INVERSEHARNESS: INTERNALIZING AGENTIC REASONING THROUGH REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

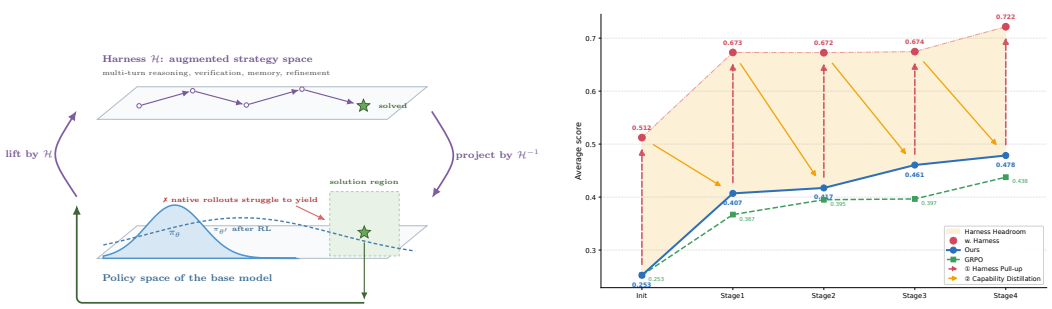
Reinforcement learning (RL) improves the reasoning capability of large language models (LLMs) by searching and then memorizing successful trajectories, but progress can stall when effective strategies are rarely discovered due to the difficulty of problems through native sampling. Agentic harnesses can elicit stronger reasoning through external orchestration, yet their executions are not directly usable as standalone training trajectories. We introduce HAIR (Harness-Assisted Internalization via Reinforcement Learning), which combines an agentic harness with InverseHarness to reconstruct successful executions into self-contained trajectories while preserving the current policy's generated reasoning. Interleaved with standard RL, this process targets problems where native rollouts fail and regenerates training trajectories from the evolving policy. On mathematical reasoning with Intern-S2-Preview at 35B and 397B, HAIR outperforms GRPO across most evaluated benchmarks under matched backbones, training corpora, and optimizer steps. At 397B, pass@1 increases from 55.5 to 68.8 on IMO-Proof-Bench and from 29.2 to 38.4 on PhD-Paper. Across rounds, harnessed performance improves with the policy, while training on successful executions improves standalone reasoning, supporting a closed loop of self-improvement. Further analysis shows that generating trajectories from the current policy matters beyond correctness alone. These findings establish harness-assisted training as a promising route to self-improvement beyond native sampling.

1 INTRODUCTION

Reinforcement learning (RL) has emerged as a central paradigm for improving large language models without proportionally increasing human supervision or relying on increasingly capable teacher models, yielding substantial gains in reasoning, compositional generalization, trial-and-error problem solving, and self-correction (Guo et al., 2025; Shao et al., 2024; Team, 2025; Lyu et al., 2025; Shen et al., 2025). Many current RL methods for reasoning repeatedly sample trajectories from the current policy, evaluate them with outcome-based rewards, and update the policy toward higher-reward behaviors. Best-of-N (BoN) sampling provides a simple abstraction of this process. Repeated sampling and reinforcement can sharpen the policy distribution and uncover more effective compositions of reasoning primitives already available to the model.

However, self-improvement through native sampling faces a fundamental limitation: the model must discover better strategies using the same policy it is trying to improve. Harder problems may require qualitatively different ways of organizing computation, such as decomposition, parallel exploration, verification, and backtracking. These strategies depend on coordinated sequences of decisions and are therefore unlikely to emerge spontaneously from the current policy (Yue et al., 2025; Prakash & Buvanesh, 2025; Sinha et al., 2025). This reveals a *policy-improvement operator bottleneck*: even when such strategies lie within the model's capacity, a BoN-based improvement operator may fail to discover them, eventually causing learning to saturate. Recent work addresses this bottleneck by preserving rare branches (Cui et al., 2025b; Yu et al., 2025), keeping training near the competence boundary (Shi et al., 2025; Li et al., 2026), or importing offline expert trajectories (Yan et al., 2025; Zhao et al., 2026). However, few of these methods directly address the challenge of reliably sampling qualitatively stronger strategies from the current policy itself.

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) The lift–project–absorb cycle.

(b) Iterative self-improvement during RL.

Figure 1: **Inverse Harness: internalizing agentic reasoning through reinforcement learning.** (a) For a hard problem requiring advanced reasoning strategy, native rollouts from the current policy π_θ may struggle to yield successful trajectories. An agentic harness $\mathcal{H}$ *lifts* the policy into an externally orchestrated reasoning process, enabling strategies that native sampling rarely discovers. The inverse harness $\mathcal{H}^{-1}$ *projects* successful executions into self-contained trajectories while preserving the current policy's generated reasoning. RL then *absorbs* these trajectories into the updated policy $\pi_{\theta'}$, improving standalone reasoning. (b) Across training rounds, harnessed performance improves with the evolving policy, while training on reconstructed trajectories improves standalone performance, supporting a closed loop of self-improvement.

Recently, agent harnesses have been widely used to improve inference-time performance by reorganizing computation through parallel exploration, critique, verification, refinement, memory, and explicit orchestration, including for challenging reasoning tasks in competition- and research-level mathematics (Gao et al., 2023; Gou et al., 2024; Huang & Yang, 2025; Feng et al., 2026; Gu et al., 2025; Chen et al., 2025a). This enables a fixed underlying model to execute problem-solving strategies that are difficult to realize within a single native trajectory, expanding the set of effective computations it can access. Such harnesses seem possible to serve as stronger policy-improvement operators for RL, but it raises a natural question: *can improvements induced by such an external operator be internalized into the model itself during RL?*

The central challenge is that successful harness executions depend on external orchestration and intermediate state, making them unsuitable for direct use as standalone reasoning trajectories. We address this gap with HAIR (Harness-Assisted Internalization via Reinforcement Learning), a training framework whose key component, InverseHarness, reconstructs successful executions into self-contained single-turn trajectories while preserving the reasoning generated by the current policy. Together, the harness, InverseHarness, and RL update form a lift–project–absorb cycle: the harness enables the policy to execute externally orchestrated reasoning strategies, InverseHarness projects successful executions into standalone trajectories, and RL trains the policy on these trajectories. This approach follows the expert-iteration paradigm (Anthony et al., 2017; Zelikman et al., 2022; Singh et al., 2024; Yu et al., 2024), with the composition of an agentic harness and InverseHarness serving as the policy-improvement operator. Crucially, the harness is applied to the evolving policy, so improvement trajectories are generated anew from the updated model in each round. We periodically interleave this cycle with standard RL, targeting problems for which native rollouts yield no successful trajectories in the current training round.

We evaluate HAIR on mathematical reasoning with Intern-S2-Preview at 35B and 397B, using Intern-S1-MO (Gu et al., 2025) as the harness. Under matched backbones, training corpora, and optimizer steps, HAIR outperforms GRPO across most evaluated benchmarks at both scales, with the largest gains on harder tasks. At 397B, it raises pass@1 from 55.5 to 68.8 on IMO-Proof-Bench and from 29.2 to 38.4 on PhD-Paper. Across training rounds, harnessed performance improves with the underlying policy, while training on the resulting successful trajectories enables the standalone model to solve previously unsolved problems, supporting a closed loop of self-improvement. Further analysis shows that trajectory correctness alone is insufficient: generation from the current policy also matters for effective RL. Qualitative analysis reveals changes in the trained policy's reasoning behavior, including strategy switching and self-verification. These findings support harness-driven training as a route to self-improvement beyond trajectories readily discovered by native sampling. In conclustion, our contributions are:

- We introduce HAIR, an iterative RL framework that uses an agentic harness to improve the evolving policy and internalizes harness-induced improvements into standalone reasoning.
- We develop InverseHarness that reconstructs successful harness executions into self-contained reasoning trajectories while preserving the current policy's own generated reasoning, enabling externally orchestrated computation to support policy training.
- We demonstrate consistent gains over GRPO at different model scales, provide evidence of closed-loop self-improvement across training rounds, and show that generating training trajectories from the current policy matters beyond correctness alone.

2 METHOD

We propose HAIR (Harness-Assisted Internalization via Reinforcement Learning), an iterative framework following a lift–project–absorb cycle: an agentic harness lifts the current policy into an externally orchestrated reasoning process; InverseHarness projects successful executions into self-contained trajectories while preserving policy-generated reasoning; and auxiliary supervised updates absorb these trajectories into the model. We periodically interleave this cycle with standard GRPO, targeting problems for which native sampling yields no successful trajectories. Each cycle generates new trajectories from the evolving policy.

2.1 HARNESS AND INVERSEHARNESS

Let π_θ denote the current policy and x a problem. An agentic harness $\mathcal{H}$ repeatedly queries π_θ under an external orchestration protocol, producing

$$\mathcal{T} = \mathcal{H}(x; \pi_\theta) = \{(s_i, c_i, m_i)\}_{i=1}^{N}, \tag{1}$$

where s_i is a policy-generated reasoning segment, c_i its harness role (e.g., solving, verification, or repair), and m_i the intermediate state recorded at that step, including lemmas, partial results, and verification conclusions. This abstraction covers tool-free harnesses with iterative solving, verification, memory, parallel exploration, and repair. DAG-structured executions can be processed in a topological order that respects their dependencies.

Successful executions may contain protocol-related content, references to external state, and repeated derivations. InverseHarness, denoted by $\mathcal{H}^{-1}$, reconstructs them into a self-contained reasoning prefix:

$$z = \mathcal{H}^{-1}(\mathcal{T}). \tag{2}$$

Here, $\mathcal{H}^{-1}$ denotes a reconstruction operator rather than a strict mathematical inverse. We implement it with an LLM editor constrained to strip protocol-related content, resolve external references, and extract incremental reasoning. The editor reorganizes the current policy's reasoning rather than supplying an independently generated solution. Policy completion and correctness filtering then yield standalone training trajectories.

2.2 INVERSEHARNESS RECONSTRUCTION

We process harness segments sequentially through three constrained editing operations, accumulating a reasoning prefix z_n with z_0 empty.

Instruction stripping. Harness segments may begin or end with role acknowledgments, summaries, or formatted reports. For segment s_n, the editor identifies the first sentence of the reasoning to retain and the first sentence of any trailing protocol-related content. These boundaries define

$$p_n = \mathrm{Strip}(s_n) = (s_n)_{a(s_n):b(s_n)-1}. \tag{3}$$

Each boundary is applied only if its sentence matches the original output exactly and uniquely; otherwise, we use $a(s_n) = 1$ or $b(s_n) = |s_n| + 1$, respectively. If the boundaries yield an empty or invalid span, we leave the segment unchanged. Thus, stripping does not rewrite retained reasoning.

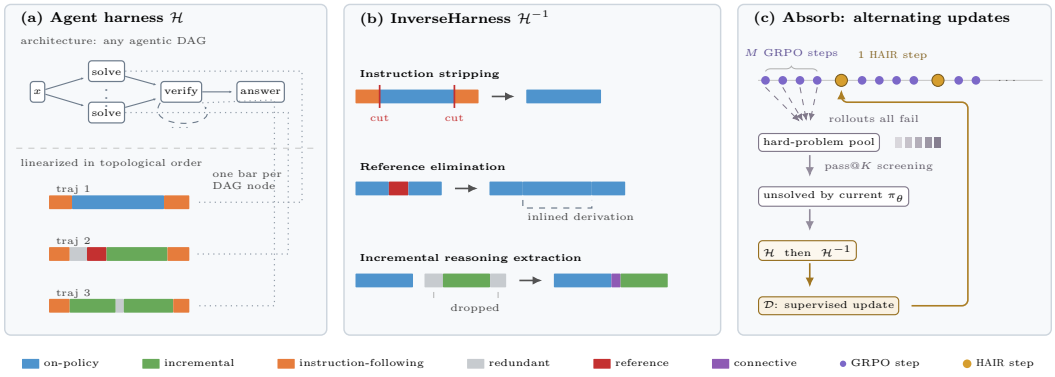


Figure 2: **The lift–project–absorb cycle in HAIR.** Each bar represents reasoning segmented by content origin. (a) *Lift.* The harness orchestrates solving and repair–verification rounds with the current policy, supplying verified intermediate results $L_{1:i-1}$ to later rounds. (b) *Project.* InverseHarness strips protocol-related content, resolves references to external results using the reasoning that established them, and retains incremental reasoning while removing repeated derivations. (c) *Complete and absorb.* The current policy continues the reconstructed prefix and produces a final answer. Correct responses are used in auxiliary supervised updates interleaved with standard GRPO.

Reference elimination. Later rounds may refer to harness-maintained results whose supporting reasoning is absent from the segment. We resolve these references using the accumulated prefix:

$$u_n = \mathrm{Deref}(p_n \mid z_{n-1}, c_n) = p_n\big[\ell \to \mathrm{content}(\ell; z_{n-1}),\ \forall \ell \in \mathcal{R}(p_n)\big], \tag{4}$$

where $\mathcal{R}(p_n)$ denotes harness-specific references and $\mathrm{content}(\ell; z_{n-1})$ the reasoning that establishes the referenced result. Role-specific prompts selected by c_n guide the editor toward common reference types; segments without such references remain unchanged. After editing, we reject reconstructions containing residual references matched by predefined role-specific filter strings.

Incremental reasoning extraction. We append reasoning that contributes beyond the prefix:

$$z_n = z_{n-1} \oplus \mathrm{Inc}(u_n \mid z_{n-1}, m_n), \tag{5}$$

where $\oplus$ denotes concatenation with a short transition when needed, and $z = z_N$. The editor removes repeated derivations while retaining complementary progress, such as a new numerical check from a verification round. Intermediate outputs in m_n help identify the reasoning that establishes those outputs, rather than passages that merely report them, so later references can be resolved to their supporting reasoning. Reference elimination and incremental extraction share the context z_{n-1} and are implemented in a single editing call.

Policy completion and acceptance. Because z combines segments generated under different harness contexts and subsequently edited, it is *policy-derived*, rather than an exact native-policy sample. We provide it to the current policy as a prefill with the reasoning block open and sample $y \sim \pi_\theta(\cdot \mid x, z)$. The completed response $a = z \| y$ is accepted using the same correctness criterion as RL judger. Failed attempts are retried up to a fixed continuation budget, and queries with no successful completion are discarded.

2.3 GRPO WITH PERIODIC INTERNALIZATION UPDATES

We alternate M standard GRPO iterations with one HAIR internalization update, leaving the GRPO objective unchanged.

Hard-problem selection. During GRPO, we accumulate queries with no correct response among their grouped native rollouts. Before each internalization update, we screen these candidates with K additional native responses from the current policy and retain only those with no correct response. This two-stage selection targets difficulty relative to the current policy and sampling budget.

Harness-assisted trajectory generation. For each retained query, we apply the harness to the current policy and reconstruct successful executions:

$$z = \mathcal{H}^{-1}\big(\mathcal{H}(x; \pi_\theta)\big). \tag{6}$$

Policy completion and correctness filtering (Section 2.2) yield the accepted training set $\mathcal{D}_{\mathrm{HAIR}} = \{(x, a)\}$.

Internalization update. We train on the accepted trajectories using the autoregressive supervised objective

$$\mathcal{L}_{\mathrm{IH}}(\theta) = -\frac{1}{|\mathcal{D}_{\mathrm{IH}}|} \sum_{(x,a)\in\mathcal{D}_{\mathrm{IH}}} \sum_{t} \log \pi_\theta(a_t \mid x, a_{<t}). \tag{7}$$

This update trains the policy to reproduce harness-derived reasoning as a standalone response. Each cycle screens candidates again and generates fresh trajectories from the updated policy, closing the loop between harness-assisted reasoning and standalone policy improvement.

3 Experiments

3.1 Experimental Setup

Implementation. We use Intern-S2-Preview 35B and 397B (Bai et al., 2026) as backbones, alternating 40 standard GRPO (Shao et al., 2024) iterations with one HAIR internalization update and setting the native screening budget to $K = 32$. Our harness, Intern-S1-MO (Gu et al., 2025), uses up to 10 solving rounds followed by up to 24 repair–verification rounds, with 3 verification votes per lemma and 5 per answer. The editor is the frozen initial backbone at the corresponding model scale. Further training and harness details are provided in the appendix.

Datasets. We train on an internal corpus of 29K competition-, undergraduate-, and research-level mathematics problems, approximately two thirds of which are proof-style. Evaluation covers four proof-style benchmarks: IMO-Proof-Bench (Luong et al., 2025), the UGD and Doctoral QE subsets of AdvancedMathBench (Kong et al., 2026), and PhD-Paper, which we plan to release; and three answer-based benchmarks: AMO-Bench (An et al., 2025), IMO-AnswerBench (Luong et al., 2025), and APEX-2025 (Dekoninck et al., 2026). All tasks use binary correctness judgments. We report pass@1 and pass@4 (Chen et al., 2021), with evaluation protocols and judge details in the appendix.

Baselines. Our primary comparison is with GRPO trained using the same backbone, corpus, and total number of optimizer steps. For context, we also report proprietary models: Gemini 3.1 Pro (Google, 2026), GPT-5.5/5.6 Sol (OpenAI, 2026a;b), and Claude Opus 4.8/5 (Anthropic, 2026a;b); and open-source models: gpt-oss 120B (OpenAI, 2025), GLM-5.2 (Z.ai, 2026), DeepSeek-V4-Pro (DeepSeek-AI, 2026), and Qwen3.8-Max (Qwen Team, 2026). All models use a 128K context-length limit, with reasoning effort set to `high` for GPT models and `max` for DeepSeek.

3.2 Overall Results

HAIR improves pass@1 over matched GRPO on all four proof-style benchmarks at both scales (Table 1). At 397B, scores increase from 55.5 to 68.8 on IMO-Proof-Bench (+24.0% relative) and from 29.2 to 38.4 on PhD-Paper (+31.5%), compared with 67.8 to 75.2 on UGD. At 35B, relative gains reach 17.3% on Doctoral QE and 17.4% on PhD-Paper, versus 1.2% on UGD. The larger relative gains on several harder benchmarks are consistent with targeting problems where native sampling provides limited positive supervision.

Relative to the initial 397B backbone, the full training procedure raises pass@1 from 29.0 to 68.8 on IMO-Proof-Bench and from 16.1 to 38.4 on PhD-Paper. Our 397B model outperforms most reported open-source baselines on proof-style evaluation, including DeepSeek-V4-Pro (30.0 on IMO-Proof-Bench). It also surpasses Gemini 3.1 Pro and Claude Opus 4.8 across these benchmarks, while remaining behind Qwen3.8-Max and GPT-5.6 Sol. These comparisons contextualize overall performance; the matched GRPO comparison isolates the benefit of the training framework under the controlled settings.

Gains also extend to answer-based benchmarks (Table 2). At 397B, pass@1 rises from 69.0 to 76.0 on AMO-Bench and from 91.0 to 93.0 on IMO-AnswerBench; at 35B, APEX-2025 improves from 34.0

Table 1: **Main results across four benchmarks.** All values are percentages. The best result per column is marked within each block separately: underlined among the reference models, and **bold** within each of our model scales. Marks are not comparable across blocks.

Model	IMO-Proof		UGD		QE		PhD-Paper	
	pass@1	pass@4	pass@1	pass@4	pass@1	pass@4	pass@1	pass@4
Gemini 3.1 Pro	46.7	60.0	52.7	74.5	35.3	57.8	23.8	38.5
Opus 4.8	35.6	50.0	47.4	64.0	25.0	42.2	9.4	15.4
Opus 5	67.5	83.3	74.8	89.0	51.9	70.5	40.1	55.8
GPT 5.5	66.7	88.3	66.5	82.5	53.3	82.2	50.7	66.3
GPT 5.6	77.1	90.8	73.0	86.5	68.3	90.0	61.5	82.7
GPT OSS	16.0	31.5	27.8	49.5	3.6	12.2	8.8	14.7
GLM 5.2	27.1	45.8	46.0	65.0	25.5	53.3	13.6	22.1
DeepSeek V4 Pro	30.0	39.2	53.5	75.5	37.2	58.9	19.2	28.8
Qwen 3.8 max	70.2	84.2	75.5	87.0	75.0	90.0	40.6	52.9
35B Baseline	19.0	30.8	45.5	65.5	23.3	47.8	13.2	20.2
GRPO	38.6	48.3	68.8	83.0	46.9	68.9	20.7	29.8
HAIR	**42.5**	**53.3**	**69.6**	**85.0**	**55.0**	**76.7**	**24.3**	**36.5**
397B Baseline	29.0	45.0	51.9	70.9	37.7	63.3	16.1	22.1
GRPO	55.5	63.7	67.8	82.7	58.6	**77.6**	29.2	41.3
HAIR	**68.8**	**78.4**	**75.2**	**85.7**	**62.0**	77.4	**38.4**	**52.2**

Table 2: **Results across AMO-Bench, AnswerBench, and APEX-2025.**

Model	AMO-Bench		AnswerBench	APEX-2025	
	pass@1	pass@4	pass@1	pass@1	pass@4
Gemini 3.1 Pro	69.0	80.0	89.0	59.0	92.0
Opus 5	72.0	80.0	94.0	96.0	100.0
GPT 5.6	74.0	81.0	96.0	91.0	100.0
GPT OSS	48.0	69.0	84.0	31.0	57.0
GLM 5.2	52.0	76.0	77.0	37.0	67.0
DeepSeek V4 Pro	65.0	78.0	87.0	31.0	60.0
Qwen 3.8 max	79.0	87.0	96.0	87.0	100.0
35B Baseline	58.0	70.0	87.0	33.0	58.0
GRPO	61.0	75.0	**91.0**	34.0	57.0
HAIR	**65.0**	**79.0**	**91.0**	**46.0**	**66.0**
397B Baseline	65.0	77.0	90.0	39.0	54.0
GRPO	69.0	83.0	91.0	**49.0**	**80.0**
HAIR	**76.0**	**85.0**	**93.0**	**49.0**	79.0

to 46.0. Improvements are not uniform: at 397B, APEX-2025 pass@1 remains 49.0, while pass@4 decreases slightly on APEX-2025 (80.0 → 79.0) and Doctoral QE (77.6 → 77.4). Qwen3.8-Max also remains ahead on AMO-Bench (79.0 vs. 76.0) and IMO-AnswerBench (96.0 vs. 93.0).

3.3 Iterative Self-Improvement across Training Stages

Figure 1b compares the standalone policy, the same policy with the harness, and matched GRPO across training stages, averaged over the four proof-style benchmarks. Per-benchmark results appear in Appendix D, Figure 5.

Standalone performance increases from 25.2 at initialization to 40.7, 41.7, 46.1, and 47.9 across four stages. Although the first stage yields the largest gain, later stages add 7.2 points, leaving a 4.1-point advantage over GRPO at stage 4.

Meanwhile, harnessed performance rises from 51.2 to 72.2, showing that the evolving policy also strengthens the source of harness-assisted trajectories. The harness retains a substantial advantage over standalone reasoning: 26.0 points initially and 24.3 at stage 4. By stage 4, standalone performance

Table 3: **S1MO harness versus Pass@K under comparable inference budgets.** One S1MO run costs $\approx$ 727k generated tokens per query, comparable to the budget of Pass@16 on the same policy.

Model	IMO-Proof		PhD-Paper	
	Pass@16	S1MO	Pass@16	S1MO
35B Baseline	36.7	**55.0**	23.1	**26.9**
HAIR	55.0	**73.3**	46.2	**48.1**

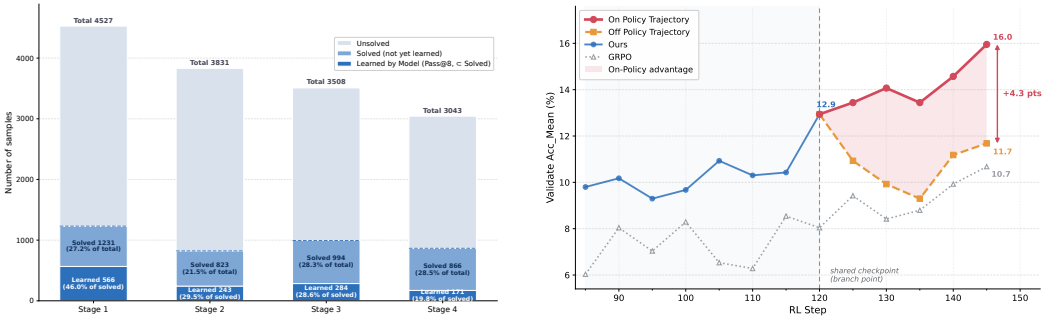


Figure 3: **Harness-assisted trajectory generation and learning.** (a) Each bar shows selected queries with no successful native rollout under the screening budget. Nested segments indicate harness-solved queries and those subsequently solved by the standalone policy under pass@8. (b) Training branches from the same checkpoint at RL step 120, using matched queries, trajectory counts, and hyperparameters. Trajectories come from either the current policy or another backbone with comparable evaluation performance.

(47.9) approaches the initial harnessed score (51.2), surpassing it on UGD but not on the hardest benchmarks. Together, these trends support continued improvement within the closed loop, while showing that a substantial inference-time benefit from orchestration remains.

3.4 Harness versus Best-of-N as a Policy-Improvement Operator

We compare harness-assisted solving with native pass@16 from the same policy. On IMO-Proof-Bench, the harness generates approximately 727K assistant-side tokens per query, comparable to 16 native responses. This approximately matches generated-token budgets on this benchmark.

The harness achieves higher success rates in every reported comparison (Table 3). On IMO-Proof-Bench, it outperforms native pass@16 both before training (55.0 vs. 36.7) and after training (73.3 vs. 55.0), maintaining an 18.3-point margin. The advantage on PhD-Paper is smaller, decreasing from 3.8 to 1.9 points after training. Notably, trained native pass@16 matches the initial harnessed score on IMO-Proof-Bench (55.0).

These results establish the harness as a stronger source of successful executions in the evaluated settings. Its composition with InverseHarness, policy completion, and correctness filtering converts these executions into standalone training trajectories.

3.5 Harness Success and Subsequent Standalone Learning

We track hard queries with no correct native response in either the grouped GRPO rollouts or the additional pass@32 screening. The harness solves 21.5%–28.5% of these queries per stage, yielding roughly 0.8K–1.2K solved queries. Of these, 19.8%–46.0% are subsequently solved by the standalone policy under pass@8 (Figure 3a). Here, "unsolved" and "learned" refer to observed success under the respective sampling budgets.

From stage 1 to stage 4, the harness-solved fraction increases from 27.2% to 28.5%, while attempted queries decrease from 4527 to 3043. Over the same stages, the fraction of harness-solved queries subsequently solved by the standalone policy falls from 46.0% to 19.8%, revealing that internalization stage may form the bottleneck of HAIR.

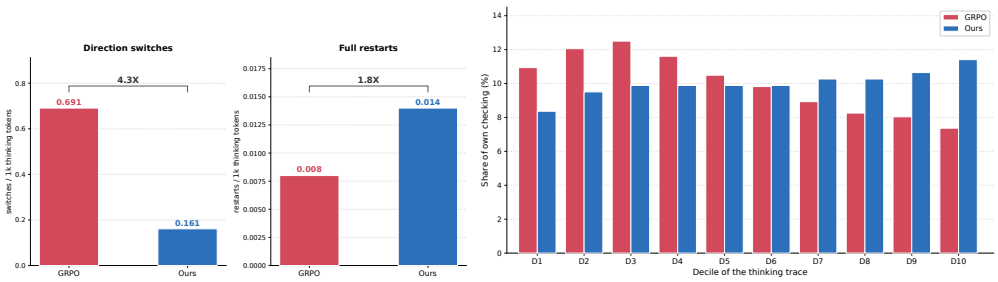


Figure 4: **Reasoning behavior after HAIR and GRPO training.** Left: direction-switch and full-restart rates per 1K thinking tokens. Right: self-verification content as a share of each reasoning decile. HAIR exhibits fewer direction switches, more full restarts, and greater emphasis on verification later.

3.6 Effect of Trajectory Provenance

Reconstructed trajectories are *policy-derived*, rather than exact native-policy samples (Section 2.2). We test whether their origin in the current policy matters beyond response correctness.

Two training branches start from the same checkpoint at RL step 120, using the same queries, number of accepted trajectories, and training hyperparameters. One uses harness-assisted trajectories from the current policy; the other uses a backbone from another training run with comparable evaluation performance. Both conditions provide correct training responses.

Current-policy trajectories improve validation accuracy from 12.9 to 16.0 (Figure 3b). The other-backbone condition initially falls to 9.3 and recovers to 11.7, compared with 10.7 for GRPO without internalization updates. The final 4.3-point gap shows that correctness alone is insufficient to explain training utility. This supports the view that generating reasoning from the evolving policy and preserving it during reconstruction are important when internalization is interleaved with RL stages.

3.7 Analysis of Reasoning Behavior

We analyze rollout trajectories across the evaluation distribution using single-blind annotation by Claude Code with Fable 5. Direction-switch and full-restart counts are normalized per 1K thinking tokens; self-verification is measured as the content share within each reasoning decile (Figure 4).

Strategy switching and restarts. Compared with GRPO, HAIR produces fewer direction switches (0.161 vs. 0.691) but more full restarts (0.014 vs. 0.008). Qualitative inspection suggests that some GRPO switches are local reformulations or parameterization changes, whereas HAIR restarts involve broader changes in approach. This pattern is consistent with the harness's multi-round solving procedure, which enables fresh attempts informed by earlier progress.

Self-verification. GRPO's verification share peaks at 12.5% in D3 and falls to 7.3% in D10. By contrast, HAIR's share rises from 8.4% in D1 to 11.4% in D10, with the curves crossing near the midpoint. This later emphasis on verification is consistent with the harness's checks of developed solutions and final answers.

These observations provide behavioral evidence consistent with internalizing harness-associated reasoning patterns, but do not establish their causal contribution to accuracy.

3.8 Ablation Studies

We ablate reconstruction and harness complexity while holding the backbone, training corpus, optimizer settings, and total optimizer steps fixed (Table 4).

Removing InverseHarness. Training on raw multi-turn transcripts for the same harness-solved problems retains orchestration instructions and references to external intermediate state. This variant scores 43.0 on average, below GRPO's 43.7 and approximately 4.8 points below full HAIR. Successful transcripts alone therefore do not reproduce the benefit of reconstruction into standalone trajectories.

Table 4: **Ablation of the two components of the operator.** pass@1 is reported.

Setting	IMO-Proof	UGD	QE	PhD-Paper	Avg.
35B Baseline	19.0	45.5	23.3	13.2	25.3
GRPO	38.6	68.8	46.9	20.7	43.7
w/o InverseHarness	38.8	66.7	48.6	18.0	43.0
w/ simplified harness	40.8	**69.8**	47.8	21.6	45.0
HAIR	**42.5**	69.6	**55.0**	**24.3**	**47.9**

Weakening the harness. Replacing Intern-S1-MO with a simplified solve–verify loop removes restarts with inherited progress, per-lemma verification, and iterative repair, while retaining reconstruction and internalization. This variant scores 45.0 on average and improves over GRPO on every benchmark, but remains approximately 2.9 points below the full method.

Together, these ablations support complementary roles for reconstruction and richer orchestration, especially on harder problems.

4 Related Work

Exploration boundaries in RLVR. RLVR is a standard approach for improving LLM reasoning (Guo et al., 2025; Shao et al., 2024; Team, 2025; Lyu et al., 2025), but on-policy sampling can limit exploration when successful trajectories are rare. RLVR often improves pass@1 more than large-k coverage (Yue et al., 2025), with updates concentrated on a small set of high-entropy decisions (Wang et al., 2025a); in zero-reward regimes, even denser rewards or better credit assignment may fail without changing the training distribution (Prakash & Buvanesh, 2025). Prolonged, stabilized RL can nevertheless expand beyond the base model's observed capabilities (Liu et al., 2025b), suggesting a difficult rather than absolute exploration boundary.

Expanding the reachable reasoning space. Existing methods address the exploration bottleneck by *preserving support* via entropy regularization (Cui et al., 2025b; Jiang et al., 2025), modified clipping or KL objectives (Yu et al., 2025; Huang et al., 2025), and exploration bonuses (Dai et al., 2025); *moving the frontier* via adaptive problem selection or reweighting (Shi et al., 2025; Chen et al., 2025b; Wang et al., 2025b; Parashar et al., 2025; Zhao et al., 2024; Zhang et al., 2025) and intermediate guidance (Li et al., 2026); or *expanding support* via off-policy demonstrations (Yan et al., 2025) and explicit search (Guan et al., 2025). Complementary approaches exploit partially correct trajectories through process supervision (Cui et al., 2025a; Wang et al., 2023) or co-evolve tasks and policies through self-play and automatic task generation (Zhao et al., 2025a; Liu et al., 2025a).

Math harnesses. A complementary line of work augments mathematical reasoning with external inference-time harnesses. These include executable tools and symbolic solvers (Gao et al., 2023; Chen et al., 2023; Gou et al., 2024), iterative generation–verification–refinement pipelines and long-horizon agents (Huang & Yang, 2025; Feng et al., 2026; Gu et al., 2025), proof-assistant interaction for formal mathematics (Hubert et al., 2026; Chen et al., 2025a; Baba et al., 2025; Kung et al., 2026), and neuro-symbolic systems for geometry (Trinh et al., 2024; Chervonyi et al., 2025; Zhao et al., 2025b). Together, these works show that external orchestration can substantially extend the effective reasoning capability of a fixed model.

5 Conclusion

We introduced HAIR, a framework that internalizes harness-induced reasoning improvements through a lift–project–absorb cycle. Its key component, InverseHarness, reconstructs successful harness executions into standalone trajectories while preserving the evolving policy's reasoning. Experiments at 35B and 397B show gains over matched GRPO and support a closed loop of self-improvement, with trajectory provenance contributing beyond correctness alone. These results suggest that agentic harnesses can serve as policy-improvement operators for training, extending self-improvement beyond trajectories readily discovered by native sampling.

AI USE STATEMENT

In this work, we used generative AI tools solely for language polishing. The scientific ideas, methodology, analyses, and conclusions were entirely developed by the authors, while the generative AI tools assisted only in improving clarity and readability of the text. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Shengnan An, Xunliang Cai, Xuezhi Cao, Xiaoyu Li, Yehao Lin, Junlin Liu, Xinxuan Lv, Dan Ma, Xuanlin Wang, Ziwen Wang, and Shuang Zhou. AMO-Bench: Large language models still struggle in high school math competitions. *arXiv preprint arXiv:2510.26768*, 2025. URL https://arxiv.org/abs/2510.26768.

Thomas Anthony, Zheng Tian, and David Barber. Thinking fast and slow with deep learning and tree search. *Advances in neural information processing systems*, 30, 2017.

Anthropic. Introducing Claude Opus 4.8, 2026a. URL https://www.anthropic.com/news/claude-opus-4-8.

Anthropic. Introducing Claude Opus 5, 2026b. URL https://www.anthropic.com/news/claude-opus-5.

Kaito Baba, Chaoran Liu, Shuhei Kurita, and Akiyoshi Sannai. Prover agent: An agent-based framework for formal mathematical proofs. *arXiv preprint arXiv:2506.19923*, 2025.

Lei Bai, Jiaqi Cao, Chiyu Chen, et al. Intern-S2-Preview: Scientific agentic foundation model. *arXiv preprint arXiv:2608.13505*, 2026. URL https://arxiv.org/abs/2608.13505.

Luoxin Chen, Jinming Gu, Liankai Huang, Wenhao Huang, Zhicheng Jiang, Allan Jie, Xiaoran Jin, Xing Jin, Chenggang Li, Kaijing Ma, et al. Seed-prover: Deep and broad reasoning for automated theorem proving. *arXiv preprint arXiv:2507.23726*, 2025a.

Mark Chen, Jerry Tworek, Heewoo Jun, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. URL https://arxiv.org/abs/2107.03374.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research*, 2023.

Xiaoyin Chen, Jiarui Lu, Minsu Kim, Dinghuai Zhang, Jian Tang, Alexandre Piché, Nicolas Gontier, Yoshua Bengio, and Ehsan Kamalloo. Self-evolving curriculum for llm reasoning. *arXiv preprint arXiv:2505.14970*, 2025b.

Yuri Chervonyi, Trieu H Trinh, Miroslav Olšák, Xiaomeng Yang, Hoang Nguyen, Marcelo Menegali, Junehyuk Jung, Vikas Verma, Quoc V Le, and Thang Luong. Gold-medalist performance in solving olympiad geometry with alphageometry2. *arXiv preprint arXiv:2502.03544*, 2025.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Ming Yan, Yuxin Du, Yuchen Fan, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025a.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, and Ning Ding. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*, 2025b.

Runpeng Dai, Linfeng Song, Haolin Liu, Zhenwen Liang, Dian Yu, Haitao Mi, Zhaopeng Tu, Rui Liu, Tong Zheng, Hongtu Zhu, and Dong Yu. Cde: Curiosity-driven exploration for efficient reinforcement learning in large language models. *arXiv preprint arXiv:2509.09675*, 2025.

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026. URL https://arxiv.org/abs/2606.19348.

Jasper Dekoninck, Nikola Jovanović, Tim Gehrunger, Kári Rögnvaldsson, Ivo Petrov, Chenhao Sun, and Martin Vechev. Beyond benchmarks: MathArena as an evaluation platform for mathematics with LLMs. *arXiv preprint arXiv:2605.00674*, 2026. URL https://arxiv.org/abs/2605.00674.

Tony Feng, Trieu H Trinh, Garrett Bingham, Dawsen Hwang, Yuri Chervonyi, Junehyuk Jung, Joonkyung Lee, Carlo Pagano, et al. Towards autonomous mathematics research. *arXiv preprint arXiv:2602.10177*, 2026.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. Pal: Program-aided language models. *International Conference on Machine Learning*, 2023.

Google. Gemini 3.1 Pro: A smarter model for your most complex tasks, 2026. URL https://blog.google/innovation-and-ai/models-and-research/gemini-models/gemini-3-1-pro/.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Minlie Huang, Nan Duan, and Weizhu Chen. Tora: A tool-integrated reasoning agent for mathematical problem solving. *International Conference on Learning Representations*, 2024.

Yuzhe Gu, Songyang Gao, Zijian Wu, Lingkai Kong, Wenwei Zhang, Zhongrui Cai, et al. Intern-s1-mo: Long-horizon reasoning agent for olympiad-level mathematical problem solving. *arXiv preprint arXiv:2512.10739*, 2025.

Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and Mao Yang. rstar-math: Small llms can master math reasoning with self-evolved deep thinking. *arXiv preprint arXiv:2501.04519*, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Guanhua Huang, Tingqiang Xu, Mingze Wang, Qi Yi, Xue Gong, Siheng Li, Ruibin Xiong, Kejiao Li, Yuhao Jiang, and Bo Zhou. Low-probability tokens sustain exploration in reinforcement learning with verifiable reward. *arXiv preprint arXiv:2510.03222*, 2025.

Yichen Huang and Lin F Yang. Winning gold at imo 2025 with a model-agnostic verification-and-refinement pipeline. *arXiv preprint arXiv:2507.15855*, 2025.

Thomas Hubert, Rishi Mehta, Laurent Sartran, Miklós Z Horváth, Goran Žužić, Eric Wieser, Aja Huang, Julian Schrittwieser, et al. Olympiad-level formal mathematical reasoning with reinforcement learning. *Nature*, 651:607–613, 2026.

Yuxian Jiang, Yafu Li, Guanxu Chen, Dongrui Liu, Yu Cheng, and Jing Shao. Rethinking entropy regularization in large reasoning models. *arXiv preprint arXiv:2509.25133*, 2025.

Lingkai Kong, Zijian Wu, Yuzhe Gu, Haiteng Zhao, Wenyong Huang, Shuang Sun, Zhicheng Xiong, Xiaotian Zhang, Shuya Zhao, Yan Wang, Disheng Xu, Wenwei Zhang, and Kai Chen. Advanced-MathBench: A benchmark suite for advanced mathematical proof generation and verification. *arXiv preprint arXiv:2607.11849*, 2026. URL https://arxiv.org/abs/2607.11849.

Po-Nien Kung, Linfeng Song, Dawsen Hwang, Jinsung Yoon, Chun-Liang Li, Simone Severini, Mirek Olšák, Edward Lockhart, et al. Leap: Supercharging llms for formal mathematics with agentic frameworks. *arXiv preprint arXiv:2606.03303*, 2026.

Jiazheng Li, Hongzhou Lin, Hong Lu, Kaiyue Wen, Zaiwen Yang, Jiaxuan Gao, Yi Wu, and Jingzhao Zhang. Questa: Expanding reasoning capacity in llms via question augmentation. *International Conference on Learning Representations*, 2026.

Bo Liu, Leon Guertler, Simon Yu, Zichen Liu, Penghui Qi, Daniel Balcells, Mickel Liu, Cheston Tan, Weiyan Shi, Min Lin, et al. Spiral: Self-play on zero-sum games incentivizes reasoning via multi-agent multi-turn reinforcement learning. *arXiv preprint arXiv:2506.24119*, 2025a.

Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models. *arXiv preprint arXiv:2505.24864*, 2025b.

Thang Luong, Dawsen Hwang, Hoang H Nguyen, Golnaz Ghiasi, Yuri Chervonyi, Insuk Seo, Junsu Kim, Garrett Bingham, Jonathan Lee, Swaroop Mishra, Alex Zhai, Huiyi Hu, Henryk Michalewski, Jimin Kim, Jeonghyun Ahn, Junhwi Bae, Xingyou Song, Trieu Hoang Trinh, Quoc V Le, and Junehyuk Jung. Towards robust mathematical reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 35418–35442. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.1794. URL https://aclanthology.org/2025.emnlp-main.1794/.

Chengqi Lyu, Songyang Gao, Yuzhe Gu, Wenwei Zhang, Jianfei Gao, Kuikun Liu, Ziyi Wang, Shuaibin Li, Qian Zhao, and Haian Huang. Exploring the limit of outcome reward for learning mathematical reasoning. 2025.

OpenAI. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025. URL https://arxiv.org/abs/2508.10925.

OpenAI. Introducing GPT-5.5, 2026a. URL https://openai.com/index/introducing-gpt-5-5/.

OpenAI. The builder's guide to GPT-5.6, 2026b. URL https://openai.com/index/builders-guide-to-gpt-5-6/.

Shubham Parashar, Shurui Gui, Xiner Li, Hongyi Ling, Sushil Vemuri, Blake Olson, Eric Li, Yu Zhang, James Caverlee, Dileep Kalathil, et al. Curriculum reinforcement learning from easy to hard tasks improves llm reasoning. *arXiv preprint arXiv:2506.06632*, 2025.

Jatin Prakash and Anirudh Buvanesh. What can you do when you have zero rewards during rl? *arXiv preprint arXiv:2510.03971*, 2025.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork, 2026. URL https://qwen.ai/blog?id=qwen3.8.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, and Y. Wu. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. 2024.

Maohao Shen, Guangtao Zeng, Zhenting Qi, Zhang-Wei Hong, Zhenfang Chen, Wei Lu, Gregory Wornell, Subhro Das, David Cox, and Chuang Gan. Satori: Reinforcement learning with chain-of-action-thought enhances llm reasoning via autoregressive search. *arXiv preprint arXiv:2502.02508*, 2025.

Taiwei Shi, Yiyang Wu, Linxin Song, Tianyi Zhou, and Jieyu Zhao. Efficient reinforcement finetuning via adaptive curriculum learning. *arXiv preprint arXiv:2504.05520*, 2025.

Avi Singh, John D Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J Liu, James Harrison, Jaehoon Lee, Kelvin Xu, et al. Beyond human data: Scaling self-training for problem-solving with language models. *Transactions on Machine Learning Research*, 2024.

Akshit Sinha, Arvindh Arun, Shashwat Goel, Steffen Staab, and Jonas Geiping. The illusion of diminishing returns: Measuring long horizon execution in llms. *arXiv preprint arXiv:2509.09677*, 2025.

Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, 2025.

Trieu H Trinh, Yuhuai Wu, Quoc V Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482, 2024.

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. *arXiv preprint arXiv:2312.08935*, 2023.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*, 2025a.

Zhenting Wang, Guofeng Cui, Yu-Jhe Li, Kun Wan, and Wentian Zhao. Dump: Automated distribution-level curriculum learning for rl-based llm post-training. *arXiv preprint arXiv:2504.09710*, 2025b.

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. Learning to reason under off-policy guidance. *arXiv preprint arXiv:2504.14945*, 2025.

Ping Yu, Jing Xu, Jason Weston, and Ilia Kulikov. Distilling system 2 into system 1. *arXiv preprint arXiv:2407.06023*, 2024.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025.

Z.ai. GLM-5.2: Built for long-horizon tasks, 2026. URL `https://z.ai/blog/glm-5.2`.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D Goodman. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35, 2022.

Xuemiao Zhang, Liangyu Xu, Feiyu Duan, Yongwei Zhou, Sirui Wang, Rongxiang Weng, Jingang Wang, and Xunliang Cai. Preference curriculum: Llms should always be pretrained on their preferred data. *arXiv preprint arXiv:2501.13126*, 2025.

Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Yang Yue, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute zero: Reinforced self-play reasoning with zero data. *arXiv preprint arXiv:2505.03335*, 2025a.

Haiteng Zhao, Junhao Shen, Yiming Zhang, Songyang Gao, Kuikun Liu, Tianyou Ma, Fan Zheng, Dahua Lin, Wenwei Zhang, and Kai Chen. Achieving olympiad-level geometry large language model agent via complexity boosting reinforcement learning. *arXiv preprint arXiv:2512.10534*, 2025b.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

Zirui Zhao, Hanze Dong, Amrita Saha, Caiming Xiong, and Doyen Sahoo. Automatic curriculum expert iteration for reliable llm reasoning. *arXiv preprint arXiv:2410.07627*, 2024.

A Editor Details

A.1 Synthesized Trajectory Examples

This appendix illustrates the output of $\mathcal{H}^{-1}$ on a real training query. The example is taken verbatim from a training run; only length truncation is applied, marked by `[...]`.

Query. A graph-theory proof problem from the undergraduate-level portion of the training corpus:

> Let G be a 2-connected graph with $\delta(G) \geqslant 3$. If G is not a clique, then there exist vertices $u, v, w \in V(G)$ with $uv, vw \in E(G)$ and $uw \notin E(G)$, such that the graph $G - u - w$ is connected. Prove the above statement.

For this query no native rollout succeeds under the screening budget (Section B.1), so it is selected for harness-assisted generation. The harness transcript spans several solving rounds plus a repair–verification loop, and the reconstructed trajectory that $\mathcal{H}^{-1}$ returns is *a single continuous stream* of roughly 134k characters. Three excerpts are shown below.

Opening of the fold. The first unit enters the accumulator unchanged, and its head is a genuine reading of the problem rather than a restatement of a harness instruction:

> The user wants a rigorous mathematical proof for a graph theory problem. The problem asks to prove that in a 2-connected graph G with minimum degree $\delta(G) \geq 3$, if G is not a clique, then there exist vertices $u, v, w \in V(G)$ such that $uv \in E(G)$, $vw \in E(G)$, $uw \notin E(G)$, and the graph $G - \{u, w\}$ is connected.
>
> Let's break down the problem statement. Graph G is 2-connected. This means G is connected and remains connected after removal of any single vertex. Equivalently, $|V(G)| \geq 3$ and there are no cut-vertices. Since $\delta(G) \geq 3$, $|V(G)|$ must be at least 4 [...]

Mid-trajectory, contributed by a later round. The text refers to concrete objects and arguments, never to a lemma by name—the effect of reference elimination (Section 2.2):

> [...] using the 2-connectivity degree argument (as sketched above). Since $a \in A$, $b \in B$ are removed, hubs u, w remain. $A \setminus \{a\}$ connects to u or w (forced by the 2-connectivity degree surplus and connectivity of $G \setminus \{a\}$), and $B \setminus \{b\}$ connects to u or w similarly. Then u, w are connected via a path in $A \cup \{u, w\}$; if $a = v = u_{\text{med}}$ we lose that path, so use the second disjoint u–w path guaranteed by Menger [...] Show every $x \in V \setminus \{a, b\}$ has a path to u avoiding a, b. If $x = u$, trivial. If $x = w$, we need a path w [...]

Late trajectory, verification-style reasoning. The tail of the synthesized thought is dominated by content originating in the harness's repair and verification stages, which is why self-verification density *rises* toward the end of HAIR traces (§3.7):

> [...] the internal vertices of P_1 are just $\{v_0\}$, so P_2 avoids v_0. Does P_2 avoid b? Maybe b is on P_2. If b is not on P_2, then $P_2 \subset H$, so u_0, w_0 are connected in H. Done. If b is on P_2, we need *another* path? 2-connectivity guarantees 2 disjoint paths. That gives us exhaustion (case analysis on b falling on P_1 or P_2). Removing the 2 vertices v_0, b might break both paths if $v_0 \in P_1$ and $b \in P_2$ [...]

Two properties visible in these excerpts are worth stating. First, there is no trace of the harness protocol: no round markers, no discussion of output format, and no "Lemma 3"-style citation of an earlier round's result. Second, the register is uniform throughout—fold boundaries are not detectable from a change in voice, because every span is a verbatim sample from the same policy and only short connectives were inserted between them.

A.2 Editor prompt

The incremental reasoning extraction of Section 2.2 is carried out by the LLM editor under the instruction reproduced verbatim below. It is applied at each fold step, with the accumulated prefix supplied as `The Main Thought Process` and the newly cleaned segment as `Additional Thought Process`. The repeated prohibition on summarizing and rewriting is what keeps the reconstructed trajectory close to the policy's own distribution, and the `No Additional Thought` flag makes the incremental criterion explicit, so a round that merely restates earlier progress contributes nothing rather than being appended as redundant text.

```
## Instruction

**Task:** You will be provided with a "Main Thought Process" and multiple "Additional Thought Processes"
that attempt to solve a Question. Your goal is to select content from Additional Thought Processes that are
 missing from the Main Thought Process as an extension that can be appended to the end.

**Input Components:**
1.  **The Question:** The primary problem being solved.
2.  **Additional Thought Process:** A collection of distinct streams of consciousness (trials, proofs,
reasoning steps) from different sources or attempts, denoted by #### Additional Thought Process 1 ####,
#### Additional Thought Process 2 ####, etc.
3.  **The Main Thought Process:** The main thought process on which the continued writing will be done.
```

```
**Task Guidelines:**

1.  **Select the Incremental Content:**
*  Identify the parts in Additional Thought Process that offer new information, corrections, or alternative
 perspectives not covered in The Main Thought Process. Note that you should not summarize the viewpoints of
 these parts; instead, extract the involved thought context completely, including the detailed analysis,
calculations, self-debates, and trial-and-error processes.

2.  **Direct Copy the Content:**
*  You must not summarize, rewrite, or draw conclusions from the selected content. Just copy these contents
 as the output, and can separate them by normal line breakers as in the provided thoughts. Your only
allowed modification is to use transitional phrases.
*  In the selected content, the stream-of-consciousness thought contains much self-validation and self-
challenging content. These contents should be kept completely, as they are necessary thoughts rather than
dead ends.
*  In the selected content, keep the streaming thought and guess before construction and trial. They are
necessary trials before reaching the correct construction and guess. Keep these contents entirely.

3.  **Use Transitional Phrases:**
*  Concatenate these new parts using natural connecting sentences to maintain a stream-of-consciousness
flow (e.g., "Wait, looking at this again...", "Alternatively, we should consider...", "Let's double-check
the calculation for...").
*  You must use diverse Transitional Phrases, not the given examples above.
*  Specifically, at the beginning of your output, which can also be considered as exactly the end of The
Main Thought Process, you must use appropriate transitional phrases to ensure the output can be directly
concatenated to The Main Thought Process.
*  The end of "The Main Thought Process" may have already pre-appended other extended thoughts. You must
not imitate the end of "The Main Thought Process", and increase the diversity of transitional phrases used
in your output compared to the end of "The Main Thought Process".

4.  **Trivial Case:**
*  If there is no incremental information provided in the Additional Thought Process, such as all the
conclusions are the same as The Main Thought Process, directly output **No Additional Thought** as a flag.

**Output Guidelines:**

1.  **Maintain Original Voice and Detail:** Keep any content that is selected exactly as it appears in "the
 Additional Thought Processes". Do not summarize the reasoning or rewrite proofs at the word level. Simply
copy the relevant sections verbatim.
2.  **Do Not Include Main Thought Process:** Your output should consist only of the new transitional
phrases and the copied text from the Additional Thought Processes. Do not output the original Main Thought
Process.
3.  **Ignore the Discussion about Output Requirement in the Thought:** There may be some instructions-
following thought, such as "I need to output like xxx format" or "Now we need to output the final
conclusion". Totally ignore these instructions, and focus on processing the thought text.
4.  **Output Thought Processes, Not Conclusions:** This task requires extracting additional thought
processes, not summarizing their conclusions. Therefore, summarizing verification conclusions based on the
additional thoughts is strictly prohibited. Output only the thought processes.
5.  **Avoid Lemma xx References:** Avoid any references to lemmas xx. You must replace any content that
references a lemma xx with the original content corresponding to that lemma. Absolutely avoid any explicit
"lemma xx" wording.


## Input Format

### The Question ###

{The Question}

### Additional Thought Process ###

#### Additional Thought Process 1 ####
{Additional Thought Process 1}

#### Additional Thought Process 2 ####
{Additional Thought Process 2}

...

### The Main Thought Process ###
{The Main Thought Process}

---

## Desired Output:

### Additional Thought ###

{Additional Thought}
```

Role-conditioned reference elimination

Guideline 5 of the prompt above,

```
5.  **Avoid Lemma xx References:** Avoid any references to lemmas xx.
You must replace any content that references a lemma xx with the original
content corresponding to that lemma.  Absolutely avoid any explicit
"lemma xx" wording.
```

is the reference-elimination instruction of Section 2.2, and it is *not* a fixed part of the prompt. Following the role c_n of the round being folded, we substitute the reference types common to that role. The clause shown here is the variant used for solving and verification rounds, whose harness state consists of named lemmas; rounds whose intermediate results are of a different kind receive the corresponding instruction instead, and the answer-verification variant carries no lemma clause at all. This is what makes reference elimination targeted rather than a single generic instruction applied uniformly to every round.

B GRPO CONFIGURATION

GRPO steps and HAIR updates share one training loop, one optimizer, and one set of model parallelism settings; only the data source and the loss differ. Table 5 lists the settings read from the training configuration.

Table 5: GRPO and optimizer configuration. Both scales share these settings.

Setting	**Value**
Optimizer	AdamW, $\beta = (0.9, 0.95)$, weight decay 0.1, $\epsilon = 10^{-15}$
Learning rate	10^{-6}, constant, no warmup
Gradient clipping	max-norm 1.0; steps with grad-norm > 1.6 are skipped
Clipping range	low $= 0.2$, high $= 0.28$ (asymmetric)
Dual clipping	$c = 2.0$
KL penalty	**disabled** (`use_kl_loss=False`, coefficient 0)
Reference model	none retained
Rollout importance sampling	token-level, bidirectional correction
Rollouts per prompt	8
RL global batch size	512 prompts
Rollout temperature	1.0 (top-p 1.0, top-k disabled)
Sequence budget	8K prompt / 56K response (RL)
Packing length	64K

No KL penalty. We train without a KL penalty to a reference model: the KL loss is disabled and no frozen reference copy is retained. The policy is therefore free to move away from its initialization, which matters here because the internalization updates are intended to shift the policy toward reasoning it does not yet produce natively. Stability instead comes from the asymmetric clipping range, dual clipping, and the gradient-norm skip threshold.

Rollout importance sampling. Rollouts are generated by an inference engine whose numerics differ slightly from the training forward pass. We correct the resulting mismatch with token-level importance sampling, together with masking and veto thresholds that drop tokens whose log-probability ratio is implausible, so that a small number of badly mismatched tokens cannot dominate a step.

Interleaving. Following Section 2.3, one internalization update follows every 40 GRPO iterations. The internalization update is a standard autoregressive supervised update on accepted trajectories, run with the same optimizer state and learning rate as the surrounding GRPO steps.

B.1 HARD-PROBLEM SELECTION

Applying the harness to a problem the current policy can already solve wastes compute and contributes no new signal. Following Section 2.3, candidates are the queries for which none of the grouped GRPO rollouts satisfies the correctness criterion; before each internalization update these are screened again

by drawing $K = 32$ additional native responses per query from the *current* policy, and only queries with no correct response in this screening are retained. Both stages therefore measure difficulty relative to the policy at that point in training, and both are re-run in every cycle, so a query retired as solvable in one cycle may re-enter in a later one only by being re-screened against the updated policy.

C Evaluation Details

C.1 Proof verification

Roughly half the evaluation is proof-style, where a correct final answer can coexist with an unsound derivation, so answer matching is not sufficient. Answer-based benchmarks (AMO-Bench, IMO-AnswerBench, APEX-2025) are graded by extracting the final answer and comparing it against ground truth. Proof-style benchmarks (IMO-Proof-Bench, UGD, QE, PhD-Paper) are graded by the step-wise procedure below. Both routes return a binary per-response label, and the same label is used for reported accuracy, for the RL reward, and for accepting reconstructed trajectories.

Step-wise verification. A verifier reads the problem and the candidate proof and reports the index of the first erroneous step, or -1 if it finds none. Where a reference solution is available it is supplied to the verifier; where the benchmark provides explicit grading criteria, a criteria-aware template is used instead.

Pessimistic aggregation. Each verifier is sampled 8 times per response. A response is labelled correct only if the fraction of votes reporting an error is below a threshold of $1/8$; that is, *a single dissenting vote out of eight is enough to mark a proof unsound*. This is deliberately conservative: for the proof-style regime we target, a verifier that overlooks a gap is more damaging than one that is too strict, since the former inflates reported accuracy and admits unsound trajectories into training.

Two verifiers and an adjudicator. Because a single verifier family has systematic blind spots, every response is judged independently by two different backbones—Deepseek V4 Pro and GLM-5.2—each aggregated to a binary label as above. When the two agree, their shared label is final. When they disagree, the dissenting errors flagged by whichever verifier judged the proof unsound are passed to a third adjudicator call, which decides whether the alleged error is real; the response is labelled incorrect only if the adjudicator confirms it. Responses for which neither verifier produces a parseable vote are excluded; responses with no verifier output at all are counted as incorrect.

C.2 The PhD-Paper benchmark

PhD-Paper is an internal benchmark of 26 research-level proof problems curated from published mathematics papers, most of them arising from a gap or an under-justified step in a published argument. It is the hardest and smallest of our evaluation sets, and the regime the method is aimed at: the strongest reference model we evaluate reaches 61.5 pass@1, and our untrained 397B backbone reaches 16.1.

Each problem carries a reference solution, and 18 of the 26 additionally carry explicit evaluation criteria used by the criteria-aware verifier template. Problem statements are short relative to the proofs they require—a median of 413 characters, ranging from 181 to 1841. Because the set is small, every problem is evaluated with 8 repeats and scored by the two-verifier procedure of Section C.1; with 26 problems a single problem is worth 3.8 points of pass@1, so differences of a few points should be read with that granularity in mind.

Public research-level proof benchmarks are thin, and contamination risk is acute for competition problems, which is why we maintain this set internally. We plan to release it.

D Per-Benchmark Self-Evolution

Figure 5 decomposes the averaged self-evolution loop of §3.3 by benchmark. Two patterns are visible that the average conceals. The loop compounds most on the hard benchmarks: on QE the base model moves $23.3 \to 44.2 \to 42.2 \to 50.3 \to 55.0$ and on PhD-Paper $13.2 \to 18.0 \to 19.7 \to$

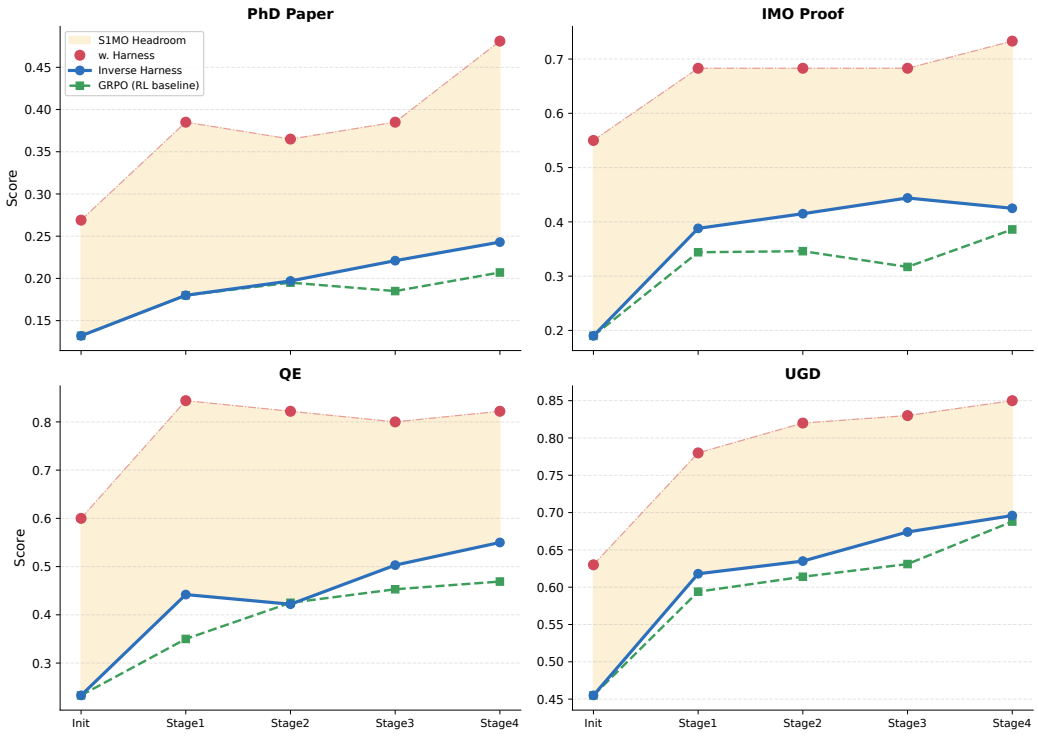


Figure 5: **Per-benchmark view of the self-evolution loop.** At each stage we plot the harnessed model (●, upper curve), the HAIR model that results from internalizing it (—), and a GRPO baseline trained for the same number of steps (– –). The shaded band is the headroom the harness opens over the current policy. The harnessed curve is not a fixed ceiling: because the harness is applied to whatever policy the previous stage produced, lifting the base model also lifts the harness, so the band is carried upward rather than consumed.

$22.1 \rightarrow 24.3$, with roughly half the total gain arriving after the first round and the gap over GRPO widening to $+8.1$ on QE by stage 4. The headroom band likewise fails to close where it matters most—on PhD-Paper it *widens* from 13.7 to 23.8, and elsewhere narrows only modestly (IMO-Proof $36.0 \rightarrow 30.8$).

UGD behaves differently, and instructively so. It is the only benchmark where the stage-4 harness-free model (69.6) overtakes the *harnessed* model at initialization (63.0), and also the one where the GRPO baseline converges to within 0.8 of HAIR (68.8 vs. 69.6)—whereas on QE the gap is ten times larger. As the easiest benchmark, ordinary RL already recovers most of what is available, so there is little left for an external operator to add. The loop is thus self-limiting in the right way: it pays off where the policy is weak and quietly stops mattering where the policy is already strong. Notably the harness retains headroom on UGD (15.4 at stage 4) that the base model does not absorb, which suggests the binding constraint there is what RL can learn rather than what the harness can solve.