

Reinforcement Learning for Tool-Integrated Interleaved Thinking towards Cross-Domain Generalization

Anonymous ACL submission

Abstract

Recent advances in large language models (LLMs) have demonstrated remarkable capabilities in reasoning and tool utilization. However, the generalization of tool-augmented reinforcement learning (RL) across diverse domains remains a significant challenge. Standard paradigms often treat tool usage as a linear or isolated event, which becomes brittle when transferring skills from restricted domains (e.g., mathematics) to open-ended tasks. In this work, we investigate the cross-domain generalization of an LLM agent trained exclusively on mathematical problem-solving. To facilitate robust skill transfer, we propose a **Reinforcement Learning for Interleaved Tool Execution (RITE)**. Unlike traditional methods, RITE enforces a continuous “Plan-Action-Reflection” cycle, allowing the model to ground its reasoning in intermediate tool outputs and self-correct during long-horizon tasks. To effectively train this complex interleaved policy, we introduce Dr. GRPO, a robust optimization objective that utilizes token-level loss aggregation with importance sampling to mitigate reward sparsity and high-variance credit assignment. Furthermore, we employ a dual-component reward system and dynamic curriculum via online rollout filtering to ensure structural integrity and sample efficiency. Extensive experiments reveal that our approach, despite being trained solely on math tasks, achieves state-of-the-art performance across diverse reasoning domains, demonstrating high token efficiency and strong generalization capabilities.

1 Introduction

Large language models (LLMs) have achieved impressive performance in a wide range of reasoning and problem-solving tasks, especially when augmented with external tools such as code interpreters, calculators, and knowledge bases (DeepSeek-AI Team, 2025; Team, 2025a; Team et al., 2025). These tool-augmented agents are ca-

pable of autonomously planning, invoking tools, and solving complex tasks that require multi-step reasoning (Gao et al., 2025a; Luo et al., 2025a; Plaat et al., 2025). While prior studies have focused on supervised fine-tuning (SFT) and reinforcement learning (RL) to enhance tool usage, most approaches rely on domain-specific training or treat tool execution as a disjoint step from reasoning, limiting the agent’s ability to generalize to unseen scenarios.

In this paper, we address a fundamental question: *Can an LLM agent trained to use a code interpreter solely on mathematical problems generalize its reasoning and tool usage strategies to diverse, unseen domains?* To answer this question, we focus on a challenging setting where the agent learns tool invocation strategies via RL in the strict logic of the math domain and is subsequently evaluated on distinct, open-ended tasks. This setup enables us to systematically analyze cross-domain generalization and skill transfer. We observe that standard RL objectives (e.g., PPO or vanilla GRPO) often struggle in this context due to the *linear* nature of traditional tool use and the severe reward sparsity inherent in long-horizon reasoning.

To overcome these limitations and enhance cross-domain transfer, we propose the **Reinforcement Learning for Interleaved Tool Execution (RITE)** framework. Our approach is built on the insight that robust generalization stems from the reasoning structure rather than domain-specific knowledge.

Learn to use tools rather than specific domain knowledge can truly enhance cross-domain generalization.

Specifically, our proposed RITE framework comprises three key technical contributions designed to stabilize training and foster domain-agnostic learning. *Tool-Integrated Interleaved Thinking*: We move beyond the separation of reasoning and execution by enforcing a “Plan-Action-Reflection”

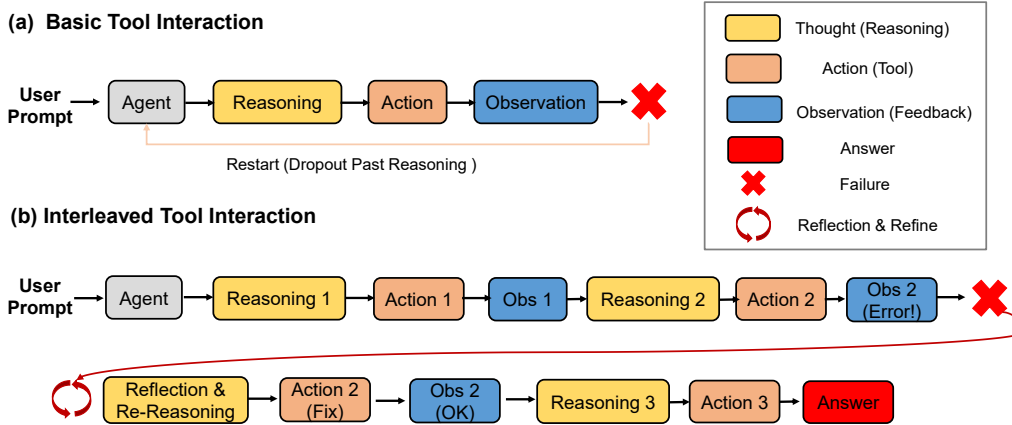


Figure 1: Interaction comparison between standard tool RL interactions (a) and our Interleaved Thinking process (b). While standard methods often discard intermediate states, our framework forms a continuous “Plan-Action-Reflection” loop, preserving reasoning history for error correction and robust generalization.

081 cycle as shown in Figure 1. This alternating pattern
 082 ensures that the agent grounds every step in the
 083 latest tool observations, significantly reducing hal-
 084 lucination and error propagation when transferring
 085 to complex tasks.

086 *Token-Level Optimization:* Training long-
 087 context interleaved policies is unstable with
 088 trajectory-level rewards alone. We introduce Dr.
 089 GRPO, which incorporates token-level loss aggre-
 090 gation using Importance Sampling (IS) and Rejec-
 091 tion Sampling (RS). This allows gradients to focus
 092 on critical decision points (e.g., tool invocation and
 093 reasoning transitions) rather than being diluted by
 094 long reasoning chains.

095 *Robust Training Curriculum:* To further align
 096 the agent with the interleaved structure, we im-
 097 plement a *Dual-Component Reward* system (com-
 098 bining outcome correctness with structural format-
 099 ting rewards) and a *Dynamic Difficulty Adjustment*
 100 strategy that filters training samples based on the
 101 agent’s pass rate, ensuring continuous learning
 102 within the Zone of Proximal Development.

103 Our contributions can be summarized as follows:
 104 First, we demonstrate that RL-based tool usage
 105 learned from mathematical tasks can be effectively
 106 transferred to diverse domains when the reasoning
 107 process is structured correctly. Second, we intro-
 108 duce the Reinforcement Learning for Interleaved
 109 Tool Execution (RITE) framework with compo-
 110 nents specifically designed to enhance RL stability
 111 and cross-domain generalizability. Third, through
 112 extensive evaluations across multiple domains us-
 113 ing diverse benchmarks, we achieve state-of-the-art
 114 performance, providing insights into the key factors

driving successful skill migration and highlighting
 the transformative potential of Tool RL for LLM
 reasoning in cross-domain settings.

2 Related Work

2.1 Tool-Integrated RL for LLM Reasoning

Enhancing LLM reasoning with tools has achieved
 great attention due to its potential for expanding
 the knowledge boundary of models’ generative sup-
 port, thereby breaking the “invisible leash” that
 constrains purely text-based models (Lin and Xu,
 2025). The code interpreter and search engine, as
 two representative tools, have been widely inte-
 grated into the natural-language reasoning process
 to individually boost the model’s performance on
 mathematical (Feng et al., 2025; Li et al., 2025a)
 and knowledge-intensive reasoning tasks (Jin et al.,
 2025; Liu et al., 2025b; Team, 2025b). Advanced
 techniques focus on addressing new challenges un-
 der more complex scenarios(e.g.multi-turn) from
 different perspectives, such as long-horizon plan-
 ning (Gao et al., 2025b; Singh et al., 2025; Erdogan
 et al., 2025), memory management (Xu et al., 2025;
 Yan et al., 2025; Zhou et al., 2025d), interaction
 efficiency (Wang et al., 2025; Song et al., 2025),
 multi-tool selection (Dong et al., 2025; Zou et al.,
 2025) and interleaved thinking structure (Xie et al.,
 2025b; MiniMax, 2025; Liu et al., 2025a). How-
 ever, neither of these works explores the general-
 izability of tool-integrated RL approaches across
 diverse reasoning tasks and domains.

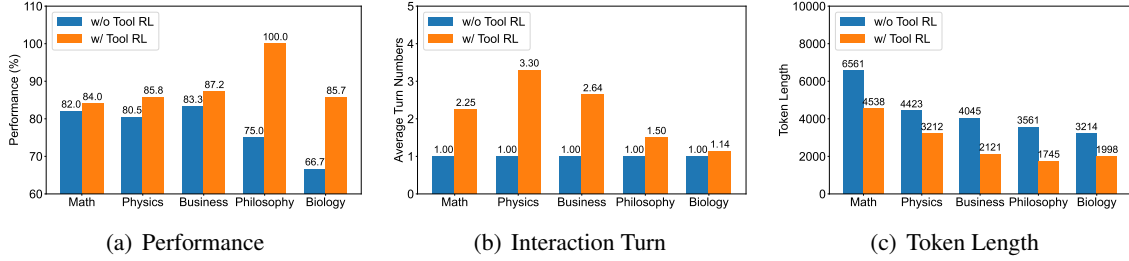


Figure 2: Cross-domain comparisons including model performance, interaction turns, and token length for the output on Webinstruct, where we perform tool RL training on Qwen2.5-7B using the code-integrated math dataset.

2.2 Cross-Domain Reasoning for LLM

Beyond mathematical and code domains, many efforts have been devoted to enhancing the cross-domain reasoning ability for LLMs (Cheng et al., 2025; Li et al., 2025b). Some pioneering works focus on how to curate high-quality cross-domain datasets (Akter et al., 2025; Ma et al., 2025a), including both synthetic and real-world questions into RL training to improve generalization across diverse reasoning tasks. Moreover, compared with calculating reward solely depending on the model-based verifier (Xie et al., 2025a), the scaling and reliable reward signal from models’ intrinsic mechanism have also been explored to broaden the reasoning boundary to general domains, assisted by Verifier-Free frameworks (Zhou et al., 2025b; Yu et al., 2025; Liu et al., 2025c) and Rubrics (Su et al., 2025; Gunjal et al., 2025; Huang et al., 2025; Zhou et al., 2025c). The most similar to our topics are those that focus on the transferability of LLM reasoning (Huan et al., 2025; Zhou et al., 2025a; Hu et al., 2025; Cheng et al., 2025), but these works only clarify the generalization effect through single-domain training without tool invocation. Our work first systematically explores the cross-domain reasoning potential of Tool RL training.

3 Uncover the Cross-Domain Generalization Brought by Tool RL

3.1 Problem Formulation

Our goal is to explore whether an LLM agent can generalize its tool-integrated reasoning performance on general domains, despite training exclusively on code-integrated math data \mathcal{D}_{CI} . Ideally, the ultimate target is to maximize expected performance on general domain queries as follows:

$$\max_{\theta} \mathbb{E}_{x \sim \mathcal{D}_{Gen}} [\mathbb{E}_{y \sim \pi_{\theta}(\cdot | x, \text{tool})} [R(y, x)]] \quad (1)$$

where \mathcal{D}_{Gen} represents the general domain query distribution and $R(y, x)$ is the reward function evaluating response y for query x .

3.2 Bridging the Domain Gap via Tools

From the theoretical perspective, we advocate that we can bridge the domain gap through tool-necessity optimization. By emphasizing high tool-necessity samples in \mathcal{D}_{CI} , we force the model to learn generalizable tool-usage reasoning patterns rather than domain-specific superficial features. The tool-necessity acts as a domain-invariant indicator of when tools are truly beneficial, enabling effective transfer to general domains. Thus, the formalization of the generalization objective to general domains is:

$$\max_{\theta} \mathbb{E}_{x \sim \mathcal{D}_{Gen}} [\mathbb{I}[\text{tool-needed}(x)] \cdot R(\pi_{\theta}(x, \text{tool}), x)] \quad (2)$$

where $\mathbb{I}[\text{tool-needed}(x)]$ indicates queries in general domains that require tool assistance.

This approach ensures that even when trained on specialized math data, the model develops robust tool-integrated reasoning capabilities that generalize broadly beyond the training distribution.

3.3 Analysis of Cross-Domain Experiments

We conduct exploration experiments to understand the generalization of tool-integrated RL approaches across diverse reasoning domains. As shown in Fig. 2, we perform the tool RL training of the Qwen2.5-7B on the code-integrated math data (as stated by Sec.5) and compare the model performance, the interaction turn numbers, and the token length of response across 5 different domains (Mathematics, Physics, Business, Philosophy, and Biology) on the Webinstruct (Ma et al., 2025b) evaluation dataset.

We can observe that: (i) Comparing with and without tool RL training, integrating the tool call into the natural reasoning process can enhance model performance with more interaction turns

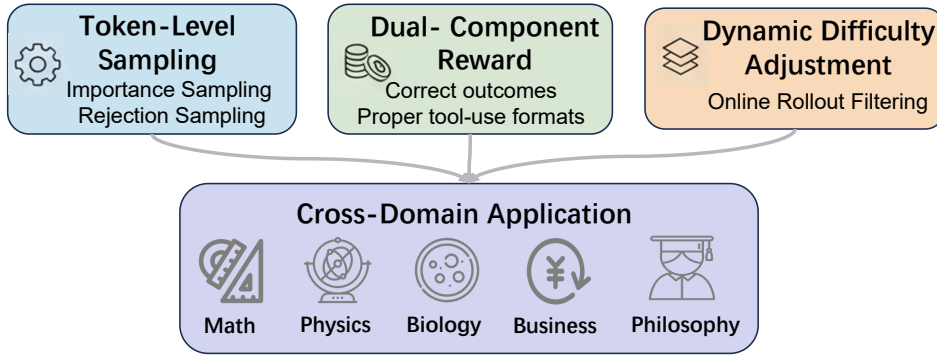


Figure 3: Illustrations of our proposed Reinforcement Learning for Interleaved Tool Execution (RITE) framework.

and less token length; (ii) Despite training only on the code-integrated math data, the model performance and the number of interaction turns and token length for the general reasoning (out of domain, such as physics, biology and so on) output have the consistent trend with in-domain math dataset. This can verify that there exists some tool use patterns (orthogonal to domain knowledge) transferred from the code-integrated math domain to the general reasoning domain, which can enhance the cross-domain reasoning abilities during RL training.

4 Reinforcement Learning for Interleaved Tool Execution

4.1 Tool-Integrated Interleaved Thinking

To enable robust cross-domain generalization, we move beyond traditional tool-use paradigms (which often separate reasoning from execution) and introduce *Tool-Integrated Interleaved Thinking*. Unlike standard methods where models perform comprehensive reasoning upfront or treat tool outputs as isolated events, Interleaved Thinking enforces an alternating cycle of *inference* and *tool invocation*.

The Plan-Action-Reflection Cycle As illustrated in Figure 1, standard Tool RL often discards intermediate reasoning states or treats them linearly (Fig. 1a). In contrast, our Interleaved Thinking framework (Fig. 1b) preserves the entire reasoning history, forming a continuous “Plan-Action-Reflection” loop:

- Plan (Thinking):** The model generates a dedicated thinking block to analyze the current state and formulate a hypothesis.
- Action (Tool Call):** Based on the plan, the model invokes a tool (e.g., a code interpreter).

- Reflection (Observation Integration):** Crucially, the model retains the reasoning context and the tool output to self-reflect, correcting deviations before the next step.

This alternating pattern is vital for long-horizon tasks. By grounding each step in the latest tool execution results rather than initial assumptions, the model reduces error propagation and hallucination—key factors when transferring skills from math domains to complex, open-ended tasks.

4.2 Robust Optimization Objective for Long-Horizon Tool Reasoning

Training the Interleaved Thinking policy introduces unique challenges not adequately addressed by standard objectives like GRPO. The multi-turn nature of the “Plan-Action-Reflection” cycle often involves extremely long contexts (up to 62K tokens) and deep interaction horizons (up to 200 turns), leading to: (i) **severe reward sparsity**, where success is only observable at the end of a long reasoning trajectory; and (ii) **high-variance credit assignment**, where early planning errors dominate the final outcome but receive weak learning signals.

To address these issues, we propose a robust optimization objective tailored for Interleaved Thinking, built upon Group Relative Policy Optimization (GRPO) (Shao et al., 2024) with a novel token-level loss aggregation strategy.

Group Relative Policy Optimization. For a given problem instance q , we sample a group of K trajectories $\{\tau_i\}_{i=1}^K$ generated via the interleaved process. Each trajectory τ_i yields a final scalar reward R_i . Instead of relying on absolute rewards, GRPO computes a *relative advantage* within the group:

$$A_i = R_i - \frac{1}{K} \sum_{j=1}^K R_j. \quad (3)$$

This relative formulation automatically normalizes reward scales across problems of varying difficulty, mitigating training instability caused by the heterogeneous task distributions typical in cross-domain transfer.

Dr. GRPO: Token-Level Loss Aggregation. While GRPO stabilizes optimization at the trajectory level, directly aggregating loss over all tokens remains brittle for long interleaved sequences. We therefore utilize Dr. GRPO, which introduces token-level loss aggregation to improve gradient fidelity (Liu et al., 2025d).

Let $r_{i,t}$ denote the policy ratio at token position t of trajectory τ_i :

$$r_{i,t} = \frac{\pi_{\theta}(a_{i,t} | s_{i,t})}{\pi_{\theta_{\text{old}}}(a_{i,t} | s_{i,t})}. \quad (4)$$

The Dr. GRPO objective is defined as:

$$\mathcal{L}_{\text{Dr-GRPO}} = \sum_{t=1}^T w_t \cdot \mathbb{E}_i \left[-\min(r_{i,t} A_i, \text{clip}(r_{i,t}, 1 - \epsilon, 1 + \epsilon) A_i) \right], \quad (5)$$

where w_t is a token-level aggregation weight. This formulation allows gradients to focus on tokens that are causally important for the "Plan-Action" decisions, rather than being diluted by long spans of low-impact text.

4.3 Token-Level Importance and Rejection Sampling

We construct the token-level weight w_t using a combination of *Token-level Importance Sampling (IS)* and *Token-level Rejection Sampling (RS)*:

$$w_t = \alpha \cdot \text{IS}_t + (1 - \alpha) \cdot \mathbb{I}[\text{RS}_t], \quad (6)$$

where IS_t prioritizes tokens involved in tool invocation and reasoning transitions (the "thinking" tags), while RS_t downweights stylistic or repetitive tokens. This decomposition is essential for reinforcing the structural integrity of the interleaved thinking process.

4.4 Dual-Component Reward System

To guide the model toward effective Interleaved Thinking while ensuring robustness across domain shifts, we implement a dual-component reward system. The overall reward function is:

$$R = R_{\text{outcome}} + R_{\text{format}}, \quad (7)$$

with $R \in \{-2, -1, 0, 1, 2\}$.

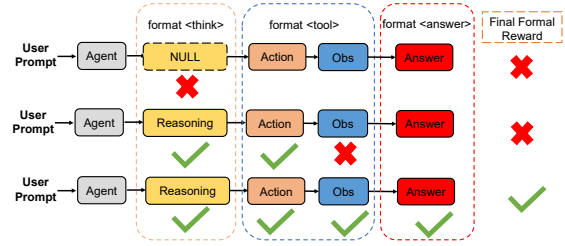


Figure 4: Illustration of format reward. The agent must maintain specific Interleaved Thinking structure (Plan-Action-Reflection) to receive positive reinforcement.

4.4.1 Outcome Reward

The Outcome Reward evaluates the correctness of the final answer, targeting domain-invariant problem-solving skills. Let \hat{a} denote the model's output and a^* the ground-truth solution:

$$R_{\text{outcome}} = \begin{cases} +1, & \text{if } \hat{a} \equiv a^*, \\ -1, & \text{otherwise.} \end{cases} \quad (8)$$

This reward encourages abstract reasoning (e.g., mapping problems to computable steps) regardless of the domain.

4.4.2 Format Reward

The Format Reward is critical for Interleaved Thinking. It incentivizes the maintenance of the correct "thinking" and "tool use" structure, ensuring the agent does not degenerate into unstructured generation. Our format reward can be defined:

$$R_{\text{format}} = \begin{cases} +1 & \text{valid interleaved structure,} \\ 0 & \text{minor formatting issues,} \\ -1 & \text{broken thinking/tool cycle.} \end{cases} \quad (9)$$

As shown in Figure 4, only when the interaction includes necessary content with correct interleaved format can the agent achieve a positive reward.

4.5 Dynamic Difficulty Adjustment via Online Filtering

To efficiently train Interleaved Thinking on complex tasks, we address the sample efficiency problem caused by static datasets. We utilize *Online Rollout Filtering*, a dynamic curriculum strategy.

For each prompt q , we maintain a running estimate of its pass rate:

$$\text{PassRate}(q) = \frac{\# \text{ successful rollouts}}{\# \text{ total rollouts}}. \quad (10)$$

Prompts are included in training only if $\text{PassRate}(q) \in [0.1, 0.9]$. This ensures the agent

continuously trains on problems within its *Zone of Proximal Development*, where the interleaved reasoning strategy is actively being refined, rather than on trivial or impossible tasks.

5 Experiment

We design our experiments to answer the core research question: *Can an agent trained exclusively on mathematical tool-use tasks generalize its reasoning capabilities to diverse, open-ended domains?* To this end, we train our models solely on code-integrated math datasets and evaluate them on a broad spectrum of reasoning benchmarks, including science, logic, and general knowledge.

5.1 Experimental Setup

Datasets and Environment. Our training data is strictly confined to the mathematical domain. We utilize the **Math3-5** dataset (Zeng et al., 2025) and the math-subset of **DeepScaler** (Luo et al., 2025b), filtering for samples solvable via Python code. The environment consists of a sandboxed Python interpreter. For evaluation, we classify benchmarks into two categories: *In-Domain (Math)*: MATH-500, HMMT 25, AIME 24, and AIME 25. *Out-of-Domain (General)*: GPQA (Graduate-Level Science), TheoremQA (STEM), and the held-out test set of WebInstruct (Multidisciplinary).

Implementation Details. We train two variants of RITE based on Qwen2.5-7B and Qwen2.5-32B. Unlike standard approaches that require Supervised Fine-Tuning (SFT) warm-up, we employ a Cold-Start RL strategy (Zero-RL), initializing directly from the base or instruct checkpoints without specific tool-tuning. We use the Dr. GRPO objective with a group size of $G = 16$. The token-level importance weight α is set to 0.8. We implement the Dynamic Difficulty Adjustment by filtering prompts with pass rates outside $[0.1, 0.9]$. The maximum context length expands from 16k to 24k during training to accommodate the growing depth of the Interleaved Thinking cycles.

5.2 Baseline Models

We evaluate our approach against a comprehensive set of baseline models across two parameter scales to demonstrate its effectiveness.

7B Parameter Baselines : **Qwen2.5-7B**, the instruct model; **Qwen2.5-7B-TIR**, a tool-integrated reasoning with instruct model; **ToRL-7B** (Li

et al., 2025a), which applies domain-specific reinforcement learning to mathematical tool usage; **ZeroTIR-7B** (Mai et al., 2025), implementing zero-shot tool integration from base models; and **SimpleTIR-7B** (Xue et al., 2025), a single-domain tool RL approaches.

32B Parameter Baselines : **Qwen2.5-32B**, the instruct model; **Qwen2.5-32B-TIR**, a tool-integrated reasoning with instruct model; **ReTool-32B** (Feng et al., 2025), which employs cold-start supervised fine-tuning followed by domain-specific RL, and **SimpleTIR-32B** (Xue et al., 2025), a scaled single-domain tool RL approaches.

These baselines enable systematic comparison across different training paradigms: instruct models without tool training, supervised fine-tuning approaches, domain-specific RL methods, and our proposed Reinforcement Learning for Interleaved Tool Execution (RITE) framework.

5.3 Main Results

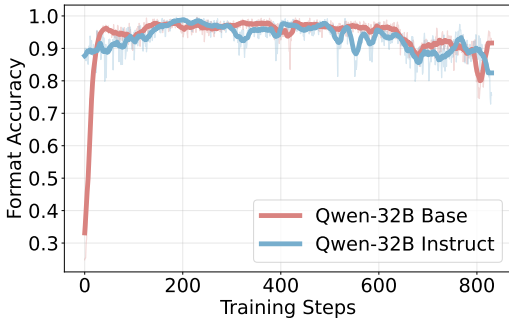
Table 1 presents the comprehensive evaluation results. We have the following observations.

RITE achieves state-of-the-art results across both 7B and 32B model scales. The 7B variant attains 36.7% on AIME 25, outperforming prior tool-augmented baselines, and demonstrates strong cross-domain generalization with 53.2% on averaged general reasoning benchmarks. Scaling to 32B parameters yields further gains: 71.3% on AIME 24, 56.7% on AIME 25, and 62.0% on the general set, significantly surpassing comparable baselines. These results validate the framework’s scalability and highlight the contribution of its standardized interface, dual-component reward, and structured prompting to superior performance in both mathematical and general reasoning tasks.

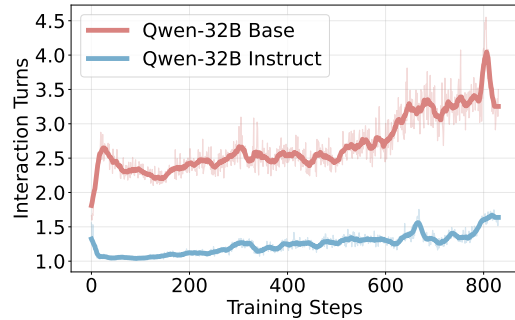
Strong Generalization from Math to Open Domains. The most significant finding is RITE’s performance on **WebInstruct** and **GPQA**, benchmarks unrelated to the math training data. RITE-7B achieves **73.6%** on WebInstruct, surpassing the math-trained ToRL-7B by 5.4%. Similarly, RITE-32B sets a new state-of-the-art with **82.3%**, significantly outperforming ReTool-32B (76.8%). This validates our hypothesis: *the “Plan-Action-Reflection” cycle learned in math is a transferable cognitive structure*. Unlike baselines that overfit to math-specific templates, RITE’s interleaved thinking allows the agent to break down general

Model	TIR	Zero-RL	GPQA	TheoremQA	WebInst.	MATH-500	HMMT 25	AIME 24	AIME 25	General	All
			Avg@4	Avg@2	Avg@2	Avg@2	Avg@2	Avg@16	Avg@16	-	-
7B Size Models											
Qwen2.5-7B	✗	–	32.4	41.4	60.4	51.9	0.0	3.2	1.1	44.7	27.2
Qwen2.5-7B-TIR	✓	–	28.5	35.2	52.3	18.0	1.9	1.7	0.6	38.7	19.7
ToRL-7B	✓	✓	35.8	48.6	68.2	82.2	26.3	40.2	27.9	50.9	47.0
ZeroTIR-7B	✓	✓	34.2	46.8	65.7	80.2	22.5	39.6	25.0	48.9	44.9
SimpleTIR-7B	✓	✓	36.2	49.5	70.7	88.4	29.7	50.5	30.9	52.1	50.8
RITE-7B(Ours)	✓	✓	35.6	50.3	73.6	86.0	31.1	50.5	36.7	53.2	52.0
32B Size Models											
Qwen2.5-32B	✗	–	38.6	48.2	68.5	43.1	0.2	4.2	1.6	51.8	29.2
Qwen2.5-32B-TIR	✓	–	35.1	42.8	61.9	37.0	5.2	7.1	5.0	46.6	27.7
ReTool-32B	✓	✗	42.3	54.7	76.8	93.2	36.2	67.0	49.3	57.9	59.9
SimpleTIR-32B	✓	✓	43.5	56.3	78.1	92.9	34.6	59.9	49.2	59.3	59.2
RITE-32B(Ours)	✓	✓	46.4	57.4	82.3	93.8	39.5	71.3	56.7	62.0	63.9

Table 1: Main results on seven reasoning benchmarks. “TIR” denotes Tool-Integrated Reasoning. General is the average of GPQA, TheoremQA, and WebInst. Notably, RITE is trained *only* on math tasks but achieves best on general domains.



Training Steps vs Format Accuracy



Training Steps vs Interaction Turns

Figure 5: Training progress of format accuracy and interaction turns.

454 problems (e.g., physics or business logic) into exe- 472
455 cutable sub-steps. 473

456 **Robustness in Long-Horizon Math Tasks.** On 475
457 complex benchmarks like AIME 25, which re- 476
458 quires deep multi-step reasoning, RITE shows a de- 477
459 cisive advantage. RITE-7B achieves **36.7%**, a re- 478
460 lative improvement of **18.7%** over SimpleTIR-7B. 479
461 This improvement stems from Dr. GRPO, which 480
462 effectively assigns credit in long trajectories where 481
463 standard GRPO often fails due to reward sparsity.

464 5.4 Analysis of Cross-Domain Transfer 482

465 To visualize the transferability of the learned policy, 483
466 as shown in Figure 6, we decompose performance 484
467 on WebInstruct into distinct sub-domains: Math, 485
468 Physics, Business, Philosophy, and Biology. 486

469 RITE (represented by the red area) encompasses 487
470 the baselines in almost all dimensions. Notably, the 488
471 performance gap is widest in **Physics** and **Business**. 489
490

472 These domains share a structural similarity with 473
474 mathematics—they require rigorous logic and step- 475
476 by-step verification—but differ in vocabulary and 477
478 knowledge retrieval. The *Plan-Action-Reflection* 479
480 mechanism allows the model to handle this shift: 481
482 the “Plan” step grounds the problem in the cur- 483
484 rent context, while the “Action” (tool use) offloads 484
485 computation, regardless of whether the variables 485
486 represent quantum states or financial assets. 486
487

481 5.5 Training Dynamics 481

482 Figure 5 visualizes the evolution of the agent during 482
483 training. **Structural Adaptation:** The agent, start- 483
484 ing from a base model with zero prior tool knowl- 484
485 edge, achieves near-perfect format compliance (In- 485
486 terleaved Structure) within the first 50 steps. This 486
487 rapid adaptation is driven by the **Format Component** 487
488 of our Dual-Component Reward. **Emergent** 488
489 **Reasoning Depth:** As training progresses, the av- 489
490 erage interaction turns increase from ~ 1.6 to ~ 4.0 . 490

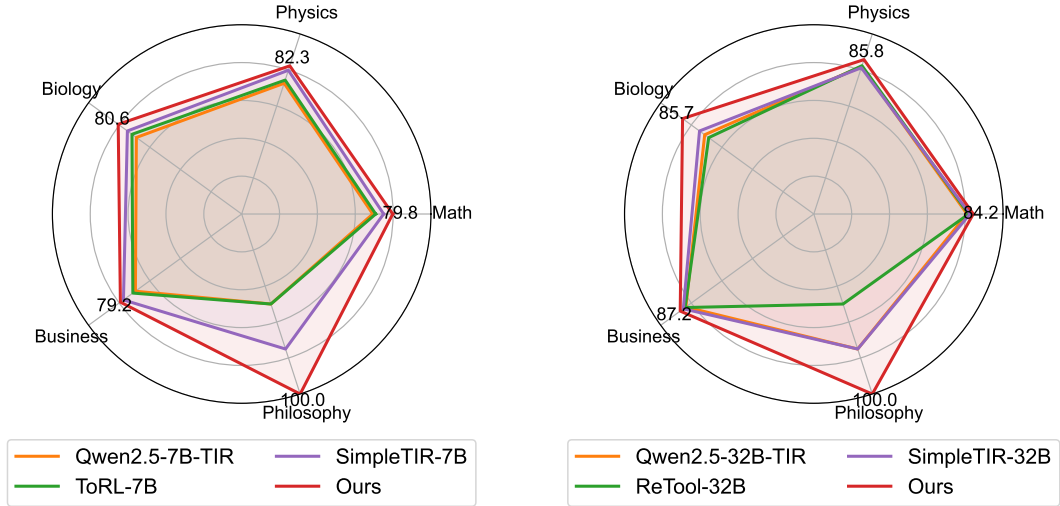


Figure 6: Cross-domain performance comparison between different Tool RL methods on WebInstruct.

Table 2: Ablation study on AIME24 (In-Domain) and WebInstruct (Out-of-Domain).

Method	AIME24	WebInst.
RITE (Full)	71.3	82.3
w/o Token Opt.	56.7	61.9
w/o Dual Reward	67.0	76.8
w/o Interleaved	59.9	78.1

Unlike the instruct-tuned baseline which stagnates at lower interaction depths, the RITE agent learns to verify and self-correct, utilizing more turns to solve harder problems. This confirms that our **Dynamic Curriculum** successfully pushes the model to explore its Zone of Proximal Development.

5.6 Ablation Study

As shown in Table 2, we validate the contribution of each component in RITE using the 32B model.

Impact of Token-Level Optimization (w/o Token Opt.). Removing token-level optimization (replacing Dr. GRPO with standard GRPO and removing importance/rejection sampling) causes the most severe degradation, with AIME24 performance dropping by 14.6%. This highlights that for long interleaved reasoning chains, token-level optimization is critical to stabilize gradients and mitigate the high variance of credit assignment.

Impact of Dual-Component Reward (w/o Dual Reward). Removing the format-aware reward

leads to a significant drop in WebInstruct (-5.5%). Without explicit structural reinforcement, the model struggles to maintain the “Plan-Action-Reflection” loop in open-ended domains, often reverting to unstructured text generation that fails to leverage tools effectively.

Impact of Interleaved Thinking (w/o Interleaved). We trained a variant where tool execution is treated linearly (Plan → Code → Result) without the enforced reflection cycle. This led to a 4.2% drop on WebInstruct. The lack of an explicit reflection step increases error propagation, proving that the cyclical structure is the key engine for cross-domain generalization.

6 Conclusion

We systematically examine the cross-domain generalization of large language model (LLM) agents equipped with a code interpreter, trained solely via reinforcement learning (RL) on mathematical problem-solving tasks. Our results show that RL-trained tool use in one domain transfers effectively to diverse reasoning tasks, demonstrating strong performance and high token efficiency. To enable such transfer, we propose the Reinforcement Learning for Interleaved Tool Execution (RITE) framework, featuring a standardized tool interface, dual-component reward system, and XML-based prompt template to promote domain-agnostic learning. Extensive experiments demonstrate the effectiveness of our proposed RITE.

7 Limitations

Despite the promising results, several limitations remain in our current study:

- **Restricted Tool Diversity:** Our experiments primarily focus on a single type of tool (code interpreter). The generalization performance across domains involving fundamentally different tools (e.g., knowledge base retrieval, image processing) remains to be explored.
- **Domain Shift Extremes:** While our benchmarks cover a variety of reasoning domains, they do not encompass highly specialized or adversarial domains where domain-specific knowledge or tool customization may be indispensable.
- **Reward Engineering:** The dual-component reward system relies on carefully designed heuristics and domain-agnostic abstraction, which may require manual tuning for new tasks or tools.
- **Scalability and Efficiency:** The framework assumes access to sufficient computational resources for RL training and evaluation. Scaling to more complex domains or larger toolsets may introduce additional efficiency and stability challenges.

Future work will address these limitations by extending the framework to support a broader range of tools, exploring more extreme domain shifts and improving scalability and prompt flexibility. We believe these directions will further advance the generalization of tool-augmented LLM agents.

8 Ethical Considerations

As large language models (LLMs) increasingly integrate with external tools and autonomous reasoning capabilities, it is imperative to address the ethical implications of their development and deployment. Our research on Reinforcement Learning for Interleaved Tool Execution (RITE) aims to enhance the reasoning and problem-solving abilities of AI agents; however, we acknowledge several critical considerations regarding safety, misuse, and bias.

- **Safety and Code Execution Risks** A core component of our framework is the integration of a code interpreter tool. While this significantly boosts mathematical and scientific reasoning, it introduces security risks associated

with arbitrary code execution. To mitigate these risks during our experiments, all tool interactions were confined to a strictly sandboxed, non-networked Python environment with rigorous resource limits (CPU, memory, and timeout constraints). We strongly advise that any deployment of the models or frameworks released in this study must implement similar isolation mechanisms (e.g., Docker containers, gVisor) to prevent potential malicious exploitation, such as unauthorized file system access or denial-of-service attacks.

- **Dual-Use and Potential Misuse** The enhanced reasoning and planning capabilities demonstrated by our agents—particularly their ability to generalize from mathematical tasks to broader domains—raise concerns regarding dual-use. While our primary goal is to advance scientific research and educational support, agents capable of complex multi-step planning and code generation could potentially be misused by malicious actors to automate cyber-attacks or generate obfuscated malware. We emphasize that the release of our models is intended for research purposes, and we encourage the community to develop robust safety guardrails and monitoring systems alongside capability improvements.
- **Data Usage and Compliance** Our training and evaluation processes utilized publicly available datasets (e.g., MATH, AIME, GPQA, WebInstruct). We have adhered to the licensing terms of these datasets and ensured that no personally identifiable information (PII) was processed or generated during our experiments. Our proposed data synthesis and filtering pipelines rely solely on open-source data and model-generated content, avoiding the use of private or proprietary user data.
- **Environmental Impact** Reinforcement learning, particularly with iterative rollout and rejection sampling, is computationally intensive. To minimize our carbon footprint, we employed efficiency-oriented techniques such as Online Rollout Filtering and dynamic context expansion, which prevent the waste of computational resources on samples that provide low information gain.

References

- 637 Syeda Nahida Akter, Shrimai Prabhumoye, Matvei
638 Novikov, Seungju Han, Ying Lin, Evelina Bakhtu-
639 rina, Eric Nyberg, Yejin Choi, Mostofa Patwary, Mo-
640 hammad Shoeybi, et al. 2025. Nemotron-crossthink:
641 Scaling self-learning beyond math reasoning. *arXiv*
642 *preprint arXiv:2504.13941*.
- 643 Wenhui Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan,
644 Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony
645 Xia. 2023. [Theoremqa: A theorem-driven question](#)
646 [answering dataset](#). *Preprint*, arXiv:2305.12524.
- 647 Zhoujun Cheng, Shibo Hao, Tianyang Liu, Fan Zhou,
648 Yutao Xie, Feng Yao, Yuexin Bian, Yonghao Zhuang,
649 Nilabjo Dey, Yuheng Zha, et al. 2025. Re-
650 visiting reinforcement learning for llm reasoning
651 from a cross-domain perspective. *arXiv preprint*
652 *arXiv:2506.14965*.
- 653 DeepSeek-AI Team. 2025. DeepSeek-R1: Incentivizing
654 Reasoning Capability in LLMs via Reinforcement
655 Learning. *arXiv preprint arXiv:2501.12948*.
- 656 Guanting Dong, Yifei Chen, Xiaoxi Li, Jiajie Jin,
657 Hongjin Qian, Yutao Zhu, Hangyu Mao, Guorui
658 Zhou, Zhicheng Dou, and Ji-Rong Wen. 2025. Tool-
659 Star: Empowering LLM-Brained Multi-Tool Rea-
660 soner via Reinforcement Learning. *arXiv preprint*
661 *arXiv:2505.16410*.
- 662 Lutfi Eren Erdogan, Nicholas Lee, Sehoon Kim, Suhong
663 Moon, Hiroki Furuta, Gopala Anumanchipalli, Kurt
664 Keutzer, and Amir Gholami. 2025. Plan-and-act:
665 Improving planning of agents for long-horizon tasks.
666 *arXiv preprint arXiv:2503.09572*.
- 667 Jiazhan Feng, Shijue Huang, Xingwei Qu, Ge Zhang,
668 Yujia Qin, Baoquan Zhong, Chengquan Jiang, Jinxin
669 Chi, and Wanjun Zhong. 2025. Retool: Reinforce-
670 ment learning for strategic tool use in llms. *arXiv*
671 *preprint arXiv:2504.11536*.
- 672 Huan-ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu,
673 Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao
674 Qiu, Xuan Qi, Yiran Wu, et al. 2025a. A survey
675 of self-evolving agents: On path to artificial super
676 intelligence. *arXiv preprint arXiv:2507.21046*.
- 677 Jiaxuan Gao, Wei Fu, Minyang Xie, Shusheng Xu,
678 Chuyi He, Zhiyu Mei, Banghua Zhu, and Yi Wu.
679 2025b. Beyond ten turns: Unlocking long-horizon
680 agentic search with large-scale asynchronous rl.
681 *arXiv preprint arXiv:2508.07976*.
- 682 Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar
683 Nath, Bing Liu, and Sean Hendryx. 2025. Rubrics as
684 rewards: Reinforcement learning beyond verifiable
685 domains. *arXiv preprint arXiv:2507.17746*.
- 686 Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul
687 Arora, Steven Basart, Eric Tang, Dawn Song, and Ja-
688 cob Steinhardt. 2021. Measuring Mathematical Prob-
689 lem Solving With the MATH Dataset. In *NeurIPS*
690 *Datasets and Benchmarks*.
- Chuxuan Hu, Yuxuan Zhu, Antony Kellermann, Caleb
Biddulph, Suppakit Waiwitlikhit, Jason Benn, and
Daniel Kang. 2025. Breaking barriers: Do reinforce-
ment post training gains transfer to unseen domains?
arXiv preprint arXiv:2506.19733.
- Maggie Huan, Yuetai Li, Tuney Zheng, Xiaoyu Xu,
Seungone Kim, Minxin Du, Radha Poovendran, Gra-
ham Neubig, and Xiang Yue. 2025. Does math rea-
soning improve general llm capabilities? understand-
ing transferability of llm reasoning. *arXiv preprint*
arXiv:2507.00432.
- Zenan Huang, Yihong Zhuang, Guoshan Lu, Zeyu Qin,
Haokai Xu, Tianyu Zhao, Ru Peng, Jiaqi Hu, Zhan-
ming Shen, Xiaomeng Hu, et al. 2025. Reinforce-
ment learning with rubric anchors. *arXiv preprint*
arXiv:2508.12790.
- Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon,
Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei
Han. 2025. Search-r1: Training llms to reason and
leverage search engines with reinforcement learning.
arXiv preprint arXiv:2503.09516.
- Xuefeng Li, Haoyang Zou, and Pengfei Liu. 2025a.
Torl: Scaling tool-integrated rl. *arXiv preprint*
arXiv:2503.23383.
- Yu Li, Zhuoshi Pan, Honglin Lin, Mengyuan Sun, Con-
ghui He, and Lijun Wu. 2025b. Can one domain
help others? a data-centric study on multi-domain
reasoning via reinforcement learning. *arXiv preprint*
arXiv:2507.17512.
- Heng Lin and Zhongwen Xu. 2025. Understand-
ing tool-integrated reasoning. *arXiv preprint*
arXiv:2508.19201.
- Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingx-
uan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang,
Chaofan Lin, Chen Dong, et al. 2025a. Deepseek-
v3. 2: Pushing the frontier of open large language
models. *arXiv preprint arXiv:2512.02556*.
- Junteng Liu, Yunji Li, Chi Zhang, Jingyang Li, Aili
Chen, Ke Ji, Weiyu Cheng, Zijia Wu, Chengyu Du,
Qidi Xu, et al. 2025b. Webexplorer: Explore and
evolve for training long-horizon web agents. *arXiv*
preprint arXiv:2509.06501.
- Wei Liu, Siya Qi, Xinyu Wang, Chen Qian, Yali Du,
and Yulan He. 2025c. Nover: Incentive training
for language models via verifier-free reinforcement
learning. *arXiv preprint arXiv:2505.16022*.
- Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi,
Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin.
2025d. Understanding R1-Zero-like Training: A Crit-
ical Perspective. *arXiv preprint arXiv:2503.20783*.
- Junyu Luo, Weizhi Zhang, Ye Yuan, Yusheng Zhao, Jun-
wei Yang, Yiyang Gu, Bohan Wu, Binqi Chen, Ziyue
Qiao, Qingqing Long, et al. 2025a. Large language
model agent: A survey on methodology, applications
and challenges. *arXiv preprint arXiv:2503.21460*.

746	Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi,	Qwen Team. 2025a. Qwen3 technical report . <i>Preprint</i> ,	801
747	William Y. Tang, Manan Roongta, Colin Cai, Jeffrey	arXiv:2505.09388.	802
748	Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica.		
749	2025b. DeepScaleR: Surpassing O1-Preview with a	Tongyi DeepResearch Team. 2025b. Tongyi-	803
750	1.5B Model by Scaling RL. Notion Blog.	deeppresearch. https://github.com/Alibaba-NLP/DeepResearch .	804
			805
751	Xueguang Ma, Qian Liu, Dongfu Jiang, Ge Zhang, Ze-	Hongru Wang, Cheng Qian, Wanjun Zhong, Xiusi Chen,	806
752	jun Ma, and Wenhui Chen. 2025a. General-reasoner:	Jiahao Qiu, Shijue Huang, Bowen Jin, Mengdi Wang,	807
753	Advancing llm reasoning across all domains. <i>arXiv</i>	Kam-Fai Wong, and Heng Ji. 2025. Otc: Optimal	808
754	<i>preprint arXiv:2505.14652</i> .	tool calls via reinforcement learning. <i>arXiv e-prints</i> ,	809
		pages arXiv-2504.	810
755	Xueguang Ma, Qian Liu, Dongfu Jiang, Ge Zhang,	Guofu Xie, Yunsheng Shi, Hongtao Tian, Ting Yao,	811
756	Zejun Ma, and Wenhui Chen. 2025b. General-	and Xiao Zhang. 2025a. Capo: Towards enhancing	812
757	reasoner: Advancing llm reasoning across all do-	llm reasoning through verifiable generative credit	813
758	mains . <i>Preprint</i> , arXiv:2505.14652.	assignment. <i>arXiv preprint arXiv:2508.02298</i> .	814
759	Xinji Mai, Haotian Xu, Xing W, Weinong Wang, Yingy-	Roy Xie, David Qiu, Deepak Gopinath, Dong Lin, Yan-	815
760	ing Zhang, and Wenqiang Zhang. 2025. Agent RL	chao Sun, Chong Wang, Saloni Potdar, and Bhuwan	816
761	Scaling Law: Agent RL with Spontaneous Code Ex-	Dhingra. 2025b. Interleaved reasoning for large lan-	817
762	ecution for Mathematical Problem Solving. <i>arXiv</i>	guage models via reinforcement learning. <i>arXiv</i>	818
763	<i>preprint arXiv:2505.07773</i> .	<i>preprint arXiv:2505.19640</i> .	819
764	MiniMax. 2025. Aligning to what? rethinking	Wujiang Xu, Kai Mei, Hang Gao, Juntao Tan, Zu-	820
765	agent generalization in minimax m2 .	jie Liang, and Yongfeng Zhang. 2025. A-mem:	821
766	https://huggingface.co/blog/MiniMax-AI/	Agentic memory for llm agents. <i>arXiv preprint</i>	822
767	aligning-to-what .	<i>arXiv:2502.12110</i> .	823
768	Aske Plaat, Max van Duijn, Niki van Stein, Mike Preuss,	Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru	824
769	Peter van der Putten, and Kees Joost Batenburg. 2025.	Li, Xiaosen Zheng, Zejun Ma, and Bo An. 2025.	825
770	Agentic large language models, a survey. <i>arXiv</i>	Simpletir: End-to-end reinforcement learning for	826
771	<i>preprint arXiv:2503.23037</i> .	multi-turn tool-integrated reasoning. <i>arXiv preprint</i>	827
		<i>arXiv:2509.02479</i> .	828
772	David Rein, Betty Li Hou, Asa Cooper Stickland,	Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong	829
773	Jackson Petty, Richard Yuanzhe Pang, Julien Di-	Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Hin-	830
774	rani, Julian Michael, and Samuel R. Bowman. 2023.	rich Schütze, Volker Tresp, and Yunpu Ma. 2025.	831
775	Gpqa: A graduate-level google-proof q&a bench-	Memory-rl: Enhancing large language model agents	832
776	mark . <i>Preprint</i> , arXiv:2311.12022.	to manage and utilize memories via reinforcement	833
		learning. <i>arXiv preprint arXiv:2508.19828</i> .	834
777	Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu,	Tianyu Yu, Bo Ji, Shouli Wang, Shu Yao, Zefan Wang,	835
778	Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan	Ganqu Cui, Lifan Yuan, Ning Ding, Yuan Yao,	836
779	Zhang, YK Li, Y Wu, et al. 2024. Deepseekmath:	Zhiyuan Liu, et al. 2025. Rlpr: Extrapolating rlvr	837
780	Pushing the limits of mathematical reasoning in open	to general domains without verifiers. <i>arXiv preprint</i>	838
781	language models. <i>arXiv preprint arXiv:2402.03300</i> .	<i>arXiv:2506.18254</i> .	839
782	Shreyas Singh, Kunal Singh, and Pradeep Moturi. 2025.	Weihaio Zeng, Yuzhen Huang, Qian Liu, Wei Liu,	840
783	Fathom-deepresearch: Unlocking long horizon in-	Keqing He, Zejun Ma, and Junxian He. 2025.	841
784	formation retrieval and synthesis for slms. <i>arXiv</i>	SimpleRL-Zoo: Investigating and Taming Zero Re-	842
785	<i>preprint arXiv:2509.24107</i> .	inforcement Learning for Open Base Models in the	843
		Wild. <i>arXiv preprint arXiv:2503.18892</i> .	844
786	Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen,	Ruo Chen Zhou, Minrui Xu, Shiqi Chen, Junteng Liu,	845
787	Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-	Yunqi Li, Xinxin Lin, Zhengyu Chen, and Junxian He.	846
788	Rong Wen. 2025. R1-Searcher: Incentivizing the	2025a. Does learning mathematical problem-solving	847
789	Search Capability in LLMs via Reinforcement Learn-	generalize to broader reasoning? <i>arXiv preprint</i>	848
790	ing. <i>arXiv preprint arXiv:2503.05592</i> .	<i>arXiv:2507.04391</i> .	849
791	Yi Su, Dian Yu, Linfeng Song, Juntao Li, Haitao	Xiangxin Zhou, Zichen Liu, Anya Sims, Haonan Wang,	850
792	Mi, Zhaopeng Tu, Min Zhang, and Dong Yu. 2025.	Tianyu Pang, Chongxuan Li, Liang Wang, Min Lin,	851
793	Crossing the reward bridge: Expanding rl with verifi-	and Chao Du. 2025b. Reinforcing general reasoning	852
794	able rewards across diverse domains. <i>arXiv preprint</i>	without verifiers. <i>arXiv preprint arXiv:2505.21493</i> .	853
795	<i>arXiv:2503.23829</i> .		
796	Kimi Team, Angang Du, Bofei Gao, Bofei Xing,		
797	Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun		
798	Xiao, Chenzhuang Du, Chonghua Liao, et al. 2025.		
799	Kimi k1.5: Scaling reinforcement learning with llms.		
800	<i>arXiv preprint arXiv:2501.12599</i> .		

854 Yang Zhou, Sunzhu Li, Shunyu Liu, Wenkai Fang, Jiale
855 Zhao, Jingwen Yang, Jianwei Lv, Kongcheng Zhang,
856 Yihe Zhou, Hengtong Lu, et al. 2025c. Breaking
857 the exploration bottleneck: Rubric-scaffolded rein-
858 forcement learning for general llm reasoning. *arXiv*
859 *preprint arXiv:2508.16949*.

860 Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim,
861 Alok Prakash, Daniela Rus, Jinhua Zhao, Bryan
862 Kian Hsiang Low, and Paul Pu Liang. 2025d.
863 Mem1: Learning to synergize memory and reason-
864 ing for efficient long-horizon agents. *arXiv preprint*
865 *arXiv:2506.15841*.

866 Jiaru Zou, Ling Yang, Yunzhe Qi, Sirui Chen, Mengting
867 Ai, Ke Shen, Jingrui He, and Mengdi Wang. 2025.
868 Autotool: Dynamic tool selection and integration for
869 agentic reasoning. *arXiv preprint arXiv:2512.13278*.

870 A Implementation Details

871 A.1 Standardized Tool Interface

872 We introduce a standardized tool interface as a crit-
873 ical component of our Reinforcement Learning for
874 Interleaved Tool Execution framework, centered
875 around a specialized answer tool. Unlike tradi-
876 tional tool-calling approaches that primarily focus
877 on interacting with external APIs, our interface en-
878 forces a unified response format and serves as an
879 explicit termination signal for the model’s reason-
880 ing process, abstracting domain-specific nuances
881 to promote transferable invocation patterns.

882 Formally, the tool is defined as a function:

$$883 \mathcal{T}_{\text{answer}} : \mathcal{X} \mapsto \mathcal{Y}, \quad (11)$$

884 where \mathcal{X} is the internal reasoning trace and \mathcal{Y} is the
885 final output. The output must satisfy the constraint

$$886 y \in \mathcal{Y}, \quad y = \boxed{\text{answer}}, \quad (12)$$

887 with a being the predicted solution content.

888 This design yields several advantages that facili-
889 tate cross-domain transfer:

- 890 • **Format Standardization:** Enforcing final an-
891 swers in `\boxed{}` ensures consistent outputs
892 for automated cross-domain evaluation, reduc-
893 ing dependence on domain-specific parsing.
- 894 • **Early Learning Signal:** The fixed schema of-
895 fers an immediate reward in early RL training,
896 speeding adaptation to the target format and
897 discovery of generalizable tool-use strategies.
- 898 • **Explicit Termination:** Calling $\mathcal{T}_{\text{answer}}$ de-
899 fines task completion, providing a domain-
900 invariant halting signal transferable across do-
901 mains (e.g., math or science tasks).

The standardized tool interface plays a pivotal
902 role in enabling effective zero-RL training, where
903 the model is optimized directly from a base model
904 without prior exposure to tool-use or structured
905 reasoning data. In this setting, the model initially
906 lacks awareness of how to format tool invocations
907 or when to terminate reasoning. The fixed and
908 minimal interface design—centered around the sin-
909 gle answer tool and the enforced `\boxed{}` output
910 schema—provides a strong, low-entropy learning
911 signal early in training. 912

913 A.2 Verifiable Reward Mechanism via 914 Symbolic Equivalence

915 A common failure mode in reinforcement learn-
916 ing with language models is *reward hacking*,
917 where agents exploit superficial patterns in the re-
918 ward function—such as formatting tricks or partial
919 guesses—without producing logically correct solu-
920 tions. This issue is especially severe in mathemat-
921 ical reasoning tasks involving tools, where incor-
922 rect intermediate reasoning can still lead to plausi-
923 ble-looking final answers. 923

924 To ensure that improvements reflect genuine rea-
925 soning capability, we design a verifiable reward
926 mechanism based on symbolic equivalence. 926

927 We introduce **MathRuleGrader**, a determinis-
928 tic evaluator that rigorously verifies mathematical
929 correctness. Unlike simple string matching, Math-
930 RuleGrader integrates: 930

- 931 • **LaTeX parsing** to extract final answers en-
932 closed in `\boxed{}`;
- 933 • **Symbolic equivalence checking** using
934 SymPy to verify mathematical correctness;
- 935 • **Code interpreter validation** to ensure con-
936 sistency between intermediate computations
937 and the final result. 937

938 This multi-layer verification ensures that only
939 mathematically valid solutions are rewarded, re-
940 gardless of superficial textual similarity. 940

941 A.3 Specialized Prompt Template Design

942 We employ a specialized prompt template,
943 `chat_template`, to structure interactions between
944 the model and tools. This template introduces sev-
945 eral innovations: 945

- 946 • **XML-Based Tool Call Format:** Tool calls are wrapped in
947 `<tool_call>...</tool_call>` tags, 948

949 simplifying the model’s learning of correct
950 syntax.

- 951 • **Explicit Thinking Section:** The model is
952 encouraged to reason through problems in a
953 dedicated <think>...</think> block before
954 making tool calls.
- 955 • **Structured Conversation Flow:** Multi-turn
956 dialogue is organized into rounds with explicit
957 role markers, such as [Round 0] USER: and
958 ASSISTANT:.
- 959 • **Tool Response Integration:** Tool outputs are
960 integrated using the TOOL: marker for seam-
961 less information flow.

962 These design choices offer several advantages:

- 963 • Clear structural boundaries for tool calls and
964 reasoning steps.
- 965 • Encouragement of deliberate, step-wise
966 problem-solving.
- 967 • Enhanced coherence in multi-turn conversa-
968 tions.
- 969 • Standardized integration of tool responses.

970 Empirical results show that this template signifi-
971 cantly improves the model’s ability to format tool
972 calls correctly and maintain coherent, multi-step
973 interactions.

974 Example template snippet:

```
975 [Round 0] USER: ...  
976 ASSISTANT: ...  
977 <think>  
978 reasoning process here  
979 </think>  
980 <tool_call>  
981 {"name": <function-name>, "arguments": <  
982 args-dict>}  
983 </tool_call>  
984 TOOL: {"name": "tool_name", "content":  
985 response_content}
```

990 A.4 Experimental Details

991 All experiments are conducted using identical com-
992 putational resources and evaluation protocols to
993 ensure fair comparison. Models are trained on
994 8xA100 GPUs with mixed precision training and
995 gradient accumulation. Evaluation is performed us-
996 ing consistent random seeds and identical sampling
997 parameters across all baselines. For cross-domain
998 evaluation, we ensure that no target domain data is

used during training, maintaining strict separation
between training and evaluation domains to provide
unbiased assessment of generalization capabilities.

A.5 Answer Tool Design

We introduce a specialized answer tool as a critical
component of our RL framework. Unlike tradi-
tional tool-calling approaches that primarily focus
on interacting with external APIs, our answer tool
enforces a standardized response format and serves
as an explicit termination signal for the model’s
reasoning process. The answer tool is defined via a
structured JSON schema:

```
{  
  "type": "function",  
  "function": {  
    "name": "answer",  
    "description": "Respond to the  
    user",  
    "parameters": {  
      "type": "object",  
      "properties": {  
        "answer": {"type": "  
        string", "description":  
        "Response content, place  
        your final answer  
        within \boxed{} notation  
        ."}  
      },  
      "required": ["answer"]  
    }  
  }  
}
```

This design provides several advantages:

- **Format Standardization:** By requiring final
answers to be enclosed in \boxed{} notation,
the model outputs are consistent and easily
extractable for automated evaluation.
- **Early Learning Signal:** The answer tool
offers an immediate and clear learning sig-
nal during early RL training, accelerating the
model’s adaptation to the expected output for-
mat.
- **Explicit Termination:** It serves as an explicit
endpoint for the model’s reasoning, clearly
indicating when a problem has been solved.

This mechanism significantly reduces the time
required for the model to match the performance
of supervised fine-tuned (SFT) baselines in terms
of answer formatting and termination.

1052 **A.6 XML-Based Prompt Template**

1053 We employ a structured XML-based prompt tem-
1054 plate \mathcal{P} to govern interactions between the model
1055 and tools, encouraging modular, domain-invariant
1056 planning and coherent multi-turn interactions. The
1057 template introduces three key elements:

- 1058 • **XML-based tool invocation:** Each tool call is
1059 enclosed within $\langle \text{tool_call} \rangle \cdot \langle / \text{tool_call} \rangle$,
1060 providing unambiguous syntax for learning
1061 and reducing hallucination in transfers.
- 1062 • **Dedicated reasoning block:** Reasoning is
1063 explicitly encouraged within a $\langle \text{think} \rangle \cdot$
1064 $\langle / \text{think} \rangle$ section, promoting structured, step-
1065 by-step derivation before tool usage, which
1066 elicits high-level strategies (e.g., problem de-
1067 composition) applicable across domains.
- 1068 • **Role-structured dialogue:** Interactions are
1069 segmented into rounds, denoted as

1070 [Round t] USER: u_t , ASSISTANT: a_t ,

1071 where tool responses are incorporated via a
1072 standardized TOOL: marker.

1073 This prompt template defines a clear syntax sep-
1074 arating reasoning, tool calls, and final responses, re-
1075 ducing redundancy and improving token efficiency
1076 across domains. By enforcing interpretable step-by-
1077 step reasoning within defined tags, it fosters struc-
1078 tured problem solving and narrows cross-domain
1079 generalization gaps. It also enhances coherence
1080 in multi-turn dialogues, enabling transferable rea-
1081 soning strategies—e.g., applying optimization be-
1082 haviors learned in math to scientific experiments.
1083 Finally, its consistent tool integration with dynamic
1084 cues supports robust evaluation of domain invari-
1085 ance and strengthens the agent’s adaptive tool-use
1086 across varied reasoning tasks. A complete example
1087 of the XML-based prompt template \mathcal{P} used in our
1088 experiments is provided in Appendix A.3.

B Case Study

1089

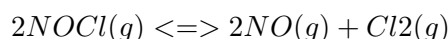
B.1 A case from Chemistry Problem Solving

1090

A case from Chemistry Problem Solving

Problem:

Nitrosyl chloride ($NOCl$) decomposes to nitric oxide (NO) and chlorine (Cl_2) when heated according to the reaction:



At $T = 600$ K, the equilibrium constant is $K_p = 0.060$. In a vessel at 600 K, a mixture of all three gases has the following partial pressures: $P_{NOCl} = 675$ torr, $P_{NO} = 43$ torr, $P_{Cl_2} = 23$ torr. What is the partial pressure of NO at equilibrium?

Solution:

ASSISTANT:

This is a chemical equilibrium problem involving the decomposition of nitrosyl chloride. I need to:

1. Determine the reaction direction by comparing Q (reaction quotient) with K_p
2. Set up an ICE table to track pressure changes
3. Solve for the equilibrium partial pressure of NO

The balanced equation is: $2NOCl(g) \rightarrow 2NO(g) + Cl_2(g)$

Given data:

- Temperature: 600 K
- $K_p = 0.060$
- Initial partial pressures: $P_{NOCl} = 675$ torr, $P_{NO} = 43$ torr, $P_{Cl_2} = 23$ torr

First, I'll calculate the reaction quotient Q to determine which direction the reaction proceeds.

Step 1: Calculate Reaction Quotient Q

Python Code

```
# Initial pressures (torr) and Kp
P_NOCl, P_NO, P_Cl2 = 675, 43, 23
Kp = 0.060

# Reaction quotient Q
Q = (P_NO**2 * P_Cl2)/(P_NOCl**2)
Q
```

Tool Output

```
0.09333772290809328
```

ASSISTANT: Since $Q_{initial} = 0.0933 > K_p = 0.060$, the reaction will shift in the reverse direction to reach equilibrium. This means some NO and Cl will combine to form more $NOCl$.

Step 2: Set up ICE Table and Solve

Let x be the extent of the reverse reaction. The changes in partial pressures will be:

- P_{NOCl} increases by $2x$

1091

- P_{NO} decreases by $2x$
- P_{Cl_2} decreases by x

At equilibrium:

$$P_{NOCl,eq} = 675 + 2x \quad (13)$$

$$P_{NO,eq} = 43 - 2x \quad (14)$$

$$P_{Cl_2,eq} = 23 - x \quad (15)$$

Substituting into the equilibrium expression:

$$K_p = \frac{(P_{NO,eq})^2 \cdot P_{Cl_2,eq}}{(P_{NOCl,eq})^2} = 0.060$$

This gives us:

$$\frac{(43 - 2x)^2(23 - x)}{(675 + 2x)^2} = 0.060$$

This is a cubic equation that requires numerical solution.

Python Code

```
import numpy as np
from scipy.optimize import fsolve

# Define the cubic equation coefficients
# Equation: 4x3 - 263.76x2 + 5967x - 15189.5 = 0
def func(x):
    return 4*x**3 - 263.76*x**2 + 5967*x - 15189.5

# Initial guess for x
initial_guess = 10 # Starting with a reasonable guess
solution = fsolve(func, initial_guess)

# Extract the valid solution (should be a real positive root)
x = solution[0]

x
```

Tool Output

2.901292604180679

ASSISTANT: Now I'll calculate the equilibrium partial pressures and verify the solution.

Step 3: Calculate Equilibrium Pressures and Verify

Python Code

```
# Calculated x
x = 2.901292604180679

# Equilibrium partial pressures
P_NOCl_eq = 675 + 2*x
P_NO_eq = 43 - 2*x
P_Cl2_eq = 23 - x

# Compute Kp
Kp_calculated = (P_NO_eq**2 * P_Cl2_eq) / (P_NOCl_eq**2)
Kp_calculated
```

Tool Output

```
0.059999999999999984
```

ASSISTANT: Perfect! The calculated $K_p \approx 0.060$ matches the given equilibrium constant, confirming our solution is correct.

Final Answer:

With $x = 2.901$ torr, the equilibrium partial pressure of NO is:

$$P_{NO,eq} = 43 - 2(2.901) = 37.2 \text{ torr}$$

Verification:

- All equilibrium pressures are positive
- K_p calculation matches the given value
- Reaction proceeds in the correct direction (reverse)

Therefore, the partial pressure of NO at equilibrium is 37.2 torr.

1093

B.2 A case from Mathematical Problem Solving

1094

A case from Mathematical Problem Solving

Problem:

Let N be the greatest four-digit positive integer with the property that whenever one of its digits is changed to 1, the resulting number is divisible by 7. Let Q and R be the quotient and remainder, respectively, when N is divided by 1000. Find $Q + R$.

Solution:

ASSISTANT: I need to find the largest four-digit number N such that changing any single digit to 1 results in a number divisible by 7.

Problem Analysis:

1. Find N : The largest four-digit number where changing any digit to 1 gives a multiple of 7
2. For a four-digit number $ABCD$, all of $1BCD$, $A1CD$, $AB1D$, $ABC1$ must be divisible by 7

1095

3. Once found, compute $Q = \lfloor N/1000 \rfloor$ and $R = N \bmod 1000$, then find $Q + R$

Approach: Since we want the greatest such number, I'll search from 9999 downward until finding a valid N .

Python Code

```
def find_n():
    for n in range(9999, 999, -1):
        s = str(n).zfill(4)
        valid = True
        for i in range(4):
            # Replace ith digit with '1'
            new_s = s[:i] + '1' + s[i+1:]
            m = int(new_s)
            if m % 7 != 0:
                valid = False
                break
        if valid:
            return n
    return None

N = find_n()
Q = N // 1000
R = N % 1000
result = Q + R
print(f"N = {N}, Q = {Q}, R = {R}, Result = {result}")
```

Error Output

```
SyntaxError: unexpected character after line continuation character
```

ASSISTANT: The error occurred due to JSON formatting issues with the f-string quotes. Let me fix this by using single quotes in the print statement to avoid conflicts with the JSON double quotes.

Python Code

```
def find_n():
    for n in range(9999, 999, -1):
        s = str(n).zfill(4)
        valid = True
        for i in range(4):
            new_s = s[:i] + '1' + s[i+1:]
            m = int(new_s)
            if m % 7 != 0:
                valid = False
                break
        if valid:
            return n
    return None

N = find_n()
Q = N // 1000
R = N % 1000
result = Q + R
print('N = {}, Q = {}, R = {}, Result = {}'.format(N, Q, R, result))
```

Successful Output

```
N = 5694, Q = 5, R = 694, Result = 699
```

ASSISTANT: Perfect! The algorithm found $N = 5694$. Let me verify this result by checking that changing each digit to 1 produces numbers divisible by 7.

Verification:

- Original number: 5694
- Change thousands digit (5→1): $1694 = 7 \times 242$
- Change hundreds digit (6→1): $5194 = 7 \times 742$
- Change tens digit (9→1): $5614 = 7 \times 802$
- Change units digit (4→1): $5691 = 7 \times 813$

All four modified numbers are indeed divisible by 7, confirming that $N = 5694$ is valid.

Final Calculation:

$$N = 5694 \quad (16)$$

$$Q = \lfloor 5694/1000 \rfloor = 5 \quad (17)$$

$$R = 5694 \bmod 1000 = 694 \quad (18)$$

$$Q + R = 5 + 694 = 699 \quad (19)$$

Therefore, $Q + R = \boxed{699}$.

B.3 A case from Financial Problem Solving

A case from Financial Mathematics Problem Solving

Problem:

A solid waste disposal company borrowed money at 10% per year interest to purchase new haulers and other equipment needed at the company-owned landfill site. If the company got the loan 2 years ago and paid it off with a single payment of \$4,600,000, what was the principal amount P of the loan?

Solution:

ASSISTANT:

This is a financial mathematics problem involving loan repayment calculations. I need to:

1. Determine whether to use simple or compound interest
2. Apply the appropriate formula to find the principal amount
3. Verify the calculation

Given data:

- Annual interest rate: 10%
- Loan duration: 2 years
- Single repayment amount: \$4,600,000

The problem states "10% per year interest" without specifying compounding. In financial contexts, this could be either simple or compound interest. Let me calculate both possibilities to determine the most appropriate approach.

Step 1: Calculate Principal Using Simple Interest

For simple interest: $A = P(1 + rt)$, where:

- $A = 4,600,000$ (total repayment)
- $r = 0.10$ (annual interest rate)
- $t = 2$ years

Solving for P : $P = \frac{A}{1+rt}$

Tool Call

```
# Calculating principal for simple interest
A_simple = 4600000
r_simple = 0.10
t = 2
P_simple = A_simple / (1 + r_simple * t)
P_simple
```

Tool Output

```
3833333.3333333335
```

ASSISTANT:

Step 2: Calculate Principal Using Compound Interest

For compound interest: $A = P(1 + r)^t$

Solving for P : $P = \frac{A}{(1+r)^t}$

Tool Call

```
# Calculating principal for compound interest
A_compound = 4600000
r_compound = 0.10
t = 2
P_compound = A_compound / (1 + r_compound)**t
P_compound
```

Tool Output

```
3801652.892561983
```

ASSISTANT:

Step 3: Analysis and Final Answer

The calculations yield different results:

- Simple interest: \$3,833,333.33
- Compound interest: \$3,801,652.89

Since the problem doesn't specify compounding and uses the phrase "10% per year interest," in most financial mathematics problems of this type, compound interest is the standard assumption unless explicitly stated as simple interest. This is because:

1. Real-world loans typically use compound interest
2. The 2-year duration suggests annual compounding
3. Financial institutions standard practice

Verification: Using compound interest: $P \times (1.10)^2 = 3,801,652.89 \times 1.21 = 4,600,000$

Therefore, the principal amount of the loan is $\boxed{\$3,801,652.89}$.

C Evaluation Benchmarks

Our comprehensive evaluation encompasses challenging benchmarks across mathematical and general domains:

Mathematical Reasoning Benchmarks:

- **MATH-500** (Hendrycks et al., 2021): A subset of 500 problems from the MATH dataset, featuring competition-level mathematical problems across algebra, geometry, number theory, and other mathematical domains. This benchmark requires sophisticated mathematical reasoning and problem-solving skills.
- **AIME 24 & AIME 25**: Problems from the American Invitational Mathematics Examination for 2024 and 2025, representing some of the most challenging high school mathematics competitions. These benchmarks test advanced mathematical reasoning capabilities and require precise computational skills.
- **HMMT 25**: Problems from the Harvard-MIT Mathematics Tournament 2025, featuring university-level mathematical challenges that demand deep mathematical insight and creative problem-solving approaches.

General Reasoning Benchmarks:

- **GPQA** (Rein et al., 2023): Graduate-level questions across physics, chemistry, and biology that require expert-level domain knowledge. We use the highest-quality GPQA-diamond subset, which contains carefully curated questions designed to be challenging even for domain experts.
- **TheoremQA** (Chen et al., 2023): A benchmark assessing models' ability to apply mathematical and scientific theorems to solve complex problems. It includes 800 high-quality questions covering 350 theorems from mathematics, physics, and other scientific domains, requiring both theoretical knowledge and practical application skills.
- **WebInstruct**: A held-out validation split from the WebInstruct dataset (Ma et al., 2025b), designed to evaluate multidisciplinary reasoning capabilities.