LEARNING TO EXPLORE FOR SELF-IMPROVING PARALLEL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

A self-improving reasoner must generate not only answers, but also experience from which it can learn. Native parallel reasoning offers multiple solution routes; turning those routes into useful exploration requires learning what to investigate and how to reuse the findings. We introduce Self-Improving Parallel Reasoning (SIPR), a training framework that connects strategy-guided exploration to autonomous reasoning and outcome-based learning. Mathematical strategies first guide the construction of multi-round demonstrations. Branches develop concrete plans from completed shared history, while current-round findings become available only after synthesis. Successful demonstrations retain intermediate candidates, including incorrect ones, so later reasoning can compare, check and revise earlier work. Supervised fine-tuning learns the visible plans, branch steps and takeaways without the private strategy directives. Reinforcement learning then updates these actions jointly from reference-scored, self-generated trajectories, using their recorded execution contexts. The updated policy generates the next batch of experience. We instantiate SIPR in two 4B model families. In the reported Qwen3-4B evaluation, ordinary RL improves over SFT on all seven mathematical benchmarks, averaging 2.31 percentage points. On 800 paired AMC/AIME responses per policy, literal within-round plan repetition falls from 29.63% to 23.38%. These results connect learning from outcomes with learning to organize the exploration that produces them.

1 INTRODUCTION

A reasoning model that learns from its own solutions also shapes the experience available for its next update. Self-training and reasoning reinforcement learning exploit this feedback loop: generated solutions are evaluated and used to improve the policy that produces subsequent ones (Zelikman et al., 2022; Huang et al., 2022; Singh et al., 2024; DeepSeek-AI et al., 2025). This raises a question central to recursive self-improvement (RSI): *how should a model organize its exploration so that its own experience becomes more useful for learning?* Final-answer feedback identifies successful outcomes, but leaves the model to discover which approaches to pursue, what to check and how to combine intermediate findings.

Native parallel reasoning brings these choices inside a single model. Rather than sampling several complete answers independently, the model can plan branches, develop them and merge their findings (Pan et al., 2025; Yang et al., 2025b; Wu et al., 2026). Yet more branches do not by themselves imply better exploration. Several branches may repeat the same calculation, and a useful intermediate result may never reach the final synthesis. Our starting point is to make *the organization of exploration* an explicit target of training: branches need mathematical purposes, an execution-consistent view of prior work, and opportunities to revisit what earlier rounds found.

Consider two routes through a mathematics problem: probing a small case can reveal a pattern, while a change of variables can expose an invariant. Their value lies in creating findings that subsequent reasoning can test and use. This suggests two complementary requirements for demonstrations. They should specify *what each branch investigates* through concrete plans, and *what each branch can observe* through a shared history of completed rounds. A branch can contribute a useful intermediate calculation before it has a complete solution. Retaining a candidate that later proves wrong also preserves the state in which comparison or correction becomes useful.

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.

We introduce Self-Improving Parallel Reasoning (SIPR), which connects this demonstration design to autonomous exploration and continued policy learning. During initialization, mathematical strategies guide branch-specific plans and incremental reasoning. Inspired by explicit composition of reasoning heuristics (Zhou et al., 2024), a selector assigns applicable strategies and removes proposals judged to repeat the same key transformation. The branch then generates the problem-specific derivation: a strategy specifies an approach, while its plan identifies concrete work to carry out. Branches advance one step at a time, retaining pending work for later rounds.

History sharing makes these intermediate findings available for continued exploration. Every construction branch in a round receives the same frozen history of completed rounds. Its current step cannot use sibling step bodies that have not yet been shared. Synthesis records the completed findings, candidate answers and remaining gaps; the calculations and takeaway then enter the history used by later rounds. A subsequent step can develop an unfinished route, check a candidate against another constraint or revisit a disagreement. We select demonstrations with correct final answers while preserving intermediate candidates, including incorrect ones, within surviving branches. The retained states therefore provide examples of reasoning before and after a candidate is checked, alongside successful final conclusions.

The transition to autonomous reasoning removes the external strategy assignment. Private directives, pending queues and construction metadata are omitted from supervision; the model learns the visible plans, branch steps and takeaways. The resulting policy must generate the plans that guide its own branches and decide how to continue and synthesize their work. Outcome-based reinforcement learning then jointly updates planning, branch development and synthesis from that policy's own trajectories. All sampled actions share a trajectory-level advantage, while each action is scored in the execution context that produced it. Ordinary RL retains invalid trajectories in its outcome feedback, allowing failures to contribute to the same learning process as successful responses.

The updated policy generates the next batch of experience, connecting two forms of reuse. Completed findings condition later rounds *within a solution*, with model parameters fixed. Reference-scored trajectories update the policy *across training batches*, changing the generator of future plans, branch steps and synthesis. The strategy catalog is used only during initialization; reference answers remain the external learning signal. This procedure makes the organization of exploration part of the policy that is repeatedly updated from its own experience.

This focus complements existing approaches to self-improvement. TRT reuses strategies at test time, Dream-RSI optimizes an external exploration policy, and RISE and SCoRe train iterative self-correction (Zhuang et al., 2026; Zheng et al., 2026a; Qu et al., 2024; Kumar et al., 2024). OPE learns outline-guided exploration, while native parallel systems provide representations and training procedures for generating and combining branches (Guo et al., 2026a; Yang et al., 2025b; Wu et al., 2026). SIPR studies strategy-guided, multi-round exploration demonstrations and their use in subsequent autonomous learning. Its focus is the experience supplied to learning and the information available when that experience is generated (Section 5).

We evaluate the framework in the Qwen3-4B and Qwen3.5-4B families. The principal Qwen3 comparison shows positive SFT-to-RL accuracy changes on all seven mathematical benchmarks, with a macro-average gain of 2.31 percentage points. On 800 paired AMC/AIME responses per policy, literal within-round plan repetition falls from 29.63% to 23.38%. Matched examples show later rounds checking and correcting earlier candidates: in a walking-time problem, RL rejects an inconsistent candidate and carries a correct branch result into the final conclusion. These observations connect the learning comparison to inspectable changes in how generated findings are used. Our contributions are:

- **Exploration-oriented demonstrations:** branch-specific mathematical strategies, frozen shared history and retained intermediate candidates make the organization of reasoning explicit in supervision.
- **A connected learning procedure:** structured SFT initializes autonomous planning, followed by outcome-based updates on self-generated actions in their recorded execution contexts.
- **An empirical study of the learning loop:** benchmark results, paired uncertainty estimates and inspectable trajectories characterize accuracy changes and the use of prior findings.

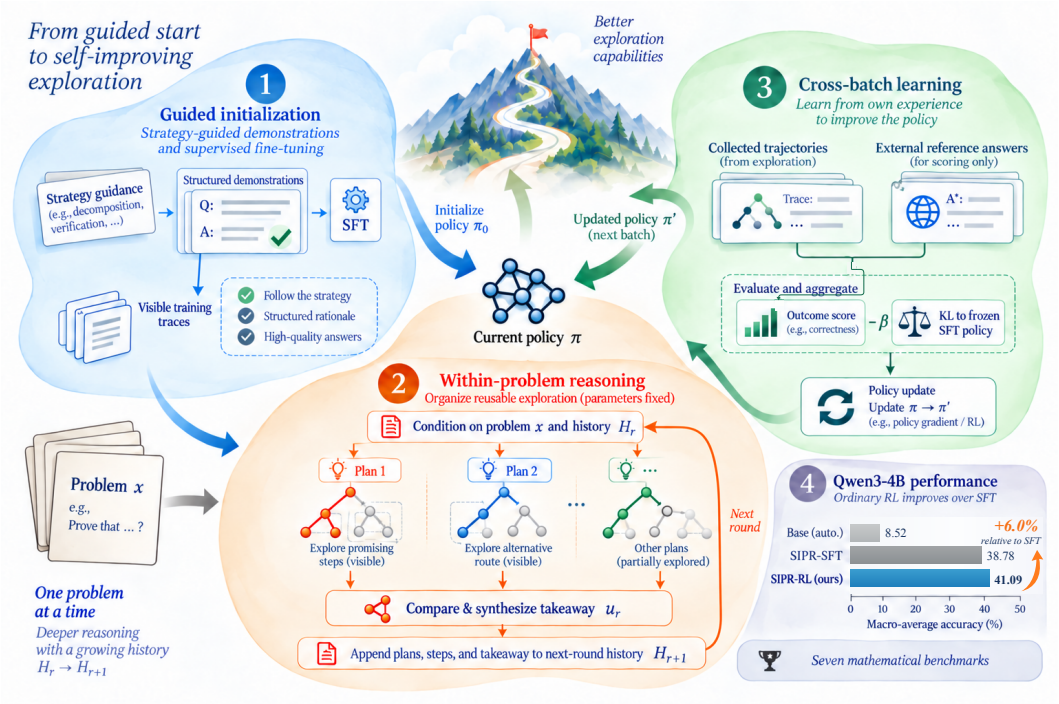

Figure 1: **Learning to organize experience for self-improving parallel reasoning.** **(1)** Strategies guide demonstrations; final-answer filtering retains intermediate candidates, and SFT omits private directives. **(2)** Schematic branches read completed history and their own plans, with sibling step bodies hidden until synthesis. Later rounds reuse completed findings with parameters fixed. **(3)** Outcome feedback jointly updates planning, branch execution and synthesis; the updated policy generates further experience. **(4)** Accuracy (%) for only the three displayed Qwen3-4B settings, equally averaged over seven math benchmarks using original responses (Table 1); gains compare ordinary RL with SFT. Answer extraction and scoring follow Appendix C. Reference answers supervise construction and RL, not test-time reasoning.

2 Organizing Exploration for Self-Improvement

SIPR uses experience within solutions and across learning stages. Completed findings inform later reasoning rounds; scored trajectories update the policy generating later solutions. Strategy-guided demonstrations initialize purposeful branch work under the information available during parallel execution (Figure 1).

2.1 Structured Exploration

For problem x, let H_r be completed history before round r. Each round contains plans $P_r = (p_{r,1}, \ldots, p_{r,k_r})$, branch steps $S_r = (s_{r,1}, \ldots, s_{r,k_r})$ and takeaway u_r. The native format (Wu et al., 2026) places serial `<plan>` blocks inside a `<guideline>`, followed by ID-matched `<step>` blocks and a `<takeaway>`. We serialize $H_{r+1} = H_r \,\|\, (P_r, S_r, u_r)$, retaining calculations alongside summaries for later checking. Here $\|$ denotes concatenation, and H_1 contains the prompt and initial orientation.

Within a round, branches develop their assigned steps without reading sibling step bodies. Let $\mathcal{H}_r$, $\mathcal{P}_{r,i}$ and $\mathcal{S}_{r,i}^{\le t}$ denote the token positions of completed history, branch i's plan and its causal step prefix. Let $\mathcal{G}_r$ contain the two guideline delimiters. The representation at input-token position t, which predicts the next action in step i, has direct visibility

$$\mathcal{V}(t) = \mathcal{H}_r \cup \mathcal{G}_r \cup \mathcal{P}_{r,i} \cup \mathcal{S}_{r,i}^{\le t}. \tag{1}$$

The takeaway reads all preceding steps, making their findings available to the next round. Figure 2 shows the attention and logical-position geometry used by the fork/merge runtime (Wu et al., 2026). Plans remain serial; sibling steps share a logical origin, and synthesis starts after the longest step. Plans share prior context, so the mask specifies direct token visibility rather than statistical independence. Appendix A.3 details execution and replay.

2.2 Strategy-Guided Demonstrations

A selector chooses applicable directives from 49 mathematical strategies in 13 families, specifying a setup, key transformation and target reduction while removing proposals judged to repeat the same transformation. Smallest-case probing, for example, seeks a pattern; a change of variables seeks a simpler representation. Strategies specify guiding principles rather than complete solutions, as in abstraction-guided prompting (Zheng et al., 2023a); Appendix B.2 gives the catalog.

Each branch receives its directive, brief plan, problem difficulty, local pending-step queue and the same frozen H_r. It derives a problem-specific plan and executes one pending step, aiming to produce a checkable value, identity, bound or resolved case. Unfinished plans remain in its queue for subsequent rounds. Current-round outputs are buffered outside H_r, even when construction calls are awaited sequentially. Demonstrations therefore respect the branch-step visibility used in parallel execution.

After the accepted steps, a synthesizer compares findings, records remaining gaps and treats branch answers as candidates to check rather than conclusions to copy. The resulting takeaway and branch calculations enter shared history. Later rounds can develop an unfinished route, test a candidate against another constraint or resolve a disagreement. The synthesizer can retire redundant routes while retaining their completed work. A branch may also finish after an answer candidate or an empty queue. Construction uses difficulty-dependent round limits and retry controls; the complete stopping procedure is in Algorithm 1. These controls build demonstrations and are not imposed as a fixed inference schedule.

2.3 From Demonstrations to Autonomous Planning

Qwen3-4B-Instruct-2507 (Qwen Team, 2025) generates the initialization corpus from Polaris mathematics (POLARIS Project, 2025). Selection targets three to six first-round plans, deduplicates examples and applies a 25,000-token cutoff, yielding 10,000 demonstrations. Accepted traces must have a correct boxed answer in the stored final takeaway and at least two surviving directives. The builder removes routes exhausted by failed generation retries and renumbers surviving headers. It preserves intermediate boxed candidates, including incorrect ones, and appends the selected final answer to the last takeaway. The supervision thus contains states in which a useful next action is to compare or correct prior work, not merely reproduce a successful final derivation.

Only the problem and visible plans, steps and takeaways enter supervised training. Private strategy definitions, selection-stage brief plans, difficulty labels, pending queues and retirement metadata are removed. The learned policy must express a branch's task through its own generated plan, replacing the external selector used during construction. Construction and deployment differ in how plans are produced: construction calls do not see other current-round plans, whereas autonomous plan i follows plans $1, \ldots, i-1$. Both withhold current sibling step bodies. Appendix B details corpus selection and metadata removal.

At inference, the policy chooses plans, branch development and synthesis within a shared 30,000-token response limit and at most nine plans per round, without a fixed per-branch token allocation or round cap. Completed history supports continuation and revision; the policy also decides when to synthesize and terminate. Section 4.3 examines the resulting behavior.

3 Learning from Self-Generated Exploration

3.1 Learning the Initial Exploration Policy

Supervised initialization precedes feedback-based RL, a common language-model training sequence (Ouyang et al., 2022). We add eight structural delimiter tokens to Qwen3-4B Base. We first freeze the transformer and train its input embeddings and output head on 1,000 examples, then train all

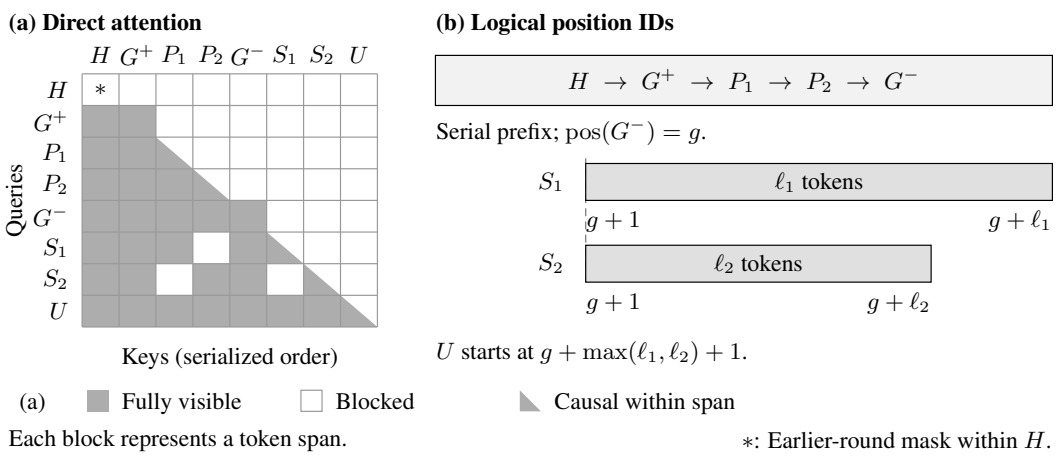

Figure 2: **Attention masks and logical positions for one reasoning round.** H is completed history; G^+, G^- delimit the guideline; P_i, S_i are paired plans and steps; U is the takeaway. Rows are queries and columns are keys. Fully shaded blocks allow all token pairs, white blocks block them, and triangles allow only causal attention within a span; $*$ preserves the history's existing mask. Each step reads its own plan and prefix, while synthesis reads all branches. Plans retain serial positions; sibling steps start at $g + 1$, and U follows the longest step. Blocks represent token spans, not equal token counts.

parameters on the 10,000-example corpus. Cross-entropy supervision covers assistant plans, branch steps, takeaways and end-of-sequence under the branch attention masks and logical positions; prompt targets are ignored. This initializes autonomous exploration without assigning correctness labels to intermediate steps. Appendix A gives the full recipe and the model-specific Qwen3.5 implementation.

3.2 Updating the Policy in Its Execution Contexts

After initialization, the current policy generates its own plans, branch steps and synthesis. Reference-answer feedback scores the resulting trajectories, and each accepted update changes the policy used to collect the next batch. This closes the learning loop without invoking the strategy catalog again. Qwen3 RL uses 17,917 prompts from DAPO-Math-17k (Yu et al., 2025), sampling $P = 16$ problems and $K = 4$ trajectories per problem for each batch. The ordinary RL variant retains invalid trajectories rather than replacing them.

Execution-context scoring. A serialized transcript is not the policy's execution history: one printed branch can follow another even though it never attended to that branch's body. We therefore record each sampled token a with the predictor context h_a that produced it, including the logical position and attention view. Let $\mathcal{I}_i$ index the sampled actions in trajectory i; repeated occurrences of the same token are distinct actions. Runtime-inserted headers condition later actions but are not sampled actions and receive no direct policy-gradient term. Recomputing probabilities in these contexts preserves the visibility in Equation 1, including for incomplete trajectories.

Outcome-based updates. A trajectory receives $R_i^{\text{out}} = +1$ for a valid, normally terminated response with a correct global answer, and -1 otherwise. Reference answers are used for scoring, not included in generation prompts. Appendix A.2 defines validity and answer extraction; these training conditions are stricter than content-based evaluation. With collection policy π_b and fixed SFT reference π_{ref}, the return and leave-one-out advantage are

$$R_i = R_i^{\text{out}} - \beta \sum_{a \in \mathcal{I}_i} \log \frac{\pi_b(a \mid h_a)}{\pi_{\text{ref}}(a \mid h_a)}, \qquad \beta = 0.001, \tag{2}$$

$$A_i = R_i - \frac{1}{K-1} \sum_{j \in \mathcal{G}(i) \setminus \{i\}} R_j, \tag{3}$$

where $\mathcal{G}(i)$ contains trajectories for the same problem. The reference-policy term is a detached sampled reward, not an additional differentiable loss. Equation 3 is the sum of the implementation's outcome and reference-policy advantages, written with a single return.

All sampled actions share the trajectory advantage. This lets final success train the choice of plans, the work done in branches and the use of their findings, without assuming that branch contributions are separable or supplying step-level rewards. We apply leave-one-out REINFORCE (Williams, 1992; Ahmadian et al., 2024) with the loss

$$\mathcal{L}_{\mathrm{RL}} = -\frac{1}{PKC}\sum_{i=1}^{PK}\mathrm{sg}(A_i)\sum_{a\in\mathcal{I}_i}\log \pi_\theta(a \mid h_a), \tag{4}$$

where sg stops gradients and $C > 0$ is a fixed normalization constant shared by all trajectories (Liu et al., 2025). Each collected batch supplies one full-batch gradient and at most one accepted update, without an importance ratio or PPO clipping (Schulman et al., 2017). Collection uses temperature 1 and top-$p = 1$; optimization uses learning rate 10^{-6} and gradient clipping at 1.0. Appendix A.2 gives the fixed constant and update-acceptance rule. The key connection is that the policy learns from the same information contexts in which it explored, then uses its updated parameters to produce further experience.

4 Experiments

We examine three questions: does learning from self-generated exploration improve accuracy, how does the organization of exploration change, and how do the resulting policies relate to published parallel reasoners? The principal comparison is SIPR-SFT versus ordinary SIPR-RL within a model family and grading protocol.

4.1 Models and Evaluation

We evaluate Qwen3-4B (Yang et al., 2025a) and Qwen3.5-4B (Qwen Team, 2026). Base uses four-shot serial generation; trained models use native parallel reasoning. The seven mathematical benchmarks are MATH-500 (Hendrycks et al., 2021; Lightman et al., 2023), AMC23, AIME24 and AIME25 (Mathematical Association of America, n.d.), HMMT February 2025 (Harvard–MIT Mathematics Tournament, 2025), Minerva (Lewkowycz et al., 2022) and English text-only OlympiadBench (He et al., 2024). GPQA Diamond provides an additional scientific-reasoning evaluation (Rein et al., 2023). Native-parallel generation uses temperature 1, top-$p = 0.7$ and a 30,000-token response limit.

Metric and comparison scope. Mean accuracy counts correct responses over all responses in each reported evaluation, including invalid, truncated and unresolved ones in the denominator. AMC/AIME/HMMT average eight responses per question; Appendix C specifies the scoring rules. Avg. equally weights the seven mathematics estimates. Qwen3 trained models are scored on the last complete boxed candidate, including candidates in unfinished branches; Qwen3.5 is scored on the retained target answer, including explicit unboxed conclusions. Candidates are checked against the original question and reference answer. SFT and RL share the scoring rule within each family. C denotes training that replaces invalid trajectories until each prompt has eight valid responses, including valid incorrect ones. R denotes evaluation with additional sampling and reference-answer-assisted historical rescue on the same checkpoint. Table 2 reports original-response coverage; Section 4.5 compares C and R.

4.2 Accuracy after Learning from Exploration

Ordinary Qwen3 SIPR-RL improves over SIPR-SFT on all seven mathematical benchmarks in Table 1. The reported macro-average rises from 38.78% to 41.09%, a gain of 2.31 percentage points with a question-paired 95% interval of $[+0.52, +4.02]$. The largest increase is on AIME25, from 13.75% to 18.33%; AIME24 rises from 15.83% to 18.75%. Appendix D gives the paired intervals and their conditioning assumptions.

On Qwen3.5-4B, SIPR-RL improves mean accuracy on six of the seven mathematics benchmarks, raising the seven-benchmark average from 48.94% to 49.99% (+1.05 points) and OlympiadBench

Table 1: **Accuracy before and after outcome-based learning (%).** The second header row gives Mean@1 or Mean@8. Avg. equally weights the seven math benchmarks; GPQA is reported alongside them. Small superscripts give RL–SFT differences in percentage points, with positive gains in green and negative changes in red. Bold highlights the better trained setting within each family. —: not reported.

Model	MATH-500 Mean@1	AMC23 Mean@8	AIME24 Mean@8	AIME25 Mean@8	HMMT Mean@8	Minerva Mean@1	Olymp. Mean@1	GPQA Mean@1	Avg. Math
Qwen3-4B									
Base	26.30	14.69	1.25	0.83	0.00	8.82	7.72	25.25	8.52
SIPR-SFT	79.80	55.94	15.83	13.75	7.92	51.47	46.74	30.81	38.78
SIPR-RL	**82.20**$^{+2.40}$	**58.13**$^{+2.19}$	**18.75**$^{+2.92}$	**18.33**$^{+4.58}$	**8.75**$^{+0.83}$	**53.13**$^{+1.66}$	**48.37**$^{+1.63}$	**35.35**$^{+4.54}$	**41.09**$^{+2.31}$
Qwen3.5-4B									
Base	47.20	21.88	1.67	0.83	0.42	19.12	15.88	35.35	15.28
SIPR-SFT	85.00	70.31	25.42	18.75	12.50	**75.74**	54.90	**47.98**	48.94
SIPR-RL	**85.60**$^{+0.60}$	**71.56**$^{+1.25}$	**27.92**$^{+2.50}$	**20.42**$^{+1.67}$	**14.17**$^{+1.67}$	72.43$^{-3.31}$	**57.86**$^{+2.96}$	45.45$^{-2.53}$	**49.99**$^{+1.05}$

from 54.90% to 57.86% (+2.96 points). These gains extend the improvement beyond Qwen3-4B, while Minerva and GPQA decrease and HMMT Pass@8 remains 30.00% despite higher mean accuracy.

Table 2: **Problem coverage with eight original responses.** Pass@8 (%) retains every response in the fixed budget. C denotes completion training; superscripts give same-family RL–SFT differences in percentage points. Bold marks the best comparable scores. —: not reported.

Model	AMC23 Pass@8	AIME24 Pass@8	AIME25 Pass@8	HMMT Pass@8
Qwen3-4B				
SIPR-SFT	80.00	30.00	30.00	20.00
SIPR-RL	85.00$^{+5.00}$	30.00$^{+0.00}$	26.67$^{-3.33}$	23.33$^{+3.33}$
SIPR-RL-C	**95.00**$^{+15.00}$	**36.67**$^{+6.67}$	**33.33**$^{+3.33}$	**26.67**$^{+6.67}$
Qwen3.5-4B				
SIPR-SFT	92.50	46.67	36.67	30.00
SIPR-RL	**95.00**$^{+2.50}$	**56.67**$^{+10.00}$	**40.00**$^{+3.33}$	30.00$^{+0.00}$

4.3 How the Exploration Changes

Table 7 (Appendix D) measures behavior beyond final-answer accuracy. Responses without repeated within-round plans rise from 563/800 (70.38%) to 613/800 (76.63%), a gain of 6.25 points. Responses containing a boxed expression rise from 674/800 (84.25%) to 710/800 (88.75%), a gain of 4.50 points. Both paired intervals exclude zero. These measures support reduced literal planning redundancy and more frequent answer candidates; boxed-answer presence does not establish correctness. Ordinary SIPR-RL's mean training reward rises from -0.470 over updates 1–20 through -0.411 to -0.368 over updates 41–60 (Appendix E.2).

Turning a candidate into a checked conclusion. In an AIME24 walking-time problem, the SFT response contains a correct branch calculation but runs out of budget before completing synthesis. RL rejects an earlier candidate because it implies negative walking time, uses the correct branch result and concludes with 204 minutes. In an AIME25 base-conversion problem, SFT treats 17_b as $10b + 7$, whereas RL uses $17_b = b + 7$, derives $b + 7 \mid 56$, checks $b = 21, 49$ and returns 70. The relevant operation is not just producing another answer: it is testing a candidate and carrying supported findings into later reasoning. These examples belong to 12 deterministically selected pairs spanning improved, regressed and unchanged outcomes; Appendix D gives the selection rule and arithmetic checks, and the accompanying materials retain the literal excerpts.

4.4 Context from Published Parallel Reasoners

Table 3: **Comparison with published Qwen3-4B results (%).** Each row retains its source metric. Parallel-R1 S1/S2 are Unseen settings; SSFT uses learned matching. Bold highlights the best displayed SIPR setting per benchmark. R includes resampling and reference-answer-assisted historical rescue, with one target slot per MATH-500 problem and eight per AIME problem. Parallel-R1 uses filtered MATH-500; generation budgets and grading protocols differ across methods (Appendix C.2).

(a) MATH-500

Method	Metric	MATH-500
Parallel-R1 S1	Mean@1	82.60
Parallel-R1 S2	Mean@1	84.50
SSFT	Pass@1	81.85
SIPR-RL-R	Slot accuracy	**84.60**

(b) AIME

Method	Metric	AIME24	AIME25
Parallel-R1 S1	Mean@16	18.30	17.70
Parallel-R1 S2	Mean@16	16.30	19.00
SIPR-RL	Mean@8	18.75	18.33
SIPR-RL-R	Slot accuracy	**22.92**	19.17
SIPR-RL-C	Mean@8	20.00	**20.42**

Table 3 situates the results against Parallel-R1 (Zheng et al., 2026b) and SSFT (Jia et al., 2026), using MATH-500 for both published methods and AIME for Parallel-R1. These are complementary reference points: Parallel-R1 changes the training curriculum, SSFT learns forking modes, and SIPR targets multi-round exploration experience. On MATH-500, SIPR-RL-R reaches 84.60%, with 423 correct answers among 500 target slots. The primary learning evidence remains the within-protocol SFT-to-RL comparison. Appendices C.2 and E.3 specify the source protocols and additional attempts.

4.5 Completion Training and Inference Resampling

Table 4 reports completion-trained RL and ordinary RL with inference resampling alongside their SFT references.

Table 4: **Completion training and inference resampling on Qwen3-4B (%).** C denotes completion-trained RL; R uses resampling and reference-assisted rescue (Table 3); accuracy divides credited correct answers by all target responses, counting unfilled positions as incorrect. Headers give Mean@1/Mean@8 for original rows and target slots for R rows. Avg. uses all seven math benchmarks. Gains are descriptive RL–SFT differences within each displayed evaluation setting; bold highlights the higher scores. —: not reported.

Model	MATH-500 Mean@1	AMC23 Mean@8	AIME24 Mean@8	AIME25 Mean@8	HMMT Mean@8	Minerva Mean@1	Olymp. Mean@1	Avg. Math
SIPR-SFT	79.80	55.94	15.83	13.75	7.92	51.47	46.74	38.78
SIPR-RL	**82.20**$^{+2.40}$	58.13$^{+2.19}$	18.75$^{+2.92}$	18.33$^{+4.58}$	8.75$^{+0.83}$	**53.13**$^{+1.66}$	**48.37**$^{+1.63}$	41.09$^{+2.31}$
SIPR-RL-C	80.00$^{+0.20}$	**61.56**$^{+5.62}$	**20.00**$^{+4.17}$	**20.42**$^{+6.67}$	**10.42**$^{+2.50}$	49.26$^{-2.21}$	48.22$^{+1.48}$	**41.41**$^{+2.63}$
Resampling and rescue: credited slot accuracy (%); @1/@8 count target responses								
SIPR-SFT-R	81.00	59.38	18.75	17.08	**9.17**	**53.31**	49.55	41.18
SIPR-RL-R	**84.60**$^{+3.60}$	**63.13**$^{+3.75}$	**22.92**$^{+4.17}$	**19.17**$^{+2.09}$	**9.17**$^{+0.00}$	52.57$^{-0.74}$	**51.48**$^{+1.93}$	**43.29**$^{+2.11}$

SIPR-RL-C replaces invalid training trajectories until each prompt has eight valid ones. Its different batch and training budgets make it a separate RL recipe rather than an isolated validity-filter ablation. It averages 41.41% across the seven math benchmarks and reaches 61.56% on AMC23, 20.00%/20.42% on AIME24/25 and 10.42% on HMMT, with less consistent changes across benchmarks than ordinary RL.

R retries invalid responses and uses reference answers to credit earlier correct responses. This *slot accuracy* uses a fixed denominator of one target response per problem for Mean@1 benchmarks and eight for Mean@8 benchmarks; R gains describe differences in credited-slot accuracy; response pools and generation counts can differ across models. This offline evaluation requires reference answers. Ordinary SIPR-RL-R reaches 84.60% on MATH-500, 63.13% on AMC23 and 22.92% on AIME24,

averaging 43.29% across seven math benchmarks. Appendix E.3 gives the resampling and scoring procedures.

5 Related Work

Learning from generated reasoning. Generated supervision supplies instruction examples and selected successful rationales for fine-tuning (Wang et al., 2022b; Zelikman et al., 2022; Huang et al., 2022; Singh et al., 2024); rationale distillation transfers reasoning across model sizes (Hsieh et al., 2023). Iterative learning can contrast earlier policy responses with demonstrations or use model-generated preference judgments (Chen et al., 2024; Yuan et al., 2024). SIPR preserves plans, partial branch work and synthesis in its initialization corpus, then learns from its own trajectories through outcome-based RL.

Revision within and across attempts. Self-feedback, verbal memories and reusable strategies retain experience in context (Madaan et al., 2023; Shinn et al., 2023; Zhuang et al., 2026; Yang et al., 2024b). RISE, SCoRe and S^2R train iterative correction (Qu et al., 2024; Kumar et al., 2024; Ma et al., 2025a); Dream-RSI improves an external exploration policy (Zheng et al., 2026a). LatentSeek optimizes instance-specific latent representations with fixed model weights (Li et al., 2025). SIPR connects within-solution history reuse to across-batch updates of the native parallel policy. Reference-answer feedback grounds these updates; prompted self-correction without external feedback can fail to improve reasoning (Huang et al., 2023).

Planning the exploration space. Reasoning can use sequential decomposition and explicit plans (Zhou et al., 2022; Wang et al., 2023a), intermediate deductions and sampled-path aggregation (Wei et al., 2022; Wang et al., 2022a), or search over states and dependencies (Yao et al., 2023; Besta et al., 2023; Hao et al., 2023; Zhou et al., 2023). SELF-DISCOVER composes reasoning modules, while OPE learns outline planning and outline-conditioned reasoning (Zhou et al., 2024; Guo et al., 2026a). SIPR instead makes branch work incremental: a strategy seeds a route, its plan specifies the next task, and synthesis shares findings for later comparison. Difficulty-dependent test-time compute allocation is complementary (Snell et al., 2024).

Native parallel reasoning. APR, Multiverse and ParaThinker learn to generate and combine concurrent reasoning branches (Pan et al., 2025; Yang et al., 2025b; Wen et al., 2025). NPR combines structured trajectories, parallel warmup, a modified runtime and self-distilled RL (Wu et al., 2026). Parallel-R1 bootstraps parallel reasoning on easier problems before general mathematics RL (Zheng et al., 2026b); SSFT learns forking modes through trace matching, followed by GFPO (Jia et al., 2026). SIPR focuses on strategy-guided exploration experience, execution-consistent visibility and intermediate candidates available across rounds.

Feedback and credit assignment. Outcome-based RL strengthens reasoning in several training systems (Shao et al., 2024; DeepSeek-AI et al., 2025; Yu et al., 2025; Hu et al., 2025), with broader-domain training and instruction quality providing additional factors (Ma et al., 2025b; Guo et al., 2026b). Benchmarks extend this evaluation to scientific problem solving and olympiad-level mathematics (Wang et al., 2023c; Sun et al., 2025). Process supervision assesses intermediate deductions (Uesato et al., 2022; Lightman et al., 2023; Wang et al., 2023b); SCRL and Parallel Shapley introduce subproblem- and path-level feedback (Jiang et al., 2026; Zhang et al., 2026). SIPR uses leave-one-out REINFORCE (Ahmadian et al., 2024), sharing a trajectory advantage across planning, branch actions and synthesis.

6 Conclusion

SIPR treats the organization of exploration as part of what a reasoning model learns. Strategy-guided demonstrations initialize branch planning and cross-round reuse; outcome-based updates then train the autonomous policy on its own exploration. The reported accuracy gains, reduced literal plan repetition and inspectable cases connect this design to improved reasoning after training. The central lesson is to learn not only from outcomes, but also how to generate and reuse the experience that precedes them.

AI USE STATEMENT

AI tools assisted with figure preparation and literature search.

ETHICS STATEMENT

This work studies mathematical reasoning. Parallel branches can repeat errors or consume substantial computation, and correct final answers can coexist with invalid intermediate deductions. Applications requiring reliable reasoning need further validation of correctness, resource use and stopping behavior.

REPRODUCIBILITY STATEMENT

The appendices specify the training recipe, demonstration construction, evaluation protocols and qualitative cases. The accompanying materials retain scored-output records, source manifests, literal case excerpts, editable figures and scripts that regenerate and verify the reported tables. These support verification of the saved evidence; reproducing model training additionally requires the original training resources.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting REINFORCE style optimization for learning from human feedback in LLMs. *arXiv preprint arXiv:2402.14740*, 2024. URL https://arxiv.org/abs/2402.14740.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of Thoughts: Solving Elaborate Problems with Large Language Models. *arXiv preprint arXiv:2308.09687*, 2023. URL https://arxiv.org/abs/2308.09687.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-Play Fine-Tuning Converts Weak Language Models to Strong Language Models. *arXiv preprint arXiv:2401.01335*, 2024. URL https://arxiv.org/abs/2401.01335.

DeepSeek-AI, Daya Guo, Dejian Yang, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. URL https://arxiv.org/abs/2501.12948.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. Improving Factuality and Reasoning in Language Models through Multiagent Debate. *arXiv preprint arXiv:2305.14325*, 2023. URL https://arxiv.org/abs/2305.14325.

Qi Guo, Jianing Wang, Deyang Kong, Xiangyu Xi, Jianfei Zhang, Yi Lu, Jingang Wang, Wei Wang, Shikun Zhang, and Wei Ye. OPE: Overcoming information saturation in parallel thinking via outline-guided path exploration. *arXiv preprint arXiv:2602.08344*, 2026a. URL https://arxiv.org/abs/2602.08344.

Yiju Guo, Tianyi Hu, Zexu Sun, and Yankai Lin. Less noise, more voice: Reinforcement learning for reasoning via instruction purification. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 13469–13482, 2026b. doi: 10.18653/v1/2026.findings-acl.659. URL https://aclanthology.org/2026.findings-acl.659/.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. Reasoning with Language Model is Planning with World Model. *arXiv preprint arXiv:2305.14992*, 2023. URL https://arxiv.org/abs/2305.14992.

Harvard–MIT Mathematics Tournament. Archive of February 2025. Official problems and solutions archive, 2025. URL https://www.hmmt.org/www/archive/282. Accessed September 26, 2026.

Chaoqun He, Renjie Luo, Yuzhuo Bai, et al. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, 2024. doi: 10.18653/v1/2024.acl-long.211. URL https://aclanthology.org/2024.acl-long.211/.

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

Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 8003–8017. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.findings-acl.507. URL https://aclanthology.org/2023.findings-acl.507/.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-Reasoner-Zero: An Open Source Approach to Scaling Up Reinforcement Learning on the Base Model. *arXiv preprint arXiv:2503.24290*, 2025. URL https://arxiv.org/abs/2503.24290.

Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large Language Models Can Self-Improve. *arXiv preprint arXiv:2210.11610*, 2022. URL https://arxiv.org/abs/2210.11610.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large Language Models Cannot Self-Correct Reasoning Yet. *arXiv preprint arXiv:2310.01798*, 2023. URL https://arxiv.org/abs/2310.01798.

Sheng Jia, Xiao Wang, and Shiva Prasad Kasiviswanathan. Training large language models to reason in parallel with global forking tokens. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2510.05132v3. Version 3; Table 6 for the Qwen3-4B comparison.

Xitai Jiang, Zihan Tang, Wenze Lin, Yang Yue, Shenzhi Wang, and Gao Huang. From reasoning chains to verifiable subproblems: Curriculum reinforcement learning enables credit assignment for LLM reasoning. *arXiv preprint arXiv:2605.22074*, 2026. URL https://arxiv.org/abs/2605.22074.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. Decomposed Prompting: A Modular Approach for Solving Complex Tasks. *arXiv preprint arXiv:2210.02406*, 2022. URL https://arxiv.org/abs/2210.02406.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, et al. Training language models to self-correct via reinforcement learning. *arXiv preprint arXiv:2409.12917*, 2024. URL https://arxiv.org/abs/2409.12917.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient Memory Management for Large Language Model Serving with PagedAttention. *arXiv preprint arXiv:2309.06180*, 2023. URL https://arxiv.org/abs/2309.06180.

Aitor Lewkowycz, Anders Andreassen, David Dohan, et al. Solving quantitative reasoning problems with language models. *arXiv preprint arXiv:2206.14858*, 2022. URL https://arxiv.org/abs/2206.14858.

Hengli Li, Chenxi Li, Tong Wu, Xuekai Zhu, Yuxuan Wang, Zhaoxin Yu, Eric Hanchen Jiang, Song-Chun Zhu, Zixia Jia, Ying Nian Wu, and Zilong Zheng. Seek in the dark: Reasoning via test-time instance-level policy gradient in latent space. *arXiv preprint arXiv:2505.13308*, 2025. URL https://arxiv.org/abs/2505.13308v2. Version 2.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. Encouraging Divergent Thinking in Large Language Models through Multi-Agent Debate. *arXiv preprint arXiv:2305.19118*, 2023. URL https://arxiv.org/abs/2305.19118.

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. *arXiv preprint arXiv:2305.20050*, 2023. URL https://arxiv.org/abs/2305.20050.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding R1-Zero-Like Training: A Critical Perspective. *arXiv preprint arXiv:2503.20783*, 2025. URL https://arxiv.org/abs/2503.20783.

Ruotian Ma, Peisong Wang, Cheng Liu, Xingyan Liu, Jiaqi Chen, Bang Zhang, Xin Zhou, Nan Du, and Jia Li. S^2R: Teaching LLMs to self-verify and self-correct via reinforcement learning. *arXiv preprint arXiv:2502.12853*, 2025a. URL https://arxiv.org/abs/2502.12853.

Xueguang Ma, Qian Liu, Dongfu Jiang, Ge Zhang, Zejun Ma, and Wenhu Chen. General-reasoner: Advancing LLM reasoning across all domains. In *Advances in Neural Information Processing Systems*, volume 38, 2025b. URL https://proceedings.nips.cc/paper_files/paper/2025/hash/51c52cbb4b3faa891ca0621d13efe1be-Abstract-Conference.html.

Aman Madaan, Niket Tandon, Prakhar Gupta, et al. Self-Refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*, 2023. URL https://arxiv.org/abs/2303.17651.

Mathematical Association of America. American Mathematics Competitions. Official competition program, n.d. URL https://maa.org/student-programs/amc/. AIME overview: https://maa.org/maa-invitational-competitions/. Accessed September 26, 2026.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022. URL https://arxiv.org/abs/2203.02155.

Jiayi Pan, Xiuyu Li, Long Lian, Charlie Snell, Yifei Zhou, Adam Yala, Trevor Darrell, Kurt Keutzer, and Alane Suhr. Learning adaptive parallel reasoning with language models. *arXiv preprint arXiv:2504.15466*, 2025. URL https://arxiv.org/abs/2504.15466.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *arXiv preprint arXiv:1912.01703*, 2019. URL https://arxiv.org/abs/1912.01703.

POLARIS Project. Polaris-Dataset-53K. Dataset card, 2025. URL https://huggingface.co/datasets/POLARIS-Project/Polaris-Dataset-53K. Accessed September 17, 2026.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A. Smith, and Mike Lewis. Measuring and Narrowing the Compositionality Gap in Language Models. *arXiv preprint arXiv:2210.03350*, 2022. URL https://arxiv.org/abs/2210.03350.

Yuxiao Qu, Tianjun Zhang, Naman Garg, and Aviral Kumar. Recursive introspection: Teaching language model agents how to self-improve. *arXiv preprint arXiv:2407.18219*, 2024. URL https://arxiv.org/abs/2407.18219.

Qwen Team. Qwen3-4B-Instruct-2507. Official model card, 2025. URL https://huggingface.co/Qwen/Qwen3-4B-Instruct-2507. Accessed September 26, 2026.

Qwen Team. Qwen3.5-4B-Base. Model card, 2026. URL https://huggingface.co/Qwen/Qwen3.5-4B-Base. Accessed September 26, 2026.

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

David Samuel, Vladislav Mikhailov, Erik Velldal, Lilja Øvrelid, Lucas Georges Gabriel Charpentier, Andrey Kutuzov, and Stephan Oepen. Small languages, big models: A study of continual training on languages of Norway. In *Proceedings of the Joint 25th Nordic Conference on Computational Linguistics and 11th Baltic Conference on Human Language Technologies (NoDaLiDa/Baltic-HLT 2025)*, pp. 573–608. University of Tartu Library, March 2025. URL https://aclanthology.org/2025.nodalida-1.61/.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. HybridFlow: A Flexible and Efficient RLHF Framework. *arXiv preprint arXiv:2409.19256*, 2024. URL https://arxiv.org/abs/2409.19256.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*, 2023. URL https://arxiv.org/abs/2303.11366v4. Version 4.

Avi Singh, John D. Co-Reyes, Rishabh Agarwal, et al. Beyond human data: Scaling self-training for problem-solving with language models. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2312.06585v4.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024. URL https://arxiv.org/abs/2408.03314.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding. *arXiv preprint arXiv:2104.09864*, 2021. URL https://arxiv.org/abs/2104.09864.

Haoxiang Sun, Yingqian Min, Zhipeng Chen, Wayne Xin Zhao, Zheng Liu, Zhongyuan Wang, Lei Fang, and Ji-Rong Wen. Challenging the boundaries of reasoning: An olympiad-level math benchmark for large language models. *arXiv preprint arXiv:2503.21380*, 2025. URL https://arxiv.org/abs/2503.21380v1. Version 1; numerical-answer benchmark.

Shubham Toshniwal, Wei Du, Ivan Moshkov, Branislav Kisacanin, Alexan Ayrapetyan, and Igor Gitman. OpenMathInstruct-2: Accelerating AI for Math with Massive Open-Source Instruction Data. *arXiv preprint arXiv:2410.01560*, 2024a. URL https://arxiv.org/abs/2410.01560.

Shubham Toshniwal, Ivan Moshkov, Sean Narenthiran, Daria Gitman, Fei Jia, and Igor Gitman. OpenMathInstruct-1: A 1.8 Million Math Instruction Tuning Dataset. *arXiv preprint arXiv:2402.10176*, 2024b. URL https://arxiv.org/abs/2402.10176.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022. URL https://arxiv.org/abs/2211.14275.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-Solve Prompting: Improving Zero-Shot Chain-of-Thought Reasoning by Large Language Models. *arXiv preprint arXiv:2305.04091*, 2023a. URL https://arxiv.org/abs/2305.04091.

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*, 2023b. URL https://arxiv.org/abs/2312.08935.

Xiaoxuan Wang, Ziniu Hu, Pan Lu, Yanqiao Zhu, Jieyu Zhang, Satyen Subramaniam, Arjun R. Loomba, Shichang Zhang, Yizhou Sun, and Wei Wang. SciBench: Evaluating college-level scientific problem-solving abilities of large language models. *arXiv preprint arXiv:2307.10635*, 2023c. URL https://arxiv.org/abs/2307.10635.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022a. URL https://arxiv.org/abs/2203.11171.

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. *arXiv preprint arXiv:2212.10560*, 2022b. URL https://arxiv.org/abs/2212.10560.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022. URL https://arxiv.org/abs/2201.11903.

Hao Wen, Yifan Su, Feifei Zhang, Yunxin Liu, Yunhao Liu, Ya-Qin Zhang, and Yuanchun Li. ParaThinker: Native parallel thinking as a new paradigm to scale LLM test-time compute. *arXiv preprint arXiv:2509.04475*, 2025. URL https://arxiv.org/abs/2509.04475.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992. doi: 10.1007/BF00992696. URL https://link.springer.com/article/10.1007/BF00992696.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In Qun Liu and David Schlangen (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6/.

Tong Wu, Yang Liu, Jun Bai, Zixia Jia, Shuyi Zhang, Ziyong Lin, Yanting Wang, Song-Chun Zhu, and Zilong Zheng. Native parallel reasoner: Reasoning in parallelism via self-distilled reinforcement learning. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2512.07461v3.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-Math Technical Report: Toward Mathematical Expert Model via Self-Improvement. *arXiv preprint arXiv:2409.12122*, 2024a. URL https://arxiv.org/abs/2409.12122.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a. URL https://arxiv.org/abs/2505.09388.

Ling Yang, Zhaochen Yu, Tianjun Zhang, Shiyi Cao, Minkai Xu, Wentao Zhang, Joseph E. Gonzalez, and Bin Cui. Buffer of Thoughts: Thought-Augmented Reasoning with Large Language Models. *arXiv preprint arXiv:2406.04271*, 2024b. URL https://arxiv.org/abs/2406.04271.

Xinyu Yang, Yuwei An, Hongyi Liu, Tianqi Chen, and Beidi Chen. Multiverse: Your language models secretly decide how to parallelize and merge generation. In *Advances in Neural Information Processing Systems*, volume 38, 2025b. doi: 10.52202/085713-3371. URL https://proceedings.nips.cc/paper_files/paper/2025/hash/91b1966914203338495a76d13b23599c-Abstract-Conference.html.

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. *arXiv preprint arXiv:2305.10601*, 2023. URL https://arxiv.org/abs/2305.10601.

Zihao Ye, Lequn Chen, Ruihang Lai, Wuwei Lin, Yineng Zhang, Stephanie Wang, Tianqi Chen, Baris Kasikci, Vinod Grover, Arvind Krishnamurthy, and Luis Ceze. FlashInfer: Efficient and Customizable Attention Engine for LLM Inference Serving. *arXiv preprint arXiv:2501.01005*, 2025. URL https://arxiv.org/abs/2501.01005.

Qiying Yu, Zheng Zhang, Ruofei Zhu, et al. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025. URL https://arxiv.org/abs/2503.14476.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-Rewarding Language Models. *arXiv preprint arXiv:2401.10020*, 2024. URL https://arxiv.org/abs/2401.10020.

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. URL https://arxiv.org/abs/2504.13837.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. STaR: Bootstrapping reasoning with reasoning. *arXiv preprint arXiv:2203.14465*, 2022. URL https://arxiv.org/abs/2203.14465v2. Version 2.

Wentao Zhang, Haoyu Zhang, Xinke Jiang, Yuxuan Cheng, Yuhan Pan, Miao Li, Zhipeng Qiao, Tao Feng, Zhen Tao, and Dengji Zhao. Fishing out free riders: Shapley-based reward attribution for parallel reasoning via reinforcement learning. *arXiv preprint arXiv:2607.18979*, 2026. URL https://arxiv.org/abs/2607.18979.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, Alban Desmaison, Can Balioglu, Pritam Damania, Bernard Nguyen, Geeta Chauhan, Yuchen Hao, Ajit Mathews, and Shen Li. PyTorch FSDP: Experiences on Scaling Fully Sharded Data Parallel. *arXiv preprint arXiv:2304.11277*, 2023. URL https://arxiv.org/abs/2304.11277.

Huaixiu Steven Zheng, Swaroop Mishra, Xinyun Chen, Heng-Tze Cheng, Ed H. Chi, Quoc V Le, and Denny Zhou. Take a Step Back: Evoking Reasoning via Abstraction in Large Language Models. *arXiv preprint arXiv:2310.06117*, 2023a. URL https://arxiv.org/abs/2310.06117.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. SGLang: Efficient execution of structured language model programs. *arXiv preprint arXiv:2312.07104*, 2023b. URL https://arxiv.org/abs/2312.07104.

Tong Zheng, Xidong Wu, Zheng Zhang, Zhankui He, Chaoyi Zhang, Benjamin Coleman, Ruoqiao Wei, Di Bai, Haolin Liu, Rui Liu, Xue Wang, Yue Zhuan, Wang-Cheng Kang, Renkai Xiang, Heng Huang, Xinwu Cheng, and Yunsong Guo. Dream-RSI: Recursive self-improvement through evolving worlds. *arXiv preprint arXiv:2609.14858*, 2026a. URL https://arxiv.org/abs/2609.14858.

Tong Zheng, Hongming Zhang, Wenhao Yu, Xiaoyang Wang, He Xing, Runpeng Dai, Rui Liu, Huiwen Bao, Chengsong Huang, Heng Huang, and Dong Yu. Parallel-R1: Towards parallel thinking via reinforcement learning. In *Proceedings of the 14th International Conference on Learning Representations*, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/c4f3846c9a230cfd3373ff419b202bd5-Abstract-Conference.html.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language Agent Tree Search Unifies Reasoning Acting and Planning in Language Models. *arXiv preprint arXiv:2310.04406*, 2023. URL https://arxiv.org/abs/2310.04406.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. *arXiv preprint arXiv:2205.10625*, 2022. URL https://arxiv.org/abs/2205.10625.

Pei Zhou, Jay Pujara, Xiang Ren, Xinyun Chen, Heng-Tze Cheng, Quoc V. Le, Ed H. Chi, Denny Zhou, Swaroop Mishra, and Huaixiu Steven Zheng. SELF-DISCOVER: Large language models self-compose reasoning structures. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-4004. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/e41efb03e20ca3c231940a3c6917ef6f-Abstract-Conference.html.

Yufan Zhuang, Chandan Singh, Liyuan Liu, Yelong Shen, Dinghuai Zhang, Jingbo Shang, Jianfeng Gao, and Weizhu Chen. Test-time recursive thinking: Self-improvement without external feedback. *arXiv preprint arXiv:2602.03094*, 2026. URL https://arxiv.org/abs/2602.03094.

A Implementation and Training

The implementation uses PyTorch and Transformers (Paszke et al., 2019; Wolf et al., 2020). RL training builds on verl (Sheng et al., 2024) with PyTorch FSDP (Zhao et al., 2023); its modified SGLang rollout path for Qwen3 uses FlashInfer attention kernels (Zheng et al., 2023b; Ye et al., 2025).

A.1 Supervised Initialization

The Qwen3 recipe starts from Qwen3-4B Base and adds eight delimiter tokens for guidelines, plans, steps and takeaways. The first stage freezes the transformer and trains the input-embedding and output-head matrices on 1,000 examples at learning rate 10^{-6}. The second trains all parameters on the 10,000-example corpus at 10^{-5}. Both stages use AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 10^{-8}$, weight decay 0.1 and gradient clipping at 1.0. Each stage runs for one epoch with cosine learning-rate decay and 5% warmup, using a 32,768-token context limit. Eight devices, one example per device and two accumulation steps give effective batch size 16, yielding 63 embedding updates and 625 full-model updates. Embedding adaptation followed by full-model training has also been used after vocabulary replacement (Samuel et al., 2025).

The SFT loss is ordinary cross-entropy averaged over supervised assistant targets under the branch attention masks and logical positions. Prompt targets are ignored; plans, branch steps, takeaways, structural tokens and end-of-sequence are supervised. During RL, structural tokens inserted by the runtime instead serve as context without a direct policy-gradient term.

A.2 Outcome-Based Training

Qwen3 RL uses 16 prompts per update with four first-attempt trajectories per prompt. All RL settings use learning rate 10^{-6} with a constant schedule, zero weight decay, no warmup, gradient clipping at 1.0, temperature 1 and top-$p = 1$. Ordinary RL retains invalid trajectories. The fixed reference policy is the SFT initialization. The KL coefficient is $\beta = 0.001$ and the normalization in Equation 4 is $C = 9983.19625$. The constant is fixed from initial visible-token statistics and rescales every trajectory equally. The reported Qwen3 RL model uses 60 updates; SIPR-RL-C uses 30 updates with 32 prompts and eight valid trajectories per prompt in its documented four-GPU recipe. The Qwen3.5 results use full-stage SFT at 625 updates and RL at 20 updates. Both families use the same 10,000 demonstrations and reasoning content, with model-specific tokenization. Qwen3.5 formatting preserves source branch identifiers in 9,981 examples. All 10,000 examples have the same reasoning text apart from these identifiers. Its SFT follows the same two-stage learning rates, effective batch size 16, one-epoch cosine schedule, 5% warmup; the first stage adapts its tied input/output embedding matrix. Both families use the same 17,917-problem DAPO-Math training file. Qwen3.5 RL samples eight trajectories per prompt, with eight prompts per update for updates 1–7 and four thereafter, learning rate 10^{-6}, temperature 1, top-$p = 1$, a 30,000-token response budget, $\beta = 0.001$ and the same fixed C. Invalid trajectories are retained. From update 8, Qwen3.5 also stops a trajectory before branching when the same normalized ordered plan list recurs; it retains the generated prefix without forcing a final synthesis.

For Qwen3 outcome training, success requires a complete, valid NPR structure, a valid global-answer region, a response within the length limit, and clean termination on the tokenizer's end-of-sequence token. The global-answer extractor first takes the last balanced boxed expression in the legal tail after the final takeaway, then the last boxed expression within that takeaway. If neither exists, extraction is restricted to the final takeaway, preferring an equality or explicit assertion for the question's target before a generic mathematical-answer fallback. The extracted answer is compared with the reference using the training implementation's mathematical-equivalence checker. A valid correct response receives $+1$; an incorrect response, a missing global answer, or a trajectory failing validity or termination checks receives -1. Thus a correct boxed candidate in an unfinished branch does not qualify as training success. Such trajectories remain in the first-attempt training group. A grading timeout raises an error rather than being converted into an incorrect-answer reward. This differs from reported answer-content evaluation, which can credit a boxed candidate in an incomplete response.

The RL optimizer also applies a transactional update check, distinct from the reference-policy KL reward. After computing one full-batch gradient, it accepts a candidate optimizer step only if numerical checks pass and the fixed-C old-to-new k_3 KL estimate is at most 0.5. Otherwise it restores the previous state, halves the learning rate, and retries the saved gradient, for at most two retries; exhaustion pauses training. At most one candidate is committed per collected batch.

All sampled actions in a trajectory share its advantage. This jointly trains planning, branch reasoning and synthesis, but supplies no separate credit for a sound branch and an erroneous sibling. Correctness alone does not determine the advantage's sign because the group baseline and reference-policy term

also enter it. A constant C rescales the raw gradient; normalizing each trajectory by its own length would instead change the objective's relative trajectory weights.

A.3 Native Parallel Execution

If g_r is the closing guideline's logical position and $\ell_{r,i}$ the length of step span i, our implementation assigns logical positions with shared branch origins (Wu et al., 2026):

$$\mathrm{pos}(s_{r,i,j}) = g_r + 1 + j, \quad j = 0, \ldots, \ell_{r,i} - 1, \tag{5}$$

$$\mathrm{pos}(u_{r,0}) = g_r + 1 + \max_i \ell_{r,i}. \tag{6}$$

Logical positions enter rotary position embeddings (Su et al., 2021); sibling steps share an origin while occupying distinct physical token spans. Their attention follows Equation 1; synthesis can read all completed branch steps. Serial control length c_r and branch lengths give serialized size $c_r + \sum_i \ell_{r,i}$ and ideal dependency depth $c_r + \max_i \ell_{r,i}$. These quantities do not determine wall-clock time, which also depends on batching, attention work, cache management and synchronization.

The implementation uses the current zero-based logical index for decode positions; execution records preserve each sampled action and its predictor context, as described in Section 3.2. The ledger separates actual sampled actions from inserted environment tokens, retaining predictor contexts even when the returned text has incomplete structure. This prevents serialized branch order from becoming an unintended conditioning assumption during policy scoring. Sampling retains truncated branch content under a shared response-length budget.

For Qwen3.5, full-attention layers use the visibility mask in Figure 2, while linear-attention layers replay convolution and recurrent state over each branch's explicit visible context. The logical positions are shared across its three text RoPE axes; a dense attention mask alone does not implement the recurrent-state restriction.

B Constructing Exploration Demonstrations

B.1 Information Flow and Generation Instructions

Algorithm 1 summarizes the construction program. The selector receives the problem and strategy catalog, chooses applicable directives, and describes each route's setup, key transformation and target reduction. Proposals judged to use the same key transformation are merged.

A branch call receives the problem, difficulty, its directive and brief plan, completed shared history, and any pending steps. It produces an ordered sequence of concrete plans but executes only its next step. That step should establish a value, identity, bound or resolved case. Explicit subtask assignment and follow-up questions expose intermediate reasoning work (Khot et al., 2022; Press et al., 2022); here the assigned unit is the next pending branch step. The prompt asks the branch to check the original problem, avoid repeating completed work, and reserve boxed answers for the original question. Current-round outputs are buffered outside shared history, including when calls execute sequentially.

The intermediate synthesizer reads the frozen history and all accepted current-round outputs. It records progress, candidate answers and unresolved gaps, withholding the final boxed answer until final synthesis. Candidate-argument comparison also appears in multiagent debate (Du et al., 2023; Liang et al., 2023); here synthesis reads completed branch work under the specified information flow. It also assigns private novelty ratings that may retire redundant routes while preserving completed work. At least one directive remains active if all would otherwise retire. After worker updates, the terminal accepted round uses final synthesis when no route remains active or the round cap is reached. It reads completed history and current steps and requests a boxed answer. A route finishes on a boxed candidate or an empty pending queue; failed parse retries may exhaust it. Rounds without accepted steps are omitted. There is no unconditional synthesis call after the loop. Worker and synthesis calls use temperature 1, top-$p = 0.7$ and no top-k truncation, with respective output limits of 2,048 and 1,024 tokens. The generator allows four worker attempts and three synthesis attempts. Worker duplicate rejection is disabled; synthesizer judgments retire redundant routes while retaining at least one active route. The round cap is $\max(12, \min(16, 12 + \lfloor d/2 \rfloor))$ for difficulty d, with an optional `MAX_ROUNDS` override.

Algorithm 1 Constructing Exploration Demonstrations

Require: Problem x, strategy catalog $\mathcal{L}$, generator q, cap R
Select applicable directives judged to have distinct key transformations
Initialize history H, active branches I and local pending queues
for $r = 1, \ldots, R$ **do**
  Freeze $H_r \leftarrow H$; clear accepted plans P_r and steps S_r
  for each active branch $i \in I$, in sequence **do**
    Query q with x, H_r, directive L_i and local metadata
    Parse/retry; buffer one accepted plan and executed step
    Update the branch's pending queue and completion status
  end for
  Update active routes I from worker completion and retry status
  if no step was accepted **then**
    if $I = \varnothing$ or $r = R$ **then**
      break
    else
      continue
    end if
  end if
  $f \leftarrow (I = \varnothing \text{ or } r = R)$
  if f **then**
    Final-synthesize u_r from H_r, P_r, S_r, requesting a boxed answer
  else
    Intermediate-synthesize u_r and private route ratings
    Retire redundant routes, keeping one if all active routes would retire
  end if
  $H \leftarrow H_r \parallel (P_r, S_r, u_r)$
  if f **then**
    break
  end if
end for
Select structurally eligible traces with an accepted correct box in the stored final takeaway
Retain visible plans, steps and takeaways; omit private metadata

B.2 Strategy Catalog

The catalog contains 49 directives in 13 families (Table 5). Each entry specifies a name, definition, illustrative first move, applicability triggers and distinguishing features. The selector assigns applicability scores from 0 to 3 and requires at least 2, an appropriate first move and an account of the structure it would reveal. It compares two selection passes, retains agreed choices, and merges proposals with the same key transformation. Selection is capped at ten directives overall and three per family; the supplemental selection pass adds at most four directives.

Table 5: **Mathematical strategy families.** Counts denote catalog entries, not their frequency in the training corpus.

Family	Entries	Family	Entries
Specialization	6	Generalization	4
Reversal	4	Re-representation	5
Decomposition	3	Analogy	4
Auxiliary construction	4	Constraint modification	3
Invariants	3	Existence	3
Estimation	3	Perspective	3
Metacognition	4	Total	49

We interpret distinct mathematical work as a different transformation, lemma or subcase yielding a checkable result. Copying a candidate or repeating one calculation does not meet this criterion. A

later derivation may corroborate an answer already in history without being an independent discovery. The qualitative cases apply this distinction; no corpus-wide semantic-diversity rate is reported.

B.3 Filtering and Corpus Composition

The 10,000-example initialization corpus uses Polaris mathematics (POLARIS Project, 2025). Synthetic mathematics corpora make question diversity and solution format explicit data-design choices (Toshniwal et al., 2024b;a; Yang et al., 2024a); our construction adds branch-specific strategies and shared intermediate history. Selection targets three to six first-round plans, applies example deduplication and a 25,000-token cutoff, requires a correct boxed answer in the stored final takeaway and retains at least two surviving directives. Routes exhausted after failed retries are removed; remaining headers are renumbered. Intermediate candidates in surviving branches remain in the trace. Private strategy definitions, selector plans, queues and retirement metadata are removed from supervision.

RL prompt pool. The RL pool contains 17,917 problems and their reference answers from DAPO-Math-17k (Yu et al., 2025). Both model families use this training file for outcome-based learning.

C Evaluation Protocols

C.1 Benchmarks, Models and Scoring

The primary datasets contain 500 MATH-500, 40 AMC23, 30 AIME24, 30 AIME25, 30 HMMT February 2025, 272 Minerva and 674 English text-only OlympiadBench problems. AMC, AIME and HMMT use eight responses per problem. Mean accuracy estimates individual-response correctness; Pass@8 is the fraction of questions with at least one correct response among eight. The Qwen3 Base MATH-500 estimate uses 1,000 responses, and Qwen3 SFT/RL Minerva estimates use 544, including the completion variant. Other non-contest main-table entries use one response per problem. Each reported score retains the full denominator, including invalid, truncated and unresolved responses. Avg. equally weights the seven mathematical benchmarks. GPQA Diamond contains 198 problems and is reported separately.

Base uses the unmodified pretrained model and tokenizer, four worked mathematical examples, serial generation and no chat template. SFT/RL use the parallel-reasoning prompt, chat template and parallel decoding with at most nine plans per round. The prompt appends "Let's think in parallel." to the question. Qwen3 Base generation uses vLLM (Kwon et al., 2023); native parallel models use modified SGLang (Zheng et al., 2023b).

Base uses automatic answer extraction and reference comparison. For Qwen3 SFT/RL, evaluation extracts the last complete boxed candidate anywhere in the response, including branch bodies, and checks it against the original question and benchmark reference. Identical dataset/candidate/reference combinations reuse the same correctness decision; contextual cases retain the question and saved response. Structural incompleteness does not by itself invalidate a mathematically correct candidate. Qwen3.5 SFT/RL evaluation checks the retained answer to the requested target, including explicit unboxed conclusions. A later withdrawal or replacement takes precedence over an earlier candidate. SFT and RL use the same scoring rule within each model family. The accompanying correctness records retain the question, candidate and reference used for each decision.

Qwen3.5 SFT and RL evaluation apply the same repeated-plan stopping rule as its resumed training. Plan-list comparison preserves order and normalizes Unicode, leading branch numbers, case and whitespace. Qwen3 evaluation does not use this additional stop rule. Two Qwen3.5 Base AIME25 extraction failures are resolved by exact-integer comparison. Base GPQA uses terminal-choice extraction; 40 Qwen3 and 18 Qwen3.5 unextracted choices count as incorrect. Unresolved answer checks receive no correct credit and remain in the denominator. Pass credits only questions with at least one confirmed correct response. Contest evaluation uses temperature 1, top-$p = 0.7$, no top-k truncation, eight responses per problem and a 30,000-token response limit. Scores are reported to two decimal places; gains subtract the displayed scores. The accompanying evaluation manifest identifies the response sets used for each reported cell; the paired bootstrap recomputes the same reported statistic from these records.

C.2 Published Baselines

We use the ICLR 2026 proceedings version of Parallel-R1, Section 4.1 and Table 2, rather than an earlier arXiv table (Zheng et al., 2026b). Its MATH benchmark removes problems containing LaTeX from MATH-500; the retained size is not stated. Generation uses temperature 1.0 and a 3,000-token response limit, with one response per MATH problem and 16 per AIME/AMC problem. "Seen" is the autoregressive variant without explicit branch isolation; "Unseen" uses the multiverse architecture, with accuracy-only (S1) or alternating accuracy/parallel rewards (S2). These names do not denote our training/test split.

For SSFT we use the MATH-500 result in the Qwen3-4B-Base learned-matching row of Table 6 in arXiv version 3 (Jia et al., 2026), not the larger-model or voting results. The source row reports MATH-500 81.85%, AIME24 23.44% and AIME25 23.13%; Table 3 uses its MATH-500 entry. Its reported Pass@1 estimates individual-generation accuracy by averaging 32 generations; the general evaluation setup uses temperature 0.7, top-$p = 0.95$ and a 32,768-token maximum length. For this model, the prompted forking token is selected by its degree in the learned matching graph. Different subsets, generation budgets, prompts and scorers make these author-reported comparisons descriptive rather than controlled rankings.

D Learning and Trajectory Analyses

Paired uncertainty. We draw 10,000 bootstrap resamples with seed 20260925, pairing policies by question and retaining all responses for each sampled question. Table 6 reports percentile intervals for the same accuracy statistics as the main table. Each policy's macro-average weights the seven mathematical benchmarks equally: the difference between the reported macro-averages is $+2.31$ percentage points with a question-paired 95% interval of $[+0.52, +4.02]$. This interval conditions on the evaluated policies, saved responses and grading decisions. Intervals are reported per benchmark without multiple-comparison adjustment.

Table 6: **Accuracy gains from SFT to RL.** Entries give the SIPR-RL minus SIPR-SFT difference in mean accuracy (percentage points) with 95% question-paired bootstrap intervals. The first seven rows use the main-table accuracy estimates; the final row gives an additional estimate using 1,000 responses per policy.

Benchmark	Δ accuracy [95% interval]
MATH-500	+2.40 [-1.00, +5.00]
AMC23	+2.19 [-5.31, +8.75]
AIME24	+2.92 [-0.83, +7.50]
AIME25	+4.58 [+0.00, +10.42]
HMMT	+0.83 [-3.33, +5.42]
Minerva	+1.66 [-2.57, +5.88]
OlympiadBench	+1.63 [-1.78, +4.90]
MATH-500 (1,000 responses per policy)	+1.50 [-0.90, +3.90]

Planning and completion. On the 100 AMC/AIME questions with eight responses per policy, responses without repeated plans increase from 70.38% to 76.63% ($+6.25$ pp; interval $[+2.25, +10.25]$). A repeated plan is an exact within-round match after deleting its numeric prefix, lowercasing and collapsing whitespace; symbols and punctuation remain. Responses containing a boxed expression increase from 84.25% to 88.75% ($+4.50$ pp; $[+1.38, +7.75]$). This measures answer-candidate presence without asserting correctness or complete reasoning. These intervals use 20,000 question-paired resamples within each dataset, preserving dataset weights. The diagnostics use all 1,600 responses, each bound to its frozen grade by a full-output hash.

Matched mathematical checks. For each of AMC23, AIME24 and AIME25 and each correctness transition, we inspect the lowest question identifier and sample index, yielding 12 matched pairs. This balances qualitative outcomes rather than estimating their prevalence. In the walking-time example, the correct constraints are $9/s + t/60 = 4$ and $9/(s+2) + t/60 = 2.4$. They give $s = 2.5$, $t = 24$,

Table 7: **Exploration behavior on matched AMC/AIME questions.** Each policy contributes 800 responses. Changes are RL–SFT in percentage points with question-paired 95% bootstrap intervals. Both measures are higher-is-better. Repetition means normalized exact-text plan repetition within a round; a boxed expression supplies a candidate, not a correctness certificate.

Observable	SIPR-SFT	SIPR-RL	Δ [95% interval]
No repeated within-round plans	70.38	76.63	+6.25 [+2.25, +10.25]
Contains a boxed expression	84.25	88.75	+4.50 [+1.38, +7.75]

and $60 \cdot 9/(s + 0.5) + t = 204$. RL first proposes $t = 240$, then explicitly rejects it when it makes the second walking duration negative; the later takeaway adopts the correct branch solution. The SFT trace contains that intermediate solution but does not complete its synthesis. In the base-conversion example, $b + 7 \mid 9b + 7$ reduces to $b + 7 \mid 56$, giving $b = 21, 49$ and sum 70. RL checks both in the original condition, whereas SFT propagates $10b + 7$ as the value of 17_b. A regression on AMC23 replaces $(2k)^3 - (2k - 1)^3$ with $(2k + 1)^3 - (2k)^3$: SFT returns the correct 3159, while RL returns 3699. Full paired traces, literal excerpts and independent arithmetic checks are retained in the evidence bundle.

E Additional Results

E.1 Accuracy Gains

Figure 3 plots the seven mathematical accuracy differences in Table 1 with paired intervals.

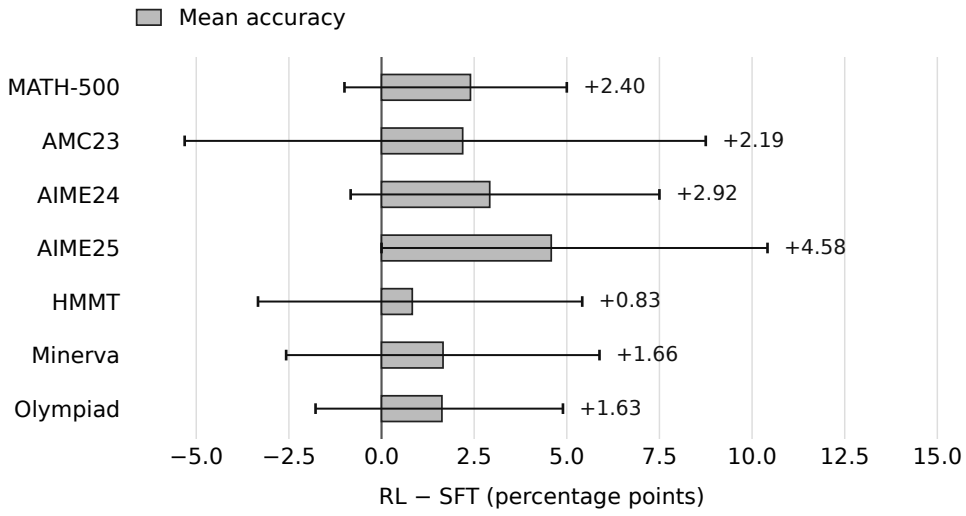

Figure 3: **Accuracy changes from SFT to RL in Qwen3-4B.** The points repeat the seven mathematical accuracy differences in Table 1. Error bars show 95% paired bootstrap intervals, resampling questions while retaining their responses (Appendix D).

E.2 Training Dynamics

Figure 4 shows increasing mean reward over three consecutive training intervals for ordinary SIPR-RL: -0.470, -0.411 and -0.368. Invalid trajectories receive -1 and are included in every batch, so the reward reflects validity and answer correctness jointly. These means are computed on changing training prompts up to the evaluated policy, not on a held-out evaluation set.

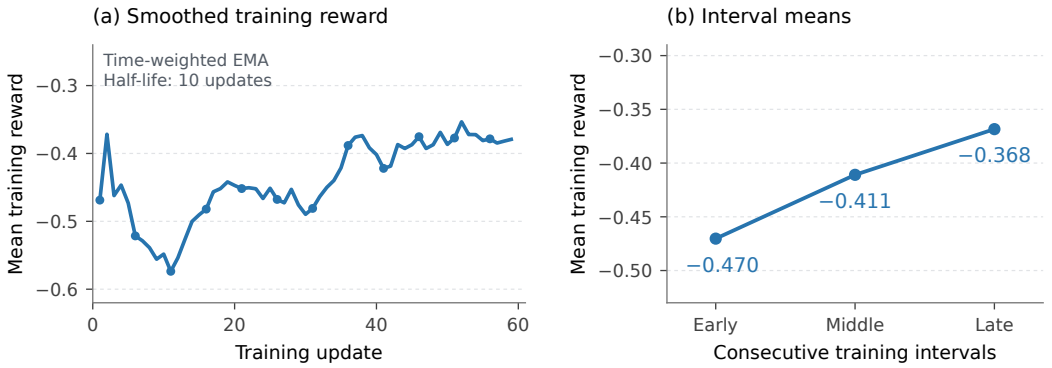

Figure 4: **Training reward of ordinary Qwen3-4B SIPR-RL.** Left: time-weighted EMA of all 59 logged rewards, with a half-life of 10 training updates and initialization debiasing. Right: interval means for updates 1–20, 21–40 and 41–60; the three intervals contain 20, 20 and 19 recorded batches, respectively. Invalid trajectories remain in the average: valid correct responses receive $+1$ and all others -1. The means summarize changing training prompts, without uncertainty estimates. Individual batch rewards range from -0.75 to $+0.03$. Higher is better.

E.3 Completion and Inference Resampling

Original-response coverage. Pass@8 is the fraction of questions with at least one correct original response among eight. It measures coverage rather than the accuracy of an individual response; gains at small sampling budgets can coexist with lower coverage at larger budgets (Yue et al., 2025). In Qwen3, ordinary RL increases AIME25 mean accuracy while Pass@8 changes from 30.00% to 26.67%; the completion-trained model reaches 95.00% on AMC23 and 36.67% on AIME24.

Table 4 in the main text reports the original and resampled variants; Table 8 below reports their problem coverage.

Table 8: **Problem coverage after resampling and rescue.** Values give the percentage of problems with at least one credited correct target slot. The second header row gives target slots per problem, not total generation attempts. R applies offline resampling and reference-answer-assisted rescue to the same checkpoint. Colored superscripts give descriptive RL–SFT changes and bold marks the higher scores; generation counts can differ across models.

Model	MATH-500 Slots: 1	AMC23 Slots: 8	AIME24 Slots: 8	AIME25 Slots: 8	HMMT Slots: 8	Minerva Slots: 1	Olymp. Slots: 1
SIPR-SFT-R	81.00	85.00	36.67	**36.67**	**23.33**	**53.31**	49.55
SIPR-RL-R	**84.60**$^{+3.60}$	**90.00**$^{+5.00}$	**40.00**$^{+3.33}$	30.00$^{-6.67}$	**23.33**$^{+0.00}$	52.57$^{-0.74}$	**51.48**$^{+1.93}$

SIPR-RL-C discards invalid training trajectories until each prompt has eight valid ones; valid incorrect responses remain eligible. Its larger batch and different training duration distinguish it from the separately trained SIPR-RL run, so it is not an isolated validity-filter ablation.

Inference resampling preserves structurally valid, normally terminated responses and retries invalid positions for up to three additional rounds, keeping fixed target denominators. Two eight-response problems whose initial responses were all invalid were skipped without retries; their 16 slots remain wrong. Unfilled positions count as incorrect. Reference-aware historical rescue checks earlier answers against the reference to credit still-empty slots, including from structurally invalid responses, with each earlier answer used at most once. This offline answer selection requires correctness information unavailable to ordinary deployment. Credited-slot accuracy includes both additional generation and this answer-selection rule.

F Qualitative Reasoning Cases

The four responses below come from the checkpoint after 40 updates of the same Qwen3 RL run. They illustrate cross-checking, inconsistent repair and continued generation after a correct answer. They are distinct from the final-policy SFT/RL pairs discussed in Section 4.3, and are not a frequency estimate. The first panel typesets selected mathematical excerpts for readability; subsequent indented excerpts reproduce saved model text, including mistakes. Problem summaries, headings and analysis are author annotations. The evidence bundle retains raw responses, excerpt offsets and independent arithmetic checks.

F.1 Checking a Candidate through Different Relations

Table 9. Cross-checking an existing candidate (intermediate RL policy)

Problem. Alicia and Beth start 45 miles apart and bike toward each other at 18 and 12 mph. How far from Alicia's starting city do they meet? The reference answer is 27 miles.

Round 1 takeaway. "... Angle 3 sets up the time-equality equation $\frac{d}{18} = \frac{45-d}{12}$, which will resolve to $d = 27$ miles."

Round 2: checking the candidate. *Equations typeset from the saved branches; connecting prose omitted. Italic headings are author labels.*

Ratio of speeds, $18 : 12 = 3 : 2$.

$$45 \div 5 = 9, \qquad 3 \times 9 = 27.$$

Time until meeting.

$$18t + 12t = 45, \qquad 30t = 45, \qquad t = \frac{45}{30} = 1.5.$$

Equality of travel times.

$$\frac{d}{18} = \frac{45 - d}{12}, \qquad 30d = 810, \qquad d = 27.$$

Synthesis. "Angle 1 confirms the distance via ratio, Angle 2 solves the combined distance equation to find $t = 1.5$, and Angle 3 verifies the time-equality equation and derives $d = 27$."

Analysis. This round checks an answer already available in history. The elapsed-time calculation also recurs in an omitted branch, so multiple plans do not ensure distinct work.

AMC23, question 0, sample 0 (zero-based). Round-2 branches 1–3 shown; branch 5 omitted; no branch 4. "Angle k" denotes branch k.

F.2 Local Repair and Inconsistent Synthesis

Table 10. Two responses to the same base-arithmetic problem (intermediate RL policy)

Problem and reference

Find the sum of all integer bases $b > 9$ for which 17_b divides 97_b. The reference answer is 70.

Context and independent check

AIME 2025, question 0, two samples from the same intermediate RL checkpoint and prompt. Since $17_b = b + 7$ and $97_b = 9b + 7 = 9(b + 7) - 56$, the admissible divisors give $b = 21, 49$ and $21 + 49 = 70$. The following samples contrast two outcomes of the same policy; they are not an SFT/RL comparison.

Sample 0: response budget exhausted

30,000 returned tokens; two rounds started and one completed takeaway.

```
[... 1,156 characters omitted ...]
Therefore, $ b + 7 \mid 56 $.
[... 3,015 characters omitted ...]
For $ b = 21 $:
[... 35 characters omitted ...]
$ 97_b = 9 \cdot 21 + 7 = 189 $.
[... 58,120 characters omitted ...]
```

Analysis. The product $9 \cdot 21 = 189$ still needs the added 7, giving 196. Round-2 branch 3 repeatedly returns to this error and remains open, using 28,129 body tokens. No second takeaway is produced.

Sample 1: evaluator success with inconsistent intermediate reasoning

6,245 returned tokens; four completed rounds.

```
[... 4,438 characters omitted ...]
\boxed{70}
[... 9,484 characters omitted ...]
The polynomial division confirms a constant remainder of 7,
   reinforcing the equivalence of the divisibility condition to $b
   + 7 \mid 56$.
[... 1,266 characters omitted ...]
The final answer is \boxed{70}.
[... 22 characters omitted ...]
```

Analysis. The first boxed 70 appears in round 2; generation continues for two more rounds. An incorrect polynomial representation yields a spurious remainder 7, which persists in the round-3 takeaway before the final correction. A correct final answer does not make every intermediate step correct.

F.3 A CORRECT ANSWER WITHOUT TERMINATION

Table 11. Finding the answer but continuing to generate (intermediate RL policy)

Problem relation and reference

For AIME 2025 question 16, reducing $3(n+3)(n^2+9)$ modulo $n+2$ gives 39. Its admissible divisors imply $n = 1, 11, 37$, with reference sum 49.

Context

The response first boxes 49 in round 2, then continues through 29 guidelines and 28 completed takeaways. One takeaway occurs 25 times. The final branch remains open when the 30,000-token response budget is exhausted.

Literal evidence

The first excerpt gives the boxed answer. The next gives the round-3 takeaway's remaining gap. The last line appears 204 times exactly, including trailing spaces, in round 29; omission markers preserve the locations in the complete response.

```
[... 8,538 characters omitted ...]
\boxed{49}
[... 5,235 characters omitted ...]
The only remaining gap is to formally confirm that no other
    positive $ n $ could satisfy the divisibility via alternative
    factorizations or substitutions, though this is already covered
    by the exhaustive check of divisors of 39.
[... 35,881 characters omitted ...]
Wait: $ 33 \times 909 = 33 \times (900 + 9) = 29700 + 297 = 30000 -
    3 = 29997 $?
[... 16,874 characters omitted ...]
```

Analysis

The arithmetic in the repeated line is correct, but the repetition accompanies failure to terminate. This case separates finding a correct candidate from deciding whether further exploration is useful.

G REPRODUCIBILITY MATERIALS

The accompanying materials provide the training and generation configurations, scored responses, integer metric counts, correctness records, and complete qualitative traces. Scripts regenerate the tables and figures from these records and verify response identities, metric arithmetic and quoted excerpts. The materials also include the strategy catalog and the code for constructing demonstrations and executing native parallel reasoning.