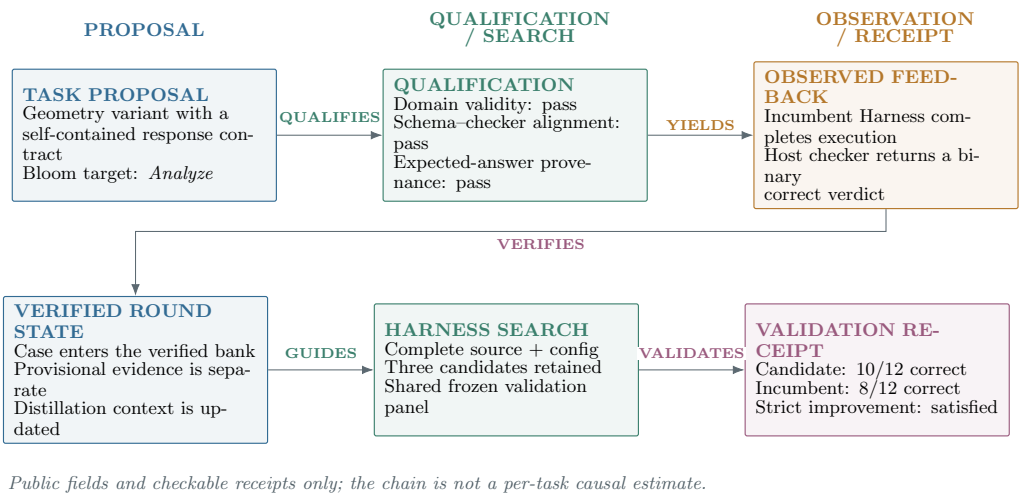
Initialize $\mathcal{B}_0 \leftarrow \varnothing$, $\mathcal{O}_0 \leftarrow \varnothing$, $\mathcal{A}_0 \leftarrow \{H_0\}$
for round $r = 1, \ldots, R$ **do**
  Generate proposals $\mathcal{G}_r \leftarrow G(\mathcal{S}, \mathcal{L}_{r-1}, \mathcal{O}^{\mathrm{pub}}_{r-1}, \mathcal{P}_{r-1})$
  Apply the three qualification predicates to each $\tau \in \mathcal{G}_r$, with feedback-contract completeness enforced within schema–checker alignment
  Execute qualified learnable tasks and eligible seeds/replays with H_{r-1}
  Record qualification and execution decisions in $\mathcal{O}_r$
  Collect newly host-verified binary outcomes ΔE_r
  Update verified state $E_r \leftarrow E_{r-1} \cup \Delta E_r$
  Distill once: $\mathcal{P}_r \leftarrow D(E_r, \mathcal{P}_{r-1}; \mathcal{O}_r)$, set $\mathcal{B}_r \leftarrow (E_r, \mathcal{P}_r)$
  Propose complete candidates $C_r \leftarrow K(H_{r-1}, \mathcal{B}_r, \mathcal{O}_r, \mathcal{A}_{r-1})$
  Append all candidates and their source, configuration, lineage, and bank reference to $\mathcal{A}_r$
  Evaluate $H_{r-1} \cup C_r$ on the frozen panel $\mathcal{V}$
  Promote only a candidate with complete evidence and strictly more correct tasks; otherwise set $H_r \leftarrow H_{r-1}$
end for
Freeze $(H_\star, \mathcal{B}_\star) \leftarrow (H_R, \mathcal{B}_R)$
Evaluate the single frozen pair on $\mathcal{T}$ without write-back

H Supplementary HMMT Paired Analysis

For completeness, we report the task-paired HMMT comparison in Table A5. The analysis uses the same 47 held-out tasks and five scheduled repeats per task for both conditions. These results are supplementary benchmark-split evidence; the main text uses AIME as the primary paired uncertainty analysis.

I Supplementary Task-side Audits

The following figures provide detailed task-side views of the six-round proposal records summarized in the main text. Figure A7 gives the round-by-round state map; the accompanying funnel and Bloom routing views report the same records at different levels of aggregation.



Public fields and checkable receipts only; the chain is not a per-task causal estimate.

Figure A2: Audited AIME few-shot evidence chain. A generated task passes the three qualification predicates, receives host-verified binary feedback, and enters the round state used to search complete Harness candidates. The selected candidate is promoted only after comparison with the incumbent on the same frozen validation panel. The chain documents an auditable information path; aggregate effects are evaluated at the benchmark level.

Table A3: Audited AIME resource accounting at the granularity supported by the receipts. Learning-stage controller receipts and held-out solver stages are reported as separate resource units; the held-out total is included for direct budget accounting.

Phase	Recorded resource unit	Calls	Input tokens	Output tokens	Evidence scope
Learning (6 rounds)	Generation / qualification / controller receipts	184	740,907	405,779	recorded receipts
Held-out (220 episodes)	Adaptation	220	552,505	285,887	per-episode records
Held-out (220 episodes)	Solving	242	695,312	2,974,145	per-episode records
Held-out (220 episodes)	Total	462	1,247,817	3,260,032	220 episodes

J AIME Case Study

In the first learning round, a generated geometry task was assigned an Analyze target, passed domain-validity, schema–checker alignment, and expected-answer provenance checks, and received a completed host-verified binary outcome. The same round evaluated complete source-and-configuration Harness candidates on a shared 12-task frozen validation panel. The selected candidate obtained 10 correct tasks versus 8 for the incumbent and was promoted under the strict improvement rule. Figure A2 reports the public evidence chain without disclosing answers used by the checker.

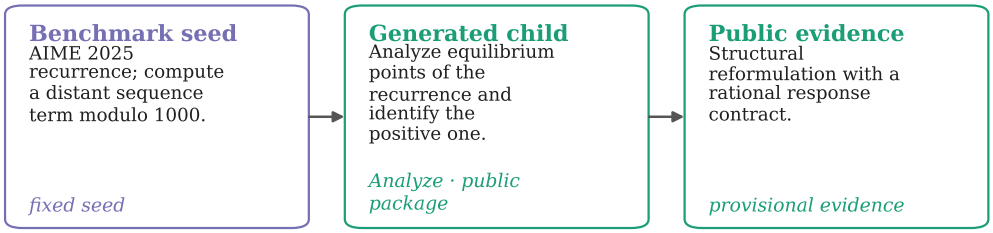
K Supplementary Round-wise Dynamics

This section resolves the temporal structure of the audited co-evolution run. The first view separates proposal routing from Harness promotion, and the second view decomposes the same records by Bloom target and qualification route. Together they provide descriptive context for the mechanism results without introducing an additional performance claim.

L Supplementary Ablation Visualization

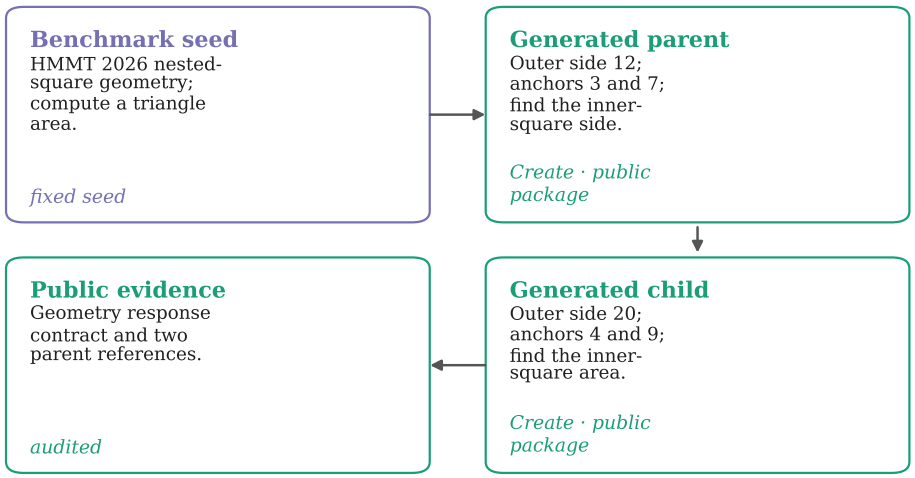
The following visualizations report the completed AIME mechanism controls on a common percentage-point scale. The first figure shows changes relative to Ours; the second preserves absolute scores for the two information-path comparisons. The direct Harness-only comparison remains in the ablation table.

AIME lineage: benchmark seed to Analyze task



Initial seed: compute a distant term of a rational recurrence modulo 1000.
Final generated task: analyze equilibrium points of the recurrence; output an integer.

HMMT lineage: benchmark seed to Create descendants



Initial seed: nested-square configuration; compute the area of a derived triangle.
Final generated task: changed geometry and target quantity; compute inner-square area.

Figure A3: Public generated-task lineage examples. The arrows follow the recorded parent references from an original benchmark seed to one or more generated descendants. The examples use different Bloom targets to expose both constructive reformulation (AIME, Create) and relational analysis (HMMT, Analyze). Only public task statements and public evidence labels are shown; private answers and checker certificates are omitted.

M Evaluation Boundaries

The three task roles enforce distinct information flows. Seed and qualified generated tasks may update the learning state; promotion-validation tasks may select among frozen candidates but cannot

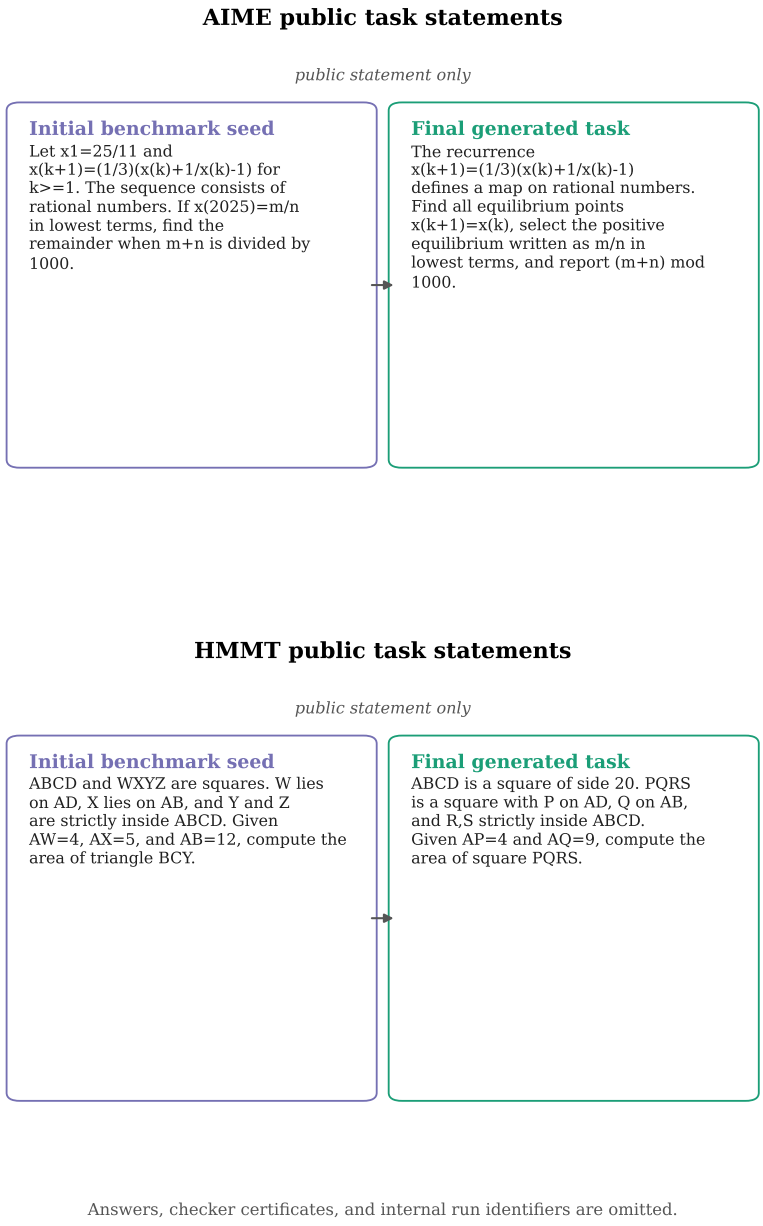


Figure A4: Full public task statements for the seed-to-generation examples in Figure A3. Each row contrasts the original benchmark seed with the final generated statement used in the lineage example. The figure shows the public problem text and its task-level design metadata; answer values and checker certificates are represented by the evidence labels.

update memory; held-out tasks may only produce final evaluation records. Results are compared only when solver identity, task split, repeat schedule, and metric definitions match. This role separation prevents validation or held-out outcomes from becoming inputs to a later recursive round.

N Dataset and Metric Details

GPQA-Diamond uses a standard-scale 120-question training pool consisting of 80 persistent Core positions and 40 Expansion candidates, together with an isolated 18-question validation set and a 60-question held-out set. Each learning round retains the 80 Core positions and selects 20 Expansion

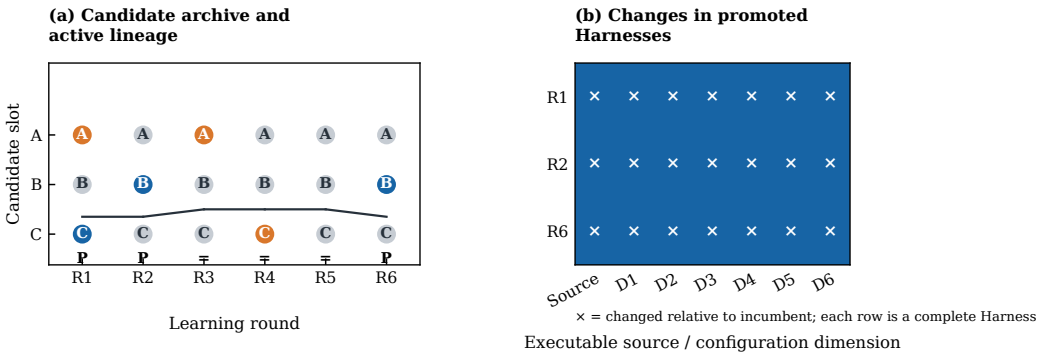


Figure A5: Structure of Harness evolution across six learning rounds. (a) Three complete executable candidates are generated and retained in the archive at each round; the highlighted lineage marks the active Harness. (b) Source and configuration dimensions changed by each promoted Harness relative to its incumbent.

Table A4: Task-paired AIME 2025–2026 comparison of Ours and MemoHarness.

Metric	Difference (pp)	Task-bootstrap 95% CI (pp)	Paired test
pass@5	+4.55	$[0.00, 11.36]$	Exact McNemar $p = 0.500$
avg@5	+3.64	$[-1.36, 9.09]$	Sign-flip $p = 0.250$
cons@5	+6.82	$[-2.27, 15.91]$	Exact McNemar $p = 0.375$

questions from the training pool using weaknesses exposed by the preceding Harness. Task-side variants alter the public question presentation while preserving the original choices and correct-answer mapping. AIME pools the 2025 and 2026 competitions into the few-shot role separation described in the main text, with four seed tasks, a frozen validation panel, and forty-four disjoint held-out tasks; HMMT uses the corresponding 2025–2026 few-shot construction. For a held-out task i, let $y_{i,j} \in \{0, 1\}$ denote correctness on repeat j. We report

$$\text{avg@5} = \frac{1}{5|\mathcal{T}|} \sum_{i \in \mathcal{T}} \sum_{j=1}^{5} y_{i,j}, \qquad \text{pass@5} = \frac{1}{|\mathcal{T}|} \sum_{i \in \mathcal{T}} \mathbf{1}\left[\sum_{j} y_{i,j} > 0 \right].$$

The consistency rate cons@5 is the fraction of held-out tasks for which the five sampled answers have a unique modal answer and that modal answer is correct. This task-level metric differs from the all-five-correct rate. Coverage is reported as completed binary samples divided by the scheduled samples.

O Baseline Contracts

MemoHarness is evaluated under the protocol-consistent comparison contract: the solver identity, held-out split, repeat schedule, and metric definitions match Ours within each benchmark, while the Harness-only procedure remains the comparison treatment. Pure-LLM controls use no evolving Harness and no experience bank. AIME and HMMT are the primary few-shot evidence, while GPQA-Diamond is a complementary standard-scale result. The main table excludes acceptance or canary results.

P Implementation and Reproducibility Details

The executable Harness is a source/configuration pair. The configuration controls context assembly, tool interaction, generation control, orchestration, memory management, and output processing. Each learning round records task lineage, qualification decisions, verified-bank updates, provisional

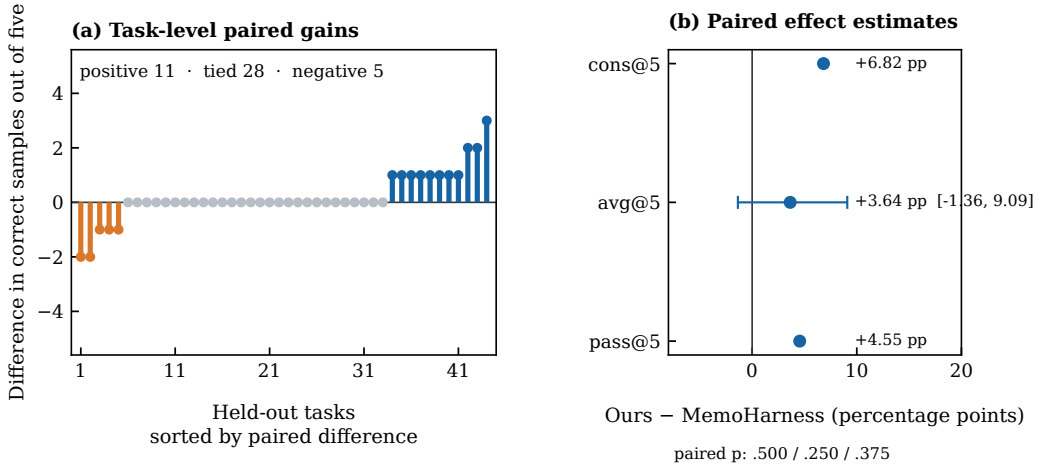


Figure A6: Task-level paired comparison on the pooled AIME 2025–2026 held-out set. (a) Difference between Ours and MemoHarness in correct outcomes across five samples for each of 44 matched tasks, sorted by paired difference. (b) Paired effect estimates in percentage points; the avg@5 interval is a task-level bootstrap interval and the displayed paired tests are not statistically significant at the conventional 0.05 level.

Table A5: Supplementary task-paired comparison of Ours and MemoHarness on HMMT 2025–2026. The interval and tests resample or compare the 47 tasks, rather than treating the 235 repeated samples as independent observations. None of the paired tests reaches the conventional 0.05 level.

Metric	Difference (pp)	Paired test / 95% interval
pass@5	+8.51	Exact McNemar $p = 0.125$
avg@5	+2.13	$[-3.83, 7.66]$; sign-flip $p = 0.568$
cons@5	+4.25	Exact McNemar $p = 0.754$

observations, candidate source/configuration, parent lineage, bank reference, validation evidence, and promotion status. Held-out evaluation reads one frozen Harness and one frozen bank snapshot and performs no write-back. Machine-readable manifests associate every reported aggregate with its task split, solver identity, repeat schedule, and evaluation records.

Q Biology Interface Portability Audit

This appendix reports a complete task-level portability audit for the Bloom-conditioned generation interface. The generator prompt, schema gate, checker-facing fields, and single-repair policy were kept unchanged; only the domain wording and Bloom goal definitions were changed from mathematics to biology. Four displayed cases cover molecular biology, physiology, vaccine formulation, and genetics. Each case records the parent and child statements, changed conditions, Bloom target, response contract, and expert recomputation.

The audit evaluates portability of the task-package and Bloom contracts. Table A6 summarizes the four parent–child cases and their independent expert checks. The expert review checks that each child answer follows from its own conditions and that the requested cognitive operation is represented. The vaccine case also exposes an interface limitation: a multi-part expert answer required quantization to the exact-integer checker, indicating where richer checker types would be useful.

The generation path is parent problem and Bloom target, generator, schema gate, child problem, and expert review. A schema failure triggers one repair call without changing the checker contract.

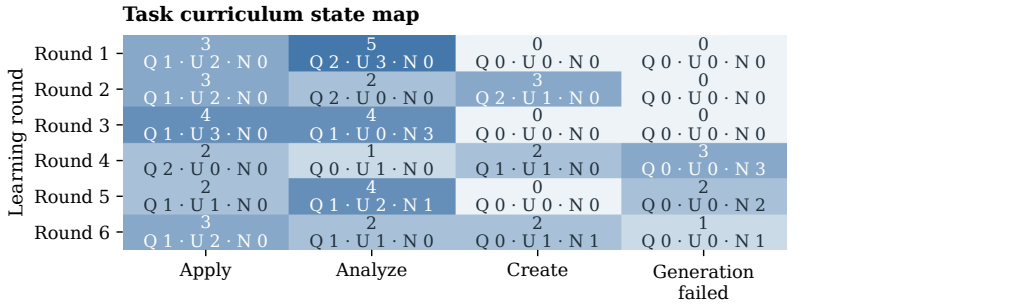


Figure A7: AIME task curriculum state map across the six audited learning rounds. Each cell reports the proposal count for one Bloom target or generation-failure outcome, followed by qualified (Q), unresolved (U), and not-executed (N) routing counts. Each round contains eight proposal slots.

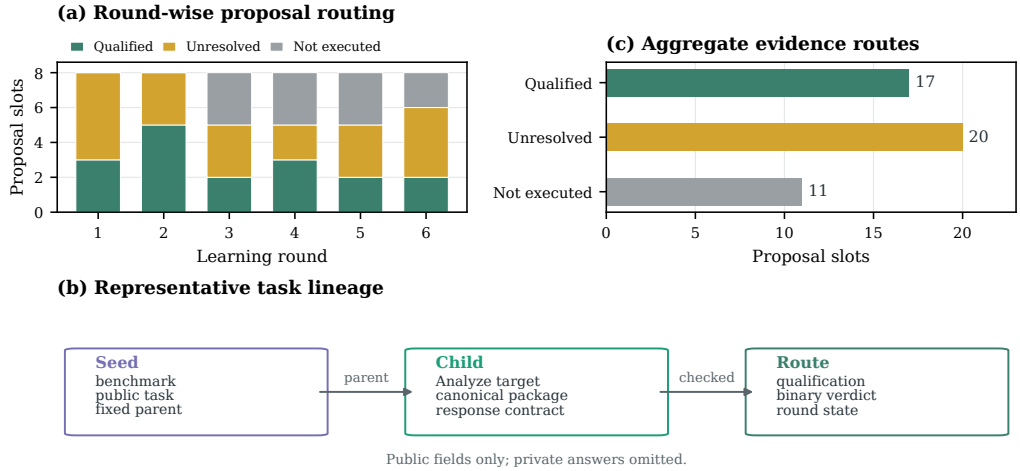


Figure A8: Task-centered AIME case-study summary. (a) Audited proposal routes across the six learning rounds; (b) a representative public seed-to-child lineage and its recorded evidence route; and (c) aggregate routing counts over 48 proposal slots. The figure is descriptive: it documents the task-side protocol and evidence boundary without claiming that an individual generated task causes the aggregate held-out improvement.

Figure A14 makes this interface explicit, and Figure A15 visualizes the two independent expert recomputations used in the molecular-biology and vaccine-formulation cases.

Molecular biology. The child rescales the gene length, insertion position, and stop position while preserving the frameshift mechanism. Codon m spans positions $3m-2$ through $3m$; a stop ending at 90 is codon 30 and therefore leaves 29 translated codons. This independently recovers the child answer and shows that the answer was not obtained by rescaling the parent.

Physiology. The child adds a clinical context, halves the collection window, and changes all measurements. The clearance calculation is

$$C_{\mathrm{Cr}} = \frac{U_{\mathrm{Cr}} V}{P_{\mathrm{Cr}} t} = \frac{80 \times 720}{1.5 \times 720} = \frac{160}{3}\ \mathrm{mL/min},$$

whereas the parent gives $90 \times 1440/(1.2 \times 1440) = 75$ mL/min.

Vaccine formulation. The expert-written parent uses two preparations and separate per-injection constraints. The child creates a three-component formulation with a shared dose and aggregate

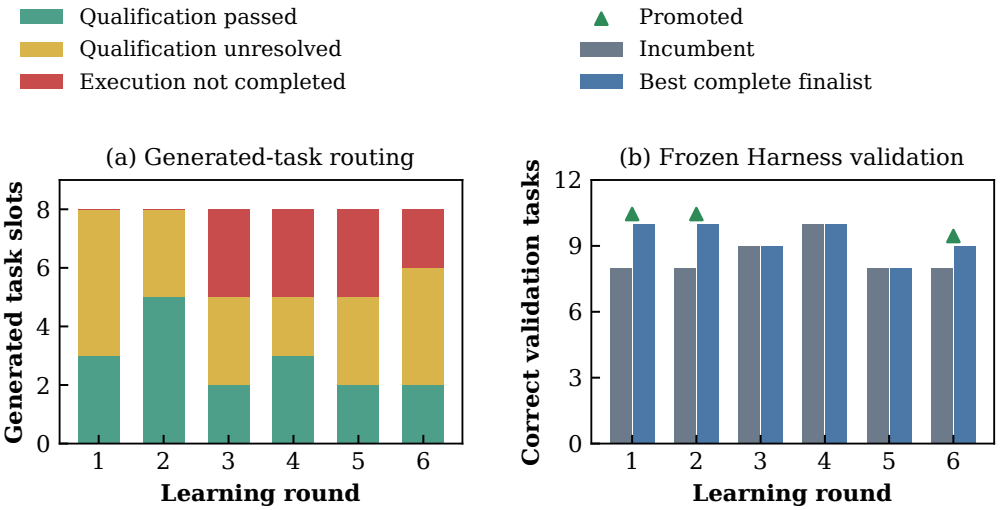


Figure A9: Round-wise Task–Harness trajectory in the audited six-round AIME run. (a) Each round contains eight generated proposals, separated into qualification-passed, qualification-unresolved, and execution-not-completed routes. (b) The strongest complete finalist is shown against its paired incumbent on the frozen validation panel; triangles mark rounds that promote a new active Harness.

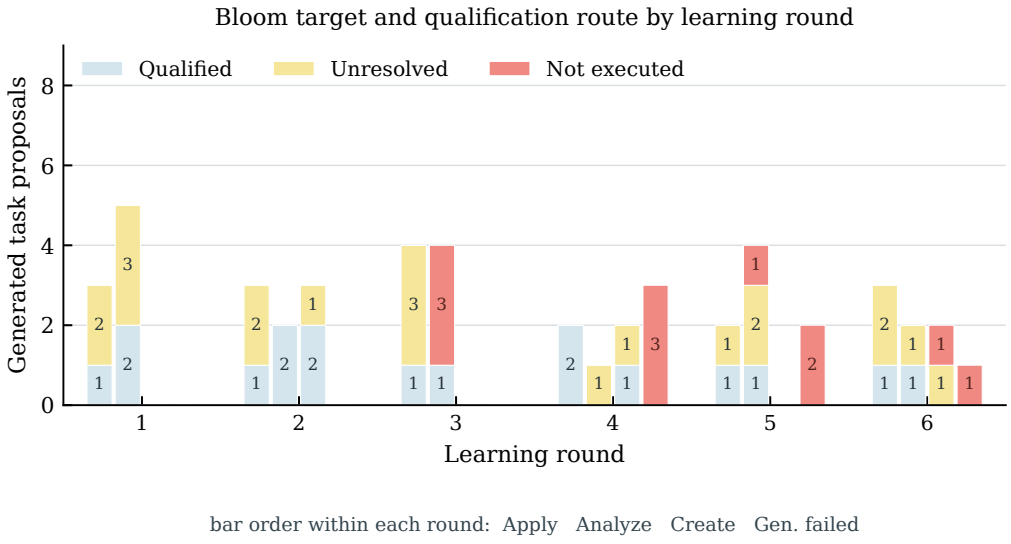


Figure A10: AIME task-curriculum dynamics across the six learning rounds. Bars show Bloom targets and downstream qualification routes.

constraints. The particle ratios are 200, 1000, and 2500 particles/FU, so the particle constraint gives $3700D \leq 5.0 \times 10^{10}$; the volume constraint gives a weaker bound. Quantising D in units of 2.5×10^{6} FU yields $X = 5$. This case identifies a checker-extension opportunity: the expert parent returns a feasibility flag plus two volumes, while the current checker expresses an exact integer. The generator preserves gradability by quantising the dose; richer biology answer forms can be supported by extending the checker contract.

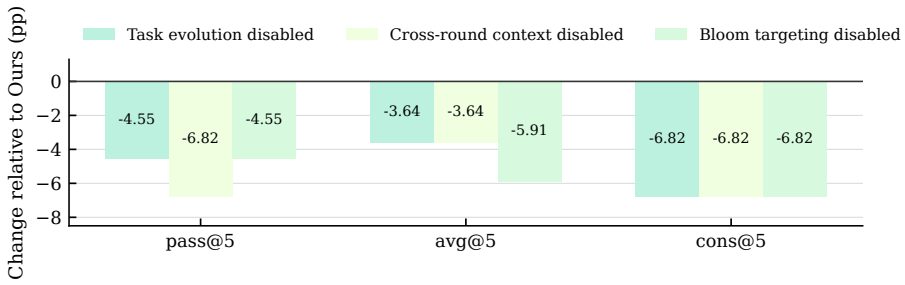


Figure A11: Completed AIME ablation effects relative to Ours. Bars show the change in pass@5, avg@5, and cons@5 for the task-evolution-disabled, cross-round-context, and Bloom-target-disabled conditions; negative values indicate a reduction from the reference co-evolution run.

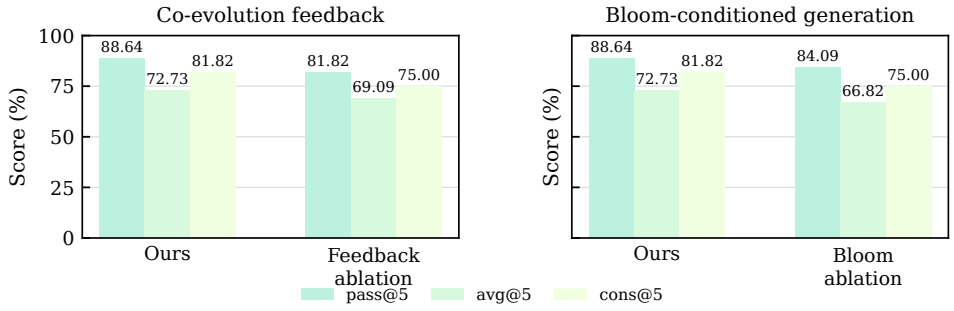


Figure A12: Pairwise absolute AIME performance profiles for the two mechanism controls. The left panel compares the main experiment with the cross-round co-evolution-feedback ablation; the right panel compares the main experiment with the Bloom ablation. The direct task-evolution-disabled comparison is reported in Table 2.

Genetics and retained records. The genetics child changes the population-size term while preserving the inheritance calculation, yielding $1/9$ from its own conditions. The ecology record and the additional HMMT generation record remain in the machine-readable package and are included in the audit inventory, although the four cases above are the expert-reviewed display set.

Additional machine-readable generation records and configuration files are released as supplementary artifacts alongside the paper.

Each machine-readable record also retains the structured cognitive contract: required actions, minimum evidence, shortcut risks, anchor relations, change claim, response schema, verification artifact, and negative examples. These fields make the generation constraint and checker-facing evidence explicit and support independent reconstruction of the audit.

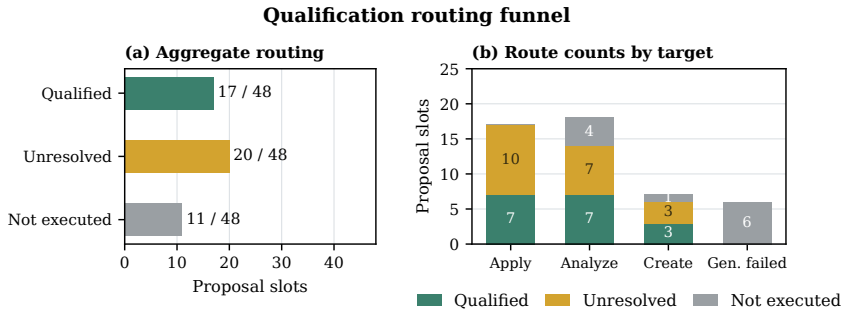


Figure A13: AIME qualification routing funnel over 48 audited generated proposal slots. The aggregate routes are 17 qualified, 20 unresolved, and 11 not executed; the second panel decomposes these routes by Bloom target and generation failure. Together, the panels summarize the task-side evidence-routing distribution.

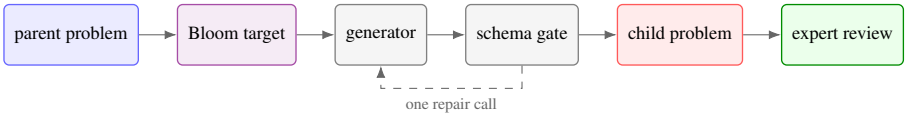


Figure A14: Biology interface-portability pipeline. The parent task, abstract Bloom target, generator, schema gate, child task, and expert recomputation are explicit interfaces; the generation code, checker fields, and repair policy remain unchanged.



Figure A15: Expert recomputation checks. Left: codon indexing after the insertion yields 29 translated residues. Right: the aggregate particle constraint accepts $X = 5$ but rejects $X = 6$.

Table A6: Biology interface portability audit. Parent answers and child answers were independently recomputed from their respective conditions.

Case	Bloom transition	Parent conditions and answer	Child conditions and answer	Review
Molecular biology	Apply → Apply	300-aa gene, insertion after nucleotide 30, stop 45 nt later; 24 aa	180-aa gene, insertion after nucleotide 60, stop at new position 90; 29 aa	✓
Physiology	Apply → Analyze	$P = 1.2$, $U = 90$, 1440 mL/24 h; 75 mL/min	$P = 1.5$, $U = 80$, 720 mL/12 h; 160/3 mL/min	✓
Vaccine formulation	Analyze → Create	Two preparations, separate injections; proposed 2×10^7 FU infeasible; maximum 10^7 FU	Three preparations in one injection; aggregate particle and volume limits; maximum integer dose $X = 5$	✓
Genetics	Analyze → Analyze	$(2/3)(98/2499)(1/4) = 1/153$	$(2/3)^2(1/4) = 1/9$	✓

R STATEMENT ON AI-ASSISTED TOOLS

In this work, generative AI tools were used only to improve the language, clarity, and readability of the manuscript. They were not used to design the methodology or experiments, generate or clean data, formulate mathematical claims, interpret results, implement the method, or create or modify scientific figures. The authors reviewed all AI-assisted edits and verified the final text, claims, references, and artifacts. The authors take full responsibility for the final content of this work.