THINK BEYOND INSTANCES: ABSTRACTION AS MODELING FOR MATHEMATICAL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Mathematical reasoning is a fundamental capability of large language models (LLMs), requiring multi-step deduction and precise symbolic manipulation. Existing training-based and inference-time approaches improve reasoning performance but often rely on instance-specific solution traces, overlooking reusable computational structures shared across related problems and thereby limiting reasoning generalization. To address this limitation, we propose Think beyond Instances, a process-level reasoning abstraction framework that transforms instance-specific solutions into reusable logic-card chains. Our framework integrates multi-expert reasoning, logic-preserving reasoning abstraction that disentangles reusable logic from instance-specific realizations at each step, and chain-level validation to ensure global reasoning consistency before applying the validated chain to solve the original query. Experiments on six mathematical reasoning benchmarks demonstrate consistent improvements over competitive baselines. On Qwen3.5-9B, our framework achieves an average accuracy of 54.72%, outperforming the strongest baseline by 3.69 percentage points. Further analyses validate the effectiveness, transferability, and scalability of our framework. Our code is publicly available at an anonymous link `https://anonymous.4open.science/r/Think-beyond-Instances-123-3E7C`.

1 INTRODUCTION

Recent years have witnessed remarkable advances in large language models, demonstrating strong capabilities across a wide range of language understanding and generation tasks (Achiam et al., 2023; Bian et al., 2025). Despite these advances, LLMs still face significant challenges in complex mathematical reasoning, which demands multi-step deduction and rigorous symbolic manipulation (Chen et al., 2024a; Yang et al., 2025).

A growing body of work has sought to improve the mathematical reasoning capabilities of LLMs, yet robust generalization remains an open challenge. Existing approaches mainly fall into two categories: training-based and inference-time strategies. Training-based approaches enhance intrinsic reasoning capabilities through supervised fine-tuning (SFT) and reinforcement learning (RL). SFT-based methods primarily focus on constructing high-quality reasoning data, including question augmentation and rephrasing (Lu et al., 2024), large-scale problem synthesis (Ding et al., 2024; Li et al., 2026b), difficulty-aware synthesis (Li et al., 2026b), and long-chain reasoning curation (Xu et al., 2025; Yeo et al., 2025). RL-based methods further introduce process reward models for step-level supervision (Wang et al., 2024a) and policy optimization to refine reasoning behaviors (Shao et al., 2024; Yu et al., 2026; Dai et al., 2026). While effective, these approaches often require substantial training data and computational resources, particularly in RL settings. In contrast, inference-time strategies improve reasoning without modifying model parameters. Representative methods include step-by-step decomposition (Besta et al., 2024; Lin et al., 2026), self-refinement and verification (Shinn et al., 2023; Liang et al., 2026), and test-time scaling through sampling, search, and adaptive inference (Snell et al., 2024; Li et al., 2026a). However, most inference-time approaches remain tightly coupled to instance-specific reasoning traces and fail to explicitly capture reusable computational structures shared across logically equivalent problems. Consequently, they can be sensitive to surface-level variations, limiting robustness and generalization.

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.

Although recent efforts have incorporated abstraction into LLM reasoning, such as extracting underlying concepts or principles (Zheng et al., 2023), introducing high-level plans (Hong et al., 2024), discovering reusable reasoning abstractions (Qu et al., 2026), and abstracting concrete values into symbolic variables or templates (Gao et al., 2025; 2026; Lai et al., 2025), several key challenges remain. First, existing abstractions are often weakly aligned with the step-by-step reasoning process. Many methods derive abstract representations from problem statements or entire solution traces without explicitly grounding them in individual computational steps, limiting their ability to guide and constrain intermediate reasoning. Second, abstraction is typically performed in a single pass with limited validation of its generality and global consistency. Prior work rarely examines whether the abstracted logic transfers to logically equivalent but different problem instances or supports iterative correction of inconsistencies across reasoning steps. Consequently, the resulting abstractions may remain tied to specific instances rather than capturing reusable computational logic.

To address these challenges, we propose *Think beyond Instances*, a process-level abstraction framework that models mathematical reasoning by disentangling reusable logic from instance-specific realizations. Our framework abstracts step-by-step solutions into a reusable logic-card chain, where each card couples an Abstract Logic Expression (ALE) with its Concrete Grounding (CG). The validated logic-card chain is then applied to the original query to guide final answer generation. Specifically, we first introduce *Multi-Expert Reasoning*, where multiple reasoning experts generate complementary solutions that are integrated into a coherent preliminary reasoning process. We then perform *Logic-Preserving Reasoning Abstraction*, which first constructs and validates mutated queries that preserve the underlying computational logic while varying instance-specific realizations. Based on these variants, each reasoning step is abstracted into candidate ALE–CG logic cards, and a Best-Card Router iteratively selects the abstraction that best transfers beyond the original instance. Finally, *Chain-level Validation and Reasoning* audits the assembled logic-card chain for global consistency, routes problematic steps back for revision, and applies the validated chain to the original query to generate the framework answer.

Contributions. Our contributions are threefold. We introduce a process-level abstraction paradigm that reformulates mathematical solutions into reusable logic-card chains, enabling reasoning beyond individual problem instances. Second, we develop *Think beyond Instances*, a three-stage framework integrating multi-expert reasoning, logic-preserving reasoning abstraction, and chain-level validation and reasoning to construct, refine, and apply reusable reasoning structures beyond individual problem instances. Third, extensive experiments on six mathematical benchmarks demonstrate consistent improvements over competitive baselines, together with strong transferability and scalability.

2 METHODOLOGY

In this section, we introduce *Think beyond Instances*, a reasoning framework that abstracts instance-specific solutions into a reusable logic-card chain and solves the original query based on the validated chain. We present the overall framework and then describe each component in detail.

2.1 FRAMEWORK

As shown in Figure 1, *Think beyond Instances* follows a three-stage pipeline from diverse solution generation to logic abstraction and validated reasoning: (1) *Multi-Expert Reasoning.* Three reasoning experts independently solve the query from complementary perspectives. If all experts agree, the consensus answer is directly adopted; otherwise, their solutions are integrated into a preliminary reasoning process and a corresponding answer for subsequent abstraction. (2) *Logic-Preserving Reasoning Abstraction.* We generate a logic-preserving query variant, use it to guide abstraction, and abstract each reasoning step into candidate ALE–CG logic cards. A Best-Card Router selects the cards that best transfer to the mutated query, forming a reusable logic-card chain. (3) *Chain-level Validation and Reasoning.* The logic-card chain is validated for global consistency, with problematic steps routed back for revision. The validated chain is then used to solve the original query, and the resulting framework answer is further integrated with expert predictions to determine the final answer.

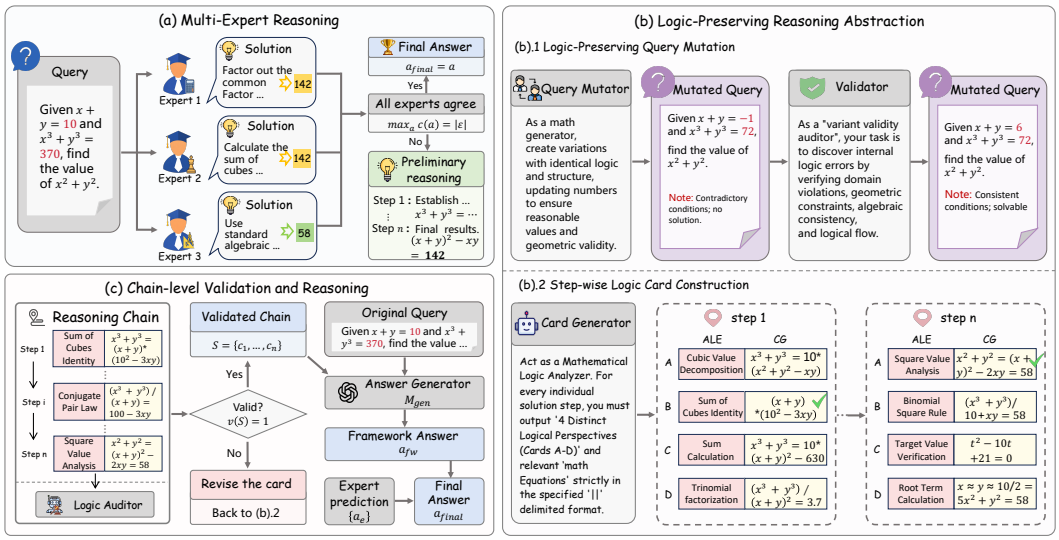


Figure 1: Overview of Think beyond Instances. Multi-expert reasoning, logic-preserving reasoning abstraction, and chain-level validation are integrated to construct a reusable logic-card chain and derive the final answer.

2.2 Multi-Expert Reasoning

The goal of this stage is to obtain diverse reasoning trajectories and construct a reliable preliminary reasoning process for subsequent logic abstraction. Given a mathematical query q, we introduce three LLM-based experts with different reasoning preferences, denoted as $\mathcal{E} = e_{\text{alg}}, e_{\text{str}}, e_{\text{geo}}$, corresponding to algebraic, strategic, and geometric reasoning, respectively. Each expert independently solves the query and generates a reasoning trajectory r_e with a candidate answer a_e, i.e., $e(q) \rightarrow (r_e, a_e)$.

We first evaluate the agreement among expert predictions. Let $c(a) = |\{e \in \mathcal{E} : a_e = a\}|$ denote the number of experts producing answer a. If all experts reach the same conclusion, i.e., $\max_a c(a) = |\mathcal{E}|$, the answer is directly adopted as the final prediction. Otherwise, we collect the expert solutions $\mathcal{R}_{\mathcal{E}} = \{(r_e, a_e)\}_{e\in\mathcal{E}}$ and feed them into a reasoning integration model M_{rea}, which synthesizes complementary expert insights into a unified preliminary reasoning process and the corresponding answer: $M_{\text{rea}}(q, \mathcal{R}_{\mathcal{E}}) \rightarrow (r^{\text{pre}}, a^*)$. The resulting r^{pre} is organized into step-wise reasoning units while preserving logical coherence and incorporating diverse expert perspectives. Together with the integrated answer a^*, it serves as the foundation for subsequent logic-preserving abstraction.

2.3 Logic-Preserving Reasoning Abstraction

The goal of this stage is to identify reusable logic structures underlying each reasoning step, disentangling generalizable reasoning patterns from instance-specific realizations. This stage consists of two components: *logic-preserving query mutation* and *step-wise logic card construction*.

Logic-Preserving Query Mutation. To evaluate whether the extracted reasoning structure captures generalizable logic rather than instance-specific information, we first construct an alternative problem instance with the same underlying computational structure. Given an original query q, an LLM-based mutator M_{mut} generates a mutated query q': $M_{\text{mut}} : q \rightarrow q'$. We decompose each query into an underlying computational logic $\mathcal{L}(\cdot)$ and an instance-specific realization $\mathcal{C}(\cdot)$. The goal of mutation is to preserve the underlying computational structure while modifying instance-specific realizations: $\mathcal{L}(q') \approx \mathcal{L}(q)$ and $\mathcal{C}(q') \neq \mathcal{C}(q)$. Here, $\mathcal{L}(\cdot)$ captures the reasoning structure required to solve the problem, while $\mathcal{C}(\cdot)$ contains query-specific information such as entities, numerical values, and surface-level descriptions. Thus, the mutated query aims to preserve the original computational logic while introducing different concrete realizations.

However, modifying concrete elements may introduce conflicts among constraints (e.g., "$x+y=-1$" and "$x^3+y^3=72$"), making q' inconsistent or unsatisfiable. We therefore introduce an LLM-based

validator M_{val}: $M_{\text{val}} : q' \mapsto v(q'), v(q') \in \{0, 1\}$, where $v(q')$ indicates whether q' constitutes a valid and satisfiable problem instance. Specifically, the validator examines logical consistency, domain validity, geometric realizability, and algebraic soundness. Only mutations satisfying $v(q') = 1$ are retained.

Step-wise Logic Card Construction. Given the reasoning process $r^{\text{pre}} = \{step_1, \ldots, step_n\}$, we abstract each reasoning step into a logic card, and concatenate the selected cards into a logic-card chain. Each logic card represents a generalized reasoning operation beyond the original instance through an ALE–CG pair. ALE captures the instance-independent logic of a reasoning step by removing query-specific entities and numerical values, while CG records its realization under the original query, including substitutions and intermediate computations. This representation disentangles reusable logic structures from instance-specific realizations, enabling transferable reasoning beyond a fixed textual solution.

For each reasoning step $step_i$, a Logic Card Synthesizer M_{syn} generates multiple candidate abstractions. Given the original query q, the current reasoning step $step_i$, the integrated answer a^*, and previously selected cards $S_{<i}$ (empty when $i = 1$), the synthesizer produces $C_i = \{(ALE_i^{(j)}, CG_i^{(j)})\}_{j=1}^{K}$. Among the candidate cards in C_i, a Best-Card Router M_{rou} selects the card c_i^* that best transfers to the mutated query q'. The selected ALE must preserve the reasoning operation required by q' while excluding numerical values or concrete entities inherited from the original query. If no candidate in C_i satisfies these constraints, the selection is rejected and the pipeline returns to the Logic Card Synthesizer. The candidate set is then regenerated using the original query q, the current step $step_i$, the integrated answer a^*, the previously selected cards $S_{<i}$, and the router feedback describing the causes of rejection. This generate–route–revise procedure progressively refines the abstraction until a valid card is obtained.

2.4 Chain-level Validation and Reasoning

After selecting a logic card for each reasoning step, the cards are concatenated into a complete reasoning chain $S = \{c_1^*, \ldots, c_n^*\}$. We then perform chain-level logical validation to ensure that locally valid abstractions also form a globally coherent reasoning process. A logic auditor M_{aud} evaluates consistency between premises and derived conclusions, validity of intermediate transformations, and coherence across reasoning steps, producing $v(S) \in \{0, 1\}$. When $v(S) = 0$, the auditor identifies the problematic step and routes it back to the corresponding card-selection stage for revision. Only chains satisfying $v(S) = 1$ are retained. To bound test-time computation, we allow at most one such rollback. If the revised chain still fails validation, we terminate the abstraction pipeline and fall back to the initial algebraic expert prediction a_{alg} as the final answer.

Given a validated logic-card chain S, an answer generator M_{gen} executes the abstract reasoning structure on the original query to obtain a framework answer: $M_{\text{gen}}(q, S) \to a_{\text{fw}}$. We then reconcile a_{fw} with the expert predictions $\{a_e\}_{e \in \mathcal{E}}$. If a_{fw} agrees with any expert prediction, it is selected as the final answer. Otherwise, if multiple experts agree on the same answer, the majority prediction is adopted. When all experts disagree, the framework answer is retained as the final prediction, i.e., $a_{\text{final}} = a_{\text{fw}}$.

3 Experiments

In this section, we conduct extensive experiments to evaluate the proposed framework and provide detailed analyses to demonstrate its effectiveness.

3.1 Experimental Setup

Benchmarks and metrics. We evaluate our method on six widely used mathematical benchmarks: MATH500 (Lightman et al., 2024), Minerva Math (Lewkowycz et al., 2022), OlympiadBench (He et al., 2024), College Math (Tang et al., 2024), AMC23,[1] and AIME25.[2] For all benchmarks except AMC23 and AIME25, we report pass@1, averaged over three independent runs. Due to the small

[1] https://huggingface.co/datasets/math-ai/amc23

[2] https://huggingface.co/datasets/math-ai/aime25

Table 1: Performance comparison (%) on mathematical benchmarks with Qwen-family base models. Best and second-best results are bolded and underlined, respectively. "Improvement" denotes the gain over the strongest baseline.

Models	MATH500	Min. Math	Olympiad	Col. Math	AMC23	AIME25	AVG
Qwen3-4B	69.0	12.5	31.3	30.3	45.5	8.9	32.92
LaTRO	67.8	12.5	31.6	29.6	47.1	8.2	32.80
GRPO	73.8	15.8	34.4	34.2	50.9	7.9	36.17
AoT	73.2	16.8	36.8	34.6	51.2	10.3	37.15
RAFT++	73.6	15.8	34.7	34.0	52.5	8.3	36.48
AdaR	73.6	15.4	37.6	33.8	53.7	11.3	37.57
GPG	74.6	13.6	34.1	34.1	49.8	7.7	35.65
InTRO	74.8	17.6	39.4	35.1	58.3	12.6	39.63
IC-DAPO	75.6	19.4	40.0	36.4	58.4	14.7	40.75
Ours	**79.3**	**23.9**	**44.5**	**41.2**	**62.5**	**16.7**	**44.68**
Improvement	3.7 ↑	4.5 ↑	4.5 ↑	4.8 ↑	4.1 ↑	2.0 ↑	3.93 ↑
Qwen3-8B	65.8	11.8	34.7	29.8	53.4	10.0	34.25
LaTRO	70.0	12.9	33.5	29.3	52.7	11.3	34.95
GRPO	74.4	14.3	36.3	33.1	55.8	10.9	37.47
AoT	74.8	16.4	36.5	34.2	54.7	11.3	37.98
RAFT++	72.6	12.5	36.1	31.1	55.4	10.9	36.43
AdaR	74.2	17.8	37.8	33.9	55.2	11.8	38.45
GPG	70.4	15.4	33.8	31.4	53.5	10.1	35.77
InTRO	75.2	18.8	38.7	35.2	56.7	12.4	39.50
IC-DAPO	76.6	20.7	39.9	36.7	57.4	15.3	41.10
Ours	**81.4**	**24.5**	**44.2**	**40.4**	**58.8**	**19.8**	**44.85**
Improvement	4.8 ↑	3.8 ↑	4.3 ↑	3.7 ↑	1.4 ↑	4.5 ↑	3.75 ↑
Qwen3.5-4B	73.4	19.6	37.2	42.0	61.5	12.6	41.05
LaTRO	75.0	21.0	39.0	43.5	63.0	13.8	42.55
GRPO	77.4	25.6	42.0	46.0	66.2	15.4	45.43
AoT	78.2	24.1	45.6	45.5	64.7	15.2	45.55
RAFT++	76.4	22.7	40.8	45.1	65.0	14.7	44.12
AdaR	76.8	23.8	43.8	42.3	67.6	16.8	45.18
GPG	77.0	23.5	41.6	45.7	65.8	15.0	44.77
IC-DAPO	78.7	25.8	45.8	46.2	67.4	17.4	46.88
Ours	**83.2**	**31.4**	**47.4**	**51.0**	**70.5**	**20.6**	**50.68**
Improvement	4.5 ↑	5.6 ↑	1.6 ↑	4.8 ↑	2.9 ↑	3.2 ↑	3.80 ↑
Qwen3.5-9B	76.5	24.2	41.8	46.5	67.0	15.2	45.20
LaTRO	78.1	25.8	43.6	48.0	68.4	16.4	46.72
GRPO	80.3	28.4	46.2	50.3	71.0	18.5	49.12
AoT	81.1	28.7	46.7	51.7	69.8	18.2	49.37
RAFT++	79.4	27.2	45.0	49.4	70.0	17.6	48.10
AdaR	79.8	30.0	48.1	49.8	72.5	20.1	50.05
GPG	80.0	27.9	45.8	50.0	70.7	18.0	48.73
IC-DAPO	81.6	30.7	48.4	51.9	72.9	20.7	51.03
Ours	**86.3**	**35.6**	**50.6**	**55.4**	**76.0**	**24.4**	**54.72**
Improvement	4.7 ↑	4.9 ↑	2.2 ↑	3.5 ↑	3.1 ↑	3.7 ↑	3.69 ↑

test set sizes of AMC23 and AIME25 (40 and 30 problems, respectively), we report avg@32 (Zhu et al., 2026), which yields a more stable performance estimate than pass@1 under limited sample sizes. To ensure a fair evaluation of the model's intrinsic reasoning ability, the evaluation protocol is kept consistent with the training setup and does not introduce additional post-processing procedures such as numerical extraction or template-based answer normalization.

Baselines. We compare our framework against a diverse set of strong and representative baselines, including LaTRO (Chen et al., 2024b), GRPO (Shao et al., 2024), AoT (Hong et al., 2024), RAFT++ (Xiong et al., 2025), AdaR (Lai et al., 2025), GPG (Chu et al., 2026), InTRO (Zhu et al., 2026), and IC-DAPO (Mei et al., 2026). Further details of these baselines are provided in Appendix A.

Implementation details. In the main experiments, we adopt models from the Qwen family as base LLMs, including Qwen3-4B/8B and Qwen3.5-4B/9B. Within each experimental setting, all framework modules, including M_{rea}, M_{mut}, M_{val}, M_{syn}, M_{rou}, M_{aud}, and M_{gen}, are instantiated with

Table 2: Ablation study of our framework. Expert Voting retains only the multi-expert voting strategy, while the other variants remove individual components. "rb" denotes the rollback mechanism. Numbers in parentheses indicate the performance drop relative to Ours.

Models	MATH500	Min. Math	Olympiad	Col. Math	AMC23	AIME25	AVG
Qwen3-8B	65.8	11.8	34.7	29.8	53.4	10.0	34.25
Expert Voting	76.9 (4.5)	20.4 (4.1)	39.7 (4.5)	36.5 (3.9)	56.8 (2.0)	15.3 (4.5)	40.93 (3.92)
w/o LPQM	78.6 (2.8)	22.1 (2.4)	41.6 (2.6)	38.0 (2.4)	57.6 (1.2)	17.1 (2.7)	42.50 (2.35)
w/o SWCC	78.1 (3.3)	21.6 (2.9)	41.2 (3.0)	37.6 (2.8)	57.3 (1.5)	16.8 (3.0)	42.10 (2.75)
w/o BCR_rb	79.5 (1.9)	23.0 (1.5)	42.7 (1.5)	39.0 (1.4)	58.0 (0.8)	18.4 (1.4)	43.43 (1.42)
w/o CLV	79.8 (1.6)	23.3 (1.2)	43.0 (1.2)	39.3 (1.1)	58.2 (0.6)	18.7 (1.1)	43.72 (1.13)
Ours	81.4	24.5	44.2	40.4	58.8	19.8	44.85
Qwen3.5-9B	76.5	24.2	41.8	46.5	67.0	15.2	45.20
Expert Voting	80.8 (5.5)	27.8 (7.8)	45.0 (5.6)	49.2 (6.2)	70.5 (5.5)	17.3 (7.1)	48.43 (6.28)
w/o LPQM	82.9 (3.4)	31.4 (4.2)	47.2 (3.4)	51.6 (3.8)	72.8 (3.2)	20.5 (3.9)	51.07 (3.65)
w/o SWCC	82.5 (3.8)	30.9 (4.7)	46.8 (3.8)	51.2 (4.2)	72.3 (3.7)	20.1 (4.3)	50.63 (4.08)
w/o BCR_rb	84.6 (1.7)	33.7 (1.9)	49.0 (1.6)	53.5 (1.9)	74.4 (1.6)	22.5 (1.9)	52.95 (1.77)
w/o CLV	85.0 (1.3)	34.1 (1.5)	49.5 (1.1)	53.9 (1.5)	74.8 (1.2)	23.0 (1.4)	53.38 (1.33)
Ours	86.3	35.6	50.6	55.4	76.0	24.4	54.72
Llama3.1-8B	37.0	17.3	13.0	12.3	21.0	0.9	16.92
Expert Voting	38.0 (11.8)	11.8 (8.4)	14.4 (7.6)	17.8 (24.7)	22.6 (7.4)	0.0 (3.3)	17.43 (10.54)
w/o LPQM	44.6 (5.2)	17.2 (3.0)	16.3 (5.7)	28.9 (13.6)	23.8 (6.2)	0.0 (3.3)	21.80 (6.17)
w/o SWCC	43.8 (6.0)	17.6 (2.6)	16.9 (5.1)	28.5 (14.0)	23.0 (7.0)	0.0 (3.3)	21.63 (6.34)
w/o BCR_rb	46.4 (3.4)	18.3 (1.9)	19.3 (2.7)	36.8 (5.7)	26.4 (3.6)	0.0 (3.3)	24.53 (3.44)
w/o CLV	45.8 (4.0)	18.7 (1.5)	18.9 (3.1)	35.4 (7.1)	25.6 (4.4)	0.0 (3.3)	24.07 (3.90)
Ours	49.8	20.2	22.0	42.5	30.0	3.3	27.97

the same base model. We set the number of candidate logic cards to $K = 4$ in Section 2.3. Unless otherwise specified, decoding uses a temperature of 0.6 and top-p of 0.95. For query mutation in Section 2.3, we increase the temperature of M_{mut} to 0.8 to encourage more diverse mutations. All experiments are conducted on a high-performance computing cluster equipped with eight NVIDIA H200 GPUs, each with 141 GB of memory.

3.2 Main Results

We evaluate our framework on six mathematical reasoning benchmarks using Qwen3-4B/8B and Qwen3.5-4B/9B as base models, and compare it against several competitive baselines. The results are reported in Table 1.

From the results, we observe that: 1) Our framework consistently achieves the best performance across all model settings. For example, with Qwen3-4B as the backbone, our method achieves an average score of 44.68%, outperforming the strongest baseline IC-DAPO (40.75%) by 3.93 percentage points. Similar improvements are observed across other backbone models, demonstrating the effectiveness of our framework in enhancing mathematical reasoning. 2) Our framework generalizes well across different backbone models and scales. Specifically, the improvements over the strongest baselines reach 3.93, 3.75, 3.80, and 3.69 percentage points for Qwen3-4B, Qwen3-8B, Qwen3.5-4B, and Qwen3.5-9B, respectively. The consistent gains across different backbone versions and model sizes demonstrate the robustness of our approach. 3) Our framework shows clear advantages over existing reasoning abstraction approaches. Compared with AoT and AdaR, our method achieves substantial improvements across all backbone settings. For instance, with Qwen3-4B, our framework improves the average score by 7.53 and 7.11 percentage points over AoT and AdaR, respectively. These results highlight the effectiveness of step-wise logic abstraction, which captures reusable reasoning structures beyond high-level planning or coarse-grained abstraction. 4) Our framework achieves consistent improvements on challenging mathematical reasoning benchmarks. On AIME25, our method surpasses the strongest baseline by 2.0–4.5 percentage points across different backbone models, while substantial gains of 3.8–5.6 percentage points are also observed on Min. Math. These results demonstrate the effectiveness of our approach in solving complex problems that require multi-step reasoning.

Table 3: Comparison with self-consistency under the same inference budget.

Methods	MATH500	Min. Math	Olympiad	Col. Math	AMC23	AIME25	AVG
			Qwen3-4B				
self-consistency	72.5	15.5	35.5	33.5	50.5	11.5	36.50
Ours	79.3	23.9	44.5	41.2	62.5	16.7	44.68
			Qwen3-8B				
self-consistency	69.2	13.8	37.5	33.6	55.3	13.3	37.12
Ours	81.4	24.5	44.2	40.4	58.8	19.8	44.85
			Qwen3.5-4B				
self-consistency	77.4	23.0	41.6	46.4	66.0	16.1	45.08
Ours	83.2	31.4	47.4	51.0	70.5	20.6	50.68
			Qwen3.5-9B				
self-consistency	79.9	26.7	45.3	50.0	70.4	18.2	48.42
Ours	86.3	35.6	50.6	55.4	76.0	24.4	54.72

3.3 Ablation Study

To assess the contribution of each component in our framework, we conduct a series of ablation experiments. Specifically, we consider the following variants: (1) retaining only the expert voting strategy; (2) removing Logic-Preserving Query Mutation (LPQM); (3) removing Step-wise Logic Card Construction (SWCC); (4) disabling the rollback mechanism in the Best-Card Router (BCR_rb); and (5) removing Chain-level Logical Validation (CLV). For variants (2)–(5), each component is removed individually while keeping the remaining components unchanged. The results are reported in Table 2.

Overall, all ablation variants consistently degrade performance across Qwen3-8B, Qwen3.5-9B, and Llama3.1-8B, demonstrating the contribution of each component. In particular, retaining only Expert Voting leads to average performance drops of 3.92, 6.28, and 10.54 percentage points, respectively. This indicates that the gains of our framework cannot be attributed to multi-expert aggregation alone, but also rely on the subsequent reasoning abstraction and validation process. Among the individual components, removing LPQM or SWCC causes the largest degradation. Removing LPQM decreases the average performance by 2.35, 3.65, and 6.17 points across the three backbones, while removing SWCC results in drops of 2.75, 4.08, and 6.34 points, respectively. The consistently larger degradation caused by removing SWCC highlights the importance of step-wise logic abstraction, while the substantial effect of LPQM confirms the value of logic-preserving instance variation for extracting reusable reasoning structures beyond individual instances. BCR_rb and CLV also provide consistent gains across all backbone models. Disabling rollback in the Best-Card Router results in average drops of 1.42, 1.77, and 3.44 points, while removing CLV causes drops of 1.13, 1.33, and 3.90 points. These results suggest that iterative card revision and chain-level validation further improve the robustness and logical consistency of the constructed reasoning chains.

3.4 Detailed Analysis

Comparison with Self-Consistency. To examine whether the performance gains of our framework simply stem from increased inference computation, we compare it with Self-Consistency under the same number of forward passes. Specifically, we first measure the average number of LLM calls required by our framework for each sample, and then allocate the same inference budget to Self-Consistency by generating an equivalent number of independent reasoning paths and aggregating their predictions. As shown in Table 3, our framework consistently outperforms Self-Consistency across all benchmarks and backbone models, improving the average score from 36.50 to 44.68 on Qwen3-4B, from 37.12 to 44.85 on Qwen3-8B, from 45.08 to 50.68 on Qwen3.5-4B, and from 48.42 to 54.72 on Qwen3.5-9B. These consistent gains under a matched inference budget indicate that the improvements arise not merely from additional computation, but from the structured reasoning abstraction and validation introduced by our framework.

Table 4: Performance comparison (%) between original and mutated queries using the same logic-card reasoning chain. Smaller gaps indicate better preservation of the underlying computational logic.

Model	MATH500		Min. Math		Olympiad		Col. Math		AMC23		AIME25		AVG	
	q	q'	q	q'	q	q'	q	q'	q	q'	q	q'	q	q'
Qwen3-8B	81.4	86.8	24.5	27.9	44.2	45.8	40.4	42.7	58.8	61.2	19.8	19.8	44.85	47.37
Qwen3.5-9B	86.3	88.0	35.6	37.8	50.6	52.1	55.4	57.2	76.0	77.5	24.4	26.7	54.72	56.55
Llama3.1-8B	49.8	52.4	20.2	24.4	22.0	23.4	42.5	44.7	30.0	31.8	3.3	6.7	27.97	30.57

Table 5: Cross-model transferability of logic-card reasoning chains. Gemma2-2B serves as the target model, with chains generated by different source models. Numbers in parentheses denote gains over Gemma2-2B without external reasoning chains.

Models	MATH500	Min. Math	Olympiad	Col. Math	AMC23	AIME25	AVG
Gemma2-2B	12.8	13.6	8.1	27.8	24.1	0.0	14.40
+ Qwen3-8B	29.3 (16.5)	16.4 (2.8)	16.5 (8.4)	37.9 (10.1)	33.6 (9.5)	3.3 (3.3)	22.83 (8.43)
+ Qwen3.5-9B	31.8 (19.0)	17.2 (3.6)	16.3 (8.2)	40.0 (12.2)	36.4 (12.3)	3.3 (3.3)	24.17 (9.77)
+ Llama3.1-8B	21.9 (9.1)	15.2 (1.6)	13.4 (5.3)	32.3 (4.5)	28.1 (4.0)	0.0 (0.0)	18.48 (4.08)

Table 6: Performance of our framework across models with different parameter sizes. Numbers in parentheses indicate the performance gains relative to the base models.

Models	MATH500	Min. Math	Olympiad	Col. Math	AMC23	AIME25	AVG
Qwen3-4B	69.0	12.5	31.3	30.3	45.5	8.9	32.92
+ Ours	79.3 (10.3)	23.9 (11.4)	44.5 (13.2)	41.2 (10.9)	62.5 (17.0)	16.7 (7.8)	44.68 (11.76)
Qwen3-8B	65.8	11.8	34.7	29.8	53.4	10.0	34.25
+ Ours	81.4 (15.6)	24.5 (12.7)	44.2 (9.5)	40.4 (10.6)	58.8 (5.4)	19.8 (9.8)	44.85 (10.60)
Qwen3-14B	69.2	27.0	52.7	44.4	62.5	36.7	48.75
+ Ours	76.0 (6.8)	33.1 (6.1)	59.1 (6.4)	47.7 (3.3)	75.0 (12.5)	43.3 (6.6)	55.70 (6.95)
Qwen3-32B	67.6	26.1	51.1	47.0	57.5	50.0	49.88
+ Ours	71.6 (4.0)	30.5 (4.4)	53.6 (2.5)	53.0 (6.0)	67.5 (10.0)	50.0 (0.0)	54.37 (4.49)

Reasonableness of query mutation. To assess the validity of the generated query mutations, we consider the following hypothesis: under a fixed logic-card reasoning chain, if a model exhibits similar performance on the original query q and its mutated variant q', the mutation is considered reasonable. In our evaluation, we first apply the proposed framework with Qwen3-8B, Qwen3.5-9B, or Llama3.1-8B as the base model to produce logic-card reasoning chains and responses for the original queries. We then pair each mutated query with the reasoning chain derived from its corresponding original query and provide both as input to the same base model to generate responses. The results are summarized in Table 4. Overall, the performance on the original and mutated queries remains comparable under the same logic-card reasoning chain. Averaged over six benchmarks, the performance gaps are only 2.5, 1.8, and 2.6 percentage points for Qwen3-8B, Qwen3.5-9B, and Llama3.1-8B, respectively. These relatively small gaps indicate that the reasoning chains derived from the original queries remain transferable to their mutated variants, suggesting that the mutations preserve the transferable computational structure required by the reasoning process while introducing different instance-specific realizations.

Cross-model transferability of logic-card chains. To evaluate the cross-model transferability of the generated logic-card reasoning chains, we use Gemma2-2B as the target model and provide it with chains generated by different source models. Specifically, we conduct an A/B comparison: in the A setting, Gemma2-2B directly solves each query, while in the B setting, the original query together with a logic-card reasoning chain generated by Qwen3-8B, Qwen3.5-9B, or Llama3.1-8B is provided to Gemma2-2B for answer generation. The results are reported in Table 5. As shown in the table, incorporating the generated logic-card reasoning chains improves Gemma2-2B across nearly all benchmarks. The average gains reach 8.43, 9.77, and 4.08 percentage points when the chains are generated by Qwen3-8B, Qwen3.5-9B, and Llama3.1-8B, respectively. Notably, chains generated by Qwen3.5-9B yield the largest improvement, increasing the average performance from 14.40% to 24.17%. These results indicate that the reasoning chains generated by our framework can

be effectively transferred across models and provide useful reasoning guidance to a weaker target model.

Scalability across different model scales. To evaluate the scalability of our framework, we apply it to Qwen3 models with different parameter scales, including Qwen3-4B, Qwen3-8B, Qwen3-14B, and Qwen3-32B. The results are summarized in Table 6. Our framework consistently improves the overall performance across all model scales, with average gains of 11.76, 10.60, 6.95, and 4.49 percentage points, respectively. The performance gain decreases as model scale increases, suggesting that smaller models benefit more substantially from the proposed reasoning augmentation. The improvement on AIME25 vanishes for Qwen3-32B. Nevertheless, even at the largest scale, our framework still provides a clear overall improvement, demonstrating its effectiveness across different model sizes.

4 Related Work

Mathematical reasoning with LLMs. Mathematical reasoning is a key capability of LLMs, requiring multi-step deduction and precise numerical computation. At inference time, many methods explicitly elicit intermediate reasoning. CoT prompting (Wei et al., 2022) improves multi-step reasoning by eliciting intermediate steps, while Program-of-Thought (PoT) (Chen et al., 2022) generates executable programs for numerical computation. Tree-of-Thought (ToT) (Yao et al., 2023) and Graph-of-Thought (GoT) (Besta et al., 2024) further enable structured exploration and backtracking over reasoning paths. Other approaches improve reliability through iterative refinement or verification, including Self-Consistency (Wang et al., 2022) and Clustering and Experience-based Verification (Liang et al., 2026). rStar-Math (Guan et al., 2025) further incorporates search-based strategies to explore mathematical solution spaces. More recent work also explores efficient management of inference-time compute through sub-question-level budgeting (Lin et al., 2026) and adaptive early stopping (Li et al., 2026a). Training-based approaches mainly rely on supervised fine-tuning (SFT) and reinforcement learning (RL). In SFT, mathematical reasoning performance depends on the availability and quality of supervised signals, and prior work has explored data synthesis to provide such supervision, including large-scale generation (Toshniwal et al., 2024), and difficulty-aware synthesis (Li et al., 2026b). MAmmoTH (Yue et al., 2023) combines CoT and PoT-style rationales, while s1 (Muennighoff et al., 2025) and LIMO (Ye et al., 2025) show that a small number of high-quality long CoT examples can be effective. In RL-based approaches, step-level feedback is commonly introduced via process reward models (PRMs) (Yuan et al., 2024), combined with policy optimization algorithms such as GRPO (Shao et al., 2024), DAPO (Yu et al., 2026), and DGPO (Dai et al., 2026).

Abstraction reasoning. Abstraction removes surface-level details while preserving underlying structure, and has been increasingly explored to improve generalization and robustness in LLM reasoning. In mathematics, Chain-of-Abstraction (CoA) (Gao et al., 2025) separates high-level reasoning from arithmetic execution using placeholders instantiated by external tools. NSDG (Li et al., 2024) converts queries into SMT-LIB representations for symbolic mutation and data augmentation. AdaR (Lai et al., 2025) synthesizes logically equivalent queries through variable variations and abstraction, and employs RLVR to discourage spurious logic. AbstRaL (Gao et al., 2026) uses reinforcement learning to transform concrete queries into symbolic abstractions for surface-invariant reasoning, while RLAD (Qu et al., 2026) learns reusable high-level procedural or factual abstractions to guide subsequent solution generation. Beyond mathematics, Step-Back Prompting (Zheng et al., 2023) extracts high-level principles to guide reasoning, Abstraction-of-Thought (AoT) (Hong et al., 2024) generates abstract sketches before detailed reasoning, and AbsInstruct (Wang et al., 2024b) improves abstraction ability by finetuning on abstraction relations (e.g., "apple"→"fruit").

5 Conclusion

We propose Think beyond Instances, a process-level abstraction framework that transforms instance-specific solutions into reusable logic-card chains. It consists of three stages: Multi-Expert Reasoning for constructing preliminary solutions, Logic-Preserving Reasoning Abstraction for extracting reusable step-wise logic, and Chain-level Validation and Reasoning for ensuring global consistency and solving the original query. Experiments on six benchmarks demonstrate consistent improvements, while further analyses validate the effectiveness, transferability, and scalability of our framework.

AI USE STATEMENT

Generative AI tools were used to assist with language polishing, improving the clarity and readability of the manuscript, standardizing mathematical notation, and debugging and testing research code. The research ideas, experimental design, analysis, and scientific conclusions were developed by the authors. All AI-assisted materials were carefully reviewed by the authors, who take full responsibility for the final scientific content and reported results.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 17682–17690, 2024.

Yuang Bian, Yupian Lin, Jingping Liu, and Tong Ruan. Ptoco: Prefix-based token-level collaboration enhances reasoning for multi-llms. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 8326–8335, 2025.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. Step-level value preference optimization for mathematical reasoning. *arXiv preprint arXiv:2406.10858*, 2024a.

Haolin Chen, Yihao Feng, Zuxin Liu, Weiran Yao, Akshara Prabhakar, Shelby Heinecke, Ricky Ho, Phil Mui, Silvio Savarese, Caiming Xiong, et al. Language models are hidden reasoners: Unlocking latent reasoning capabilities via self-rewarding. *arXiv preprint arXiv:2411.04282*, 2024b.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*, 2022.

Xiangxiang Chu, Hailang Huang, Xiao Zhang, Fei Wei, and Yong Wang. Gpg: A simple and strong reinforcement learning baseline for model reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 59637–59659, 2026.

Yanqi Dai, Yuxiang Ji, Xiao Zhang, Yong Wang, Xiangxiang Chu, and Zhiwu Lu. Harder is better: Boosting mathematical reasoning via difficulty-aware grpo and multi-aspect question reformulation. *arXiv preprint arXiv:2601.20614*, 2026.

Yuyang Ding, Xinyu Shi, Juntao Li, Qiaoming Zhu, Min Zhang, et al. Unleashing reasoning capability of llms via scalable question synthesis from scratch. 2024.

Silin Gao, Jane Dwivedi-Yu, Ping Yu, Xiaoqing Ellen Tan, Ramakanth Pasunuru, Olga Golovneva, Koustuv Sinha, Asli Celikyilmaz, Antoine Bosselut, and Tianlu Wang. Efficient tool use with chain-of-abstraction reasoning. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 2727–2743, 2025.

Silin Gao, Antoine Bosselut, Samy Bengio, and Emmanuel Abbe. Abstral: Augmenting llms' reasoning by reinforcing abstract thinking. In *International Conference on Learning Representations*, volume 2026, pp. 116757–116780, 2026.

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

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, 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.

Ruixin Hong, Hongming Zhang, Xiaoman Pan, Dong Yu, and Changshui Zhang. Abstraction-of-thought makes language models better reasoners. *arXiv preprint arXiv:2406.12442*, 2024.

Zhejian Lai, Xiang Geng, Zhijun Wang, Yang Bai, Jiahuan Li, Rongxiang Weng, Jingang Wang, Xuezhi Cao, Xunliang Cai, and Shujian Huang. Making mathematical reasoning adaptive. *arXiv preprint arXiv:2510.04617*, 2025.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

Jiakun Li, Xingwei He, Kefan Li, Hongzheng Chai, Hongyue Yu, and Yuan Yuan. Efficient test-time scaling via temporal reasoning aggregation. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 13304–13318, 2026a.

Xuchen Li, Jing Chen, Xuzhao Li, Hao Liang, Xiaohuan Zhou, Taifeng Wang, and Wentao Zhang. Mathmixup: Boosting llm mathematical reasoning with difficulty-controllable data synthesis and curriculum learning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 2497–2513, 2026b.

Zenan Li, Zhi Zhou, Yuan Yao, Xian Zhang, Yu-Feng Li, Chun Cao, Fan Yang, and Xiaoxing Ma. Neuro-symbolic data generation for math reasoning. *Advances in Neural Information Processing Systems*, 37:23488–23515, 2024.

Zhenwen Liang, Ruosen Li, Yujun Zhou, Linfeng Song, Dian Yu, Xinya Du, Haitao Mi, and Dong Yu. Your reasoning model is secretly a reward model-optimization-free verification from experience. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 17358–17372, 2026.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Junhong Lin, Xinyue Zeng, Jie Zhu, Song Wang, Julian Shun, Jun Wu, and Dawei Zhou. Plan and budget: Effective and efficient test-time scaling on reasoning large language models. In *International Conference on Learning Representations*, volume 2026, pp. 106869–106893, 2026.

Zimu Lu, Aojun Zhou, Houxing Ren, Ke Wang, Weikang Shi, Junting Pan, Mingjie Zhan, and Hongsheng Li. Mathgenie: Generating synthetic data with question back-translation for enhancing mathematical reasoning of llms. *arXiv preprint arXiv:2402.16352*, 2024.

Tiehua Mei, Minxuan Lv, Leiyu Pan, Zhenpeng Su, Hongru Hou, Hengrui Chen, Ao Xu, and Deqing Yang. Good reasoning makes good demonstrations: Implicit reasoning quality supervision via in-context reinforcement learning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 30452–30469, 2026.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20286–20332, 2025.

Yuxiao Qu, Anikait Singh, Yoonho Lee, Amrith Setlur, Russ Salakhutdinov, Chelsea Finn, and Aviral Kumar. Rlad: Training llms to discover abstractions for solving reasoning problems. In *International Conference on Learning Representations*, volume 2026, pp. 147993–148014, 2026.

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

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36:8634–8652, 2023.

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.

Zhengyang Tang, Xingxing Zhang, Benyou Wang, and Furu Wei. Mathscale: Scaling instruction tuning for mathematical reasoning. *arXiv preprint arXiv:2403.02884*, 2024.

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*, 2024.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, 2024a.

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*, 2022.

Zhaowei Wang, Wei Fan, Qing Zong, Hongming Zhang, Sehyun Choi, Tianqing Fang, Xin Liu, Yangqiu Song, Ginny Wong, and Simon See. Absinstruct: Eliciting abstraction ability from llms through explanation tuning with plausibility estimation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 973–994, 2024b.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang, Tong Zhang, Caiming Xiong, et al. A minimalist approach to llm reasoning: from rejection sampling to reinforce. *arXiv preprint arXiv:2504.11343*, 2025.

Haotian Xu, Xing Wu, Weinong Wang, Zhongzhi Li, Da Zheng, Boyuan Chen, Yi Hu, Shijia Kang, Jiaming Ji, Yingying Zhang, et al. Redstar: Does scaling long-cot data unlock better slow-reasoning systems? *arXiv preprint arXiv:2501.11284*, 2025.

Wen Yang, Minpeng Liao, and Kai Fan. Markov chain of thought for efficient mathematical reasoning. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7132–7157, 2025.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. Limo: Less is more for reasoning. *arXiv preprint arXiv:2502.03387*, 2025.

Edward Yeo, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. Demystifying long chain-of-thought reasoning in llms. *arXiv preprint arXiv:2502.03373*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *Advances in Neural Information Processing Systems*, 38:113222–113244, 2026.

Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. Free process rewards without process labels. *arXiv preprint arXiv:2412.01981*, 2024.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. Mammoth: Building math generalist models through hybrid instruction tuning. *arXiv preprint arXiv:2309.05653*, 2023.

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*, 2023.

Mingye Zhu, Yi Liu, Zheren Fu, Quan Wang, and Yongdong Zhang. In-token rationality optimization: Towards accurate and concise llm reasoning via self-feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 35195–35203, 2026.

A BASELINES

We provide descriptions of the baseline methods used for comparison below.

LaTRO (Chen et al., 2024b) trains LLMs to strengthen multi-step reasoning by optimizing an underlying latent reasoning space, without relying on reward signals or external supervision.

GRPO (Shao et al., 2024) is an RL post-training method derived from PPO that improves mathematical reasoning while being more memory-efficient.

AoT (Hong et al., 2024) proposes a multi-level reasoning framework that separates high-level planning from low-level execution in language model reasoning, leading to stronger generalization and performance than Chain-of-Thought.

RAFT++ (Xiong et al., 2025) is a lightweight policy-gradient method that improves training stability and KL efficiency by discarding completely wrong or completely correct samples.

AdaR (Lai et al., 2025) abstracts concrete queries into variable-based templates and translates reasoning logic into executable code, enabling models to reason over problem-solving logic rather than surface values while supporting execution-based verification.

GPG (Chu et al., 2026) is a minimalist reinforcement learning method based on policy gradients that directly optimizes the original RL objective without surrogate losses.

InTRO (Zhu et al., 2026) guides next-token generation with token-wise correction factors from the discrepancy between a policy model and its answer-conditioned counterpart, enabling token-level exploration and self-feedback in one forward pass.

IC-DAPO (Mei et al., 2026) leverages a model's in-context learning ability to estimate reasoning quality and reweights RLVR rewards accordingly, encouraging high-quality reasoning traces without requiring external evaluators or process-level supervision.

Expert 1 (algebraic expert)

SYSTEM:
You are a mathematics expert. Solve using algebraic manipulation.

USER:
Solve this problem: {question}

Expert 2 (strategic expert)

SYSTEM:
You are a strategic problem solver. Test edge cases and boundaries.

USER:
Solve this problem: {question}

Expert 3 (geometric expert)

SYSTEM:
You are a visual mathematician. Use geometric intuition or structure.

USER:
Solve this problem: {question}

Reasoning Synthesis

You are a mathematical reasoning expert.

Task: Integrate multiple candidate solutions into an optimal, step-by-step derivation process.

Guiding Principles:

1. Integration: Combine the best reasoning process from all candidate solutions. If one expert uses an algebraic method and another uses a geometric method, verify that their reasoning is consistent and confirm that they prefer the more rigorous algebraic method. However, you can also mention geometric intuition if it helps with understanding.

2. Deduplication: Remove redundant steps. If multiple experts performed the same calculation, write it only once.

3. Structure: Strictly organize the output results in the order of "Step 1", "Step 2", etc.

4. Consistency: Ensure logical flow. The final result must be consistent with the consensus reached by the experts.

Figure 2: Task prompts used in Multi-Expert Reasoning.

B TASK PROMPTS

Our framework consists of three modules: Multi-Expert Reasoning, Logic-Preserving Reasoning Abstraction, and Chain-level Validation and Reasoning. The prompts used for the tasks in these modules are illustrated in Figures 2, 3, and 4, respectively.

Query Mutator

You are a Math Problem Generator. Create a 'Mutated Variant' of the user's problem.

RULES:

1. Keep the logic/structure EXACTLY the same. DO NOT change the question type.

2. Change numbers to REASONABLE values.

- If original result was an integer, try to ensure new result is likely an integer (or simple fraction).

- Ensure geometry constraints hold (e.g., small side changes).

3. Output ONLY the new problem text.

Mutation Validator

SYSTEM:

You are a Strict Math Logic Auditor. Your job is to crash-test a math problem for contradictions.

Analyze the 'Mutated Problem' for the following FATAL FLAWS:

Domain Contradictions: (e.g., Probability > 1, Negative Length/Time/Count).

2. Geometry Violations: (e.g., Triangle inequality a+b<=c, Sum of angles != 180).

3. Algebra Impossibilities: (e.g., Sqrt of negative, Denominator is zero, Discriminant < 0 for real roots).

4. Integer Constraints: (e.g., Problem implies 'people' or 'cars' but equations yield 3.5).

5. Logical Consistency: Do the new numbers make the premises mutually exclusive?

Output Format:

ANALYSIS: [Brief thought process about constraints]

RESULT: [PASS or FAIL]

REASON: [If FAIL, explain the contradiction]

Figure 3: Task prompts used in Logic-Preserving Query Mutation.

Card Generator

You are a Mathematical Logic Analyzer.

Task: Break down the provided Reference Solution into clear, distinct logical steps.

RULES:

1. Quad Cards: For each step, define 4 distinct ways to describe the logic (Cards A/B/C/D).

2. Sequence: Ensure the steps flow logically from start to finish.

- "FORMAT:\n"
- "Step 1: || Card A: [Logic 1] Math: [Equation/Result] || Card B: [Logic 2] Math: [Equation/Result] || Card C: [Logic 3] Math: [Equation/Result] || Card D: [Logic 4] Math: [Equation/Result]\n"
- "Step 2: || ...\n"

Card Selector

You are a Logic Validator.

Task: Select the BEST logic card (A, B, C, or D) that correctly applies to the MUTATED problem.

Some cards might rely on old numbers (Invalid). Some are general (Valid).

Regenerate mechanism

You are a Robust Math Solver. A previous derivation was detected as INCORRECT.

Task: Regenerate the NEXT step, fixing the reported error.

Use the Quad Card format.

Logic Validator

You are a Logic Auditor. Check the solution trace for consistency.

Analyze specific errors:

1. Common Sense (e.g., negative length, people < 0)

2. Premise Conflict (e.g., Question says $x + y = 10$, derived $x + y = 6$)

3. Calculation Consistency (Does Step N follow Step N-1?)

Output Format:

STATUS: PASS or FAIL

REASON: [If FAIL, describe specific error to help correction]

Figure 4: Task prompts used in Step-wise Reasoning Abstraction.

C Effect of Expert Configuration

To investigate whether expert diversity is better achieved through heterogeneous models or role-specific prompting of a shared base model, we compare our default expert configuration with a multi-model variant. Specifically, for the 4B settings, the multi-model variant uses Qwen3-4B, Qwen3.5-4B, and Llama3.2-3B as the three experts, while for the larger settings, it uses Qwen3-8B, Qwen3.5-9B, and Llama3.1-8B. In contrast, our framework instantiates all three experts using the same base model with distinct reasoning prompts, while keeping all other components unchanged. As shown in Table 7, our default configuration consistently achieves higher average performance than the multi-model variant across all four settings, with gains of 2.5, 2.4, 1.6, and 2.4 percentage points, respectively. These results suggest that model heterogeneity does not necessarily improve expert collaboration, while role-specific prompting of a shared model provides sufficient reasoning diversity with more consistent overall performance.

Table 7: Performance comparison of different expert configurations.

Methods	MATH500	Min.Math	Olympiad	Col.Math	AMC23	AIME25	AVG
			Qwen3-4B				
Multi-model Experts	73.6	22.6	41.6	40.6	58.1	16.7	42.2
Ours	79.3	23.9	44.5	41.2	62.5	16.7	44.7
			Qwen3-8B				
Multi-model Experts	66.7	24.9	47.1	42.1	54.3	20.1	42.5
Ours	81.4	24.5	44.2	40.4	58.8	19.8	44.9
			Qwen3.5-4B				
Multi-model Experts	81.1	30.1	45.6	49.7	68.6	19.7	49.1
Ours	83.2	31.4	47.4	51.0	70.5	20.6	50.7
			Qwen3.5-9B				
Multi-model Experts	83.7	33.3	48.3	53.1	73.1	22.4	52.3
Ours	86.3	35.6	50.6	55.4	76.0	24.4	54.7

D Inference Cost Analysis

To evaluate the computational overhead of our framework, we compare its per-problem inference cost with InTRO and AoT in terms of generated tokens, LLM calls, and inference latency. We select InTRO and AoT as representative baselines closely related to reasoning abstraction and refinement. Specifically, we report the average cost over six mathematical benchmarks using Qwen3-4B and Qwen3-8B, together with the corresponding reasoning accuracy and cost-to-accuracy ratio (RAT). As shown in Table 8, our framework incurs higher inference costs than the baselines due to the additional abstraction, routing, and validation procedures, but achieves substantially better reasoning performance. For example, with Qwen3-8B, our method reaches an average accuracy of 44.85%, compared with 39.50% for InTRO and 37.98% for AoT, while requiring 23.83K tokens and 13.17 LLM calls per problem on average. We also observe that Qwen3-8B requires fewer tokens and LLM calls than Qwen3-4B under our framework (23.83K vs. 26.83K tokens and 13.17 vs. 14.50 calls), suggesting that stronger base models can reduce the frequency of intermediate retries.

E Analysis of Step-wise Execution Overhead

To evaluate the computational overhead introduced by the retry mechanism, we measure the average number of executions required for each reasoning step. Specifically, the metric is computed as the total number of step executions, including retries, divided by the total number of reasoning steps, where a value of 1 indicates that each step is executed only once. As shown in Table 9, the average numbers of executions per step are 1.25, 1.30, 1.23, and 1.17 for Qwen3-4B, Qwen3-8B, Qwen3.5-4B, and Qwen3.5-9B, respectively, with all values ranging from 1.1 to 1.4 across benchmarks. These results show that most reasoning steps require only a small number of additional executions, indicating that the retry mechanism introduces limited computational overhead.

Table 8: Inference cost and efficiency comparison across different methods. AVG denotes the average inference cost across benchmarks, ACC denotes the average accuracy, and RAT is computed as AVG/ACC.

Metrics		MATH500	Min. Math	Olympiad	Col. Math	AMC23	AIME25	AVG	ACC	RAT
Qwen3-4B										
Tokens per Problem (k)	InTRO	13	12	26	17	18	32	19.67	39.63	0.50
	AoT	11	10	24	15	16	30	17.67	37.15	0.48
	Ours	18	19	35	24	25	40	26.83	44.68	0.60
LLM Calls per Problem	InTRO	9	8	9	9	8	12	9.17	39.63	0.23
	AoT	8	7	8	7	7	11	8.00	37.15	0.22
	Ours	13	14	16	14	12	18	14.50	44.68	0.32
Inference Latency (s)	InTRO	5.1	7.7	8.9	8.3	7.5	9.6	7.85	39.63	0.20
	AoT	4.7	7.4	8.4	8.0	7.1	9.1	7.45	37.15	0.20
	Ours	11.9	16.3	15.7	15.2	13.8	19.0	15.32	44.68	0.34
Qwen3-8B										
Tokens per Problem (k)	InTRO	11	10	21	14	16	35	17.83	39.50	0.45
	AoT	9	8	19	12	14	33	15.83	37.98	0.42
	Ours	15	16	28	20	22	42	23.83	44.85	0.53
LLM Calls per Problem	InTRO	8	7	8	8	7	11	8.17	39.50	0.21
	AoT	7	7	8	6	7	10	7.50	37.98	0.20
	Ours	12	13	14	13	10	17	13.17	44.85	0.29
Inference Latency (s)	InTRO	6.2	9.2	10.2	9.6	9.2	11.7	9.35	39.50	0.24
	AoT	5.9	8.8	9.7	9.1	8.7	11.2	8.90	37.98	0.23
	Ours	13.6	17.9	17.2	16.9	15.6	20.7	16.98	44.85	0.38

Table 9: Average number of executions per reasoning step.

Methods	MATH500	Min.Math	Olympiad	Col.Math	AMC23	AIME25	AVG
Qwen3-4B	1.1	1.2	1.3	1.3	1.2	1.4	1.25
Qwen3-8B	1.3	1.3	1.3	1.3	1.2	1.4	1.30
Qwen3.5-4B	1.1	1.2	1.3	1.2	1.2	1.4	1.23
Qwen3.5-9B	1.1	1.2	1.2	1.1	1.1	1.3	1.17