Chain-of-Reasoning: Towards Unified Mathematical Reasoning in Large Language Models via a Multi-Paradigm Perspective

Yiyao Yu[1*] **Yuxiang Zhang**[1*] **Dongdong Zhang**[2†] **Xiao Liang**[1]
Hengyuan Zhang[1] **Xingxing Zhang**[2] **Mahmoud Khademi**[2] **Hany Awadalla**[2]
Junjie Wang[1†] **Yujiu Yang**[1†] **Furu Wei**[2]

[1]Tsinghua University [2]Microsoft
yuyy23@mails.tsinghua.edu.cn, joell070408@gmail.com, wangjunjie@sz.tsinghua.edu.cn
{dozhang,xizhang,mkhademi,hanyh,fuwei}@microsoft.com

Abstract

Large Language Models (LLMs) have made notable progress in mathematical reasoning, yet they often rely on single-paradigm reasoning that limits their effectiveness across diverse tasks. In this paper, we introduce Chain-of-Reasoning (CoR), a novel unified framework that integrates multiple reasoning paradigms — Natural Language Reasoning (NLR), Algorithmic Reasoning (AR), and Symbolic Reasoning (SR) — to enable synergistic collaboration. CoR generates multiple potential answers using different reasoning paradigms and synthesizes them into a coherent final solution. We propose a Progressive Paradigm Training (PPT) strategy that allows models to progressively master these paradigms, culminating in the development of CoR-Math-7B. Experimental results demonstrate that CoR-Math-7B significantly outperforms current SOTA models, achieving up to a 41.0% absolute improvement over GPT-4o in theorem proving tasks and a 15% improvement over RL-based methods on the MATH benchmark in arithmetic tasks. These results show the enhanced mathematical comprehensive ability of our model, enabling zero-shot generalization across tasks. The code is available at https://github.com/microsoft/CoR.

1 Introduction

While LLMs have shown strong performance in solving mathematical tasks (Feigenbaum et al., 1963; Hosseini et al., 2014), advanced open-source reasoners still struggle with solving comprehensive mathematical problems, including both arithmetic computation and theorem proving.

Existing works (Xin et al., 2024; Yang et al., 2024; Wu et al., 2024; Zhang et al., 2024) are often trained on specific tasks, aiming to enhance their ability to independently derive answers based on specific structured knowledge representation. This representation is known as the reasoning paradigm, involving Natural Language Reasoning (NLR), Algorithmic Reasoning (AR), and Symbolic Reasoning (SR), as depicted in Fig. 2 (a). Specifically, NLR uses natural language for reasoning via common sense and semantic context, providing explicit step-by-step explanations (Wei et al., 2022). AR leverages code

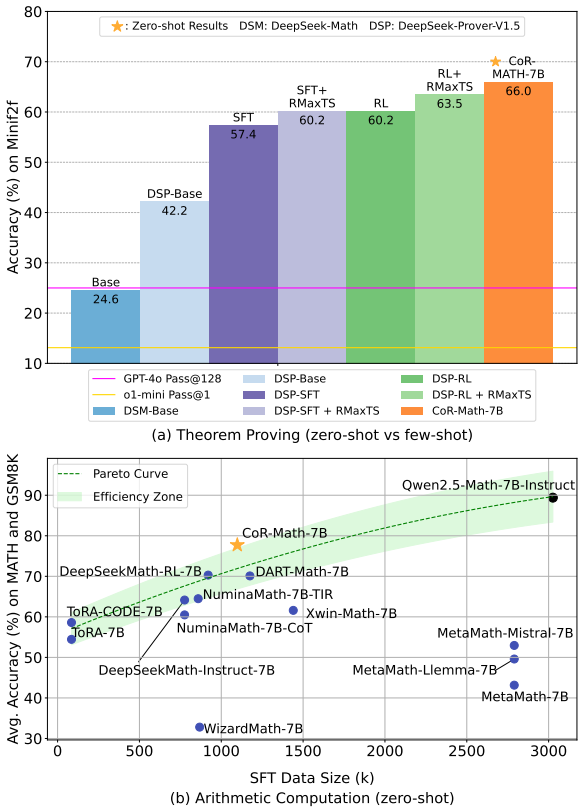

Figure 1: A comprehensive comparative analysis of CoR-Math-7B and baseline models across mathematical tasks. (a) shows the effectiveness of CoR-Math-7B (zero-shot) in theorem proving tasks. (b) shows a resource-efficiency analysis for arithmetic computation tasks, where CoR-Math-7B achieves optimal resource efficiency and near-optimal zero-shot performance.

*Equal contribution. Yiyao Yu did this work during the internship at Microsoft Research Asia. Yuxiang Zhang did this work as a research assistant at Tsinghua University.
†Corresponding Author.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 24914–24937
July 27 - August 1, 2025

to emphasize computational operations and execution processes, such as generating Python code for execution (Chen et al., 2023; Gao et al., 2023). SR uses logical symbols and axiomatic systems for formalized reasoning, with recent methods (Xin et al., 2024; Huang et al., 2024; Wu et al., 2024) considering numerous symbolic trajectories via tree-based search for theorem proving. However, these methods mainly focus on optimizing single-paradigm reasoning, creating models that demonstrate asymmetrical performance across different mathematical tasks. For instance, models specialized in NLR may exhibit deficiencies in theorem proving and vice versa. This fragmented paradigm constrains the upper-bound performance on individual tasks and undermines the model's capacity for cross-paradigm generalization.

Researchers have explored various strategies to tackle these challenges. To improve the single-task performance, some works integrate tools to overcome the limitations of single-paradigm reasoning (Gou et al., 2024; LI et al., 2024), as shown in Fig. 2 (b). While recognizing the benefits of combining reasoning paradigms, they still rely on a single paradigm, neglecting the possibility that the second paradigm could independently complete the reasoning, thereby constraining overall potential. Besides, to improve cross-task generalization, several studies (Shao et al., 2024; Huang et al., 2024) incorporate diverse task samples into large-scale datasets, such as those drawn from theorem proving tasks that focus exclusively on SR solutions, or from arithmetic problems that emphasize NLR solutions. Although models trained on such data are capable of cross-task reasoning, they still rely on demonstrations for effective transfer.

To address these limitations, we propose Chain-of-Reasoning (CoR), a unified framework integrating NLR, AR, and SR to produce synergistic benefits. As shown in Fig. 2 (c), CoR enables multi-paradigm reasoning by applying different reasoning paradigms to derive multiple potential answers, which are then summarized into a final solution. This framework allows iterative reasoning across paradigms, leveraging prior results to enhance single-task performance. Moreover, CoR unifies multi-paradigm reasoning across tasks, enabling zero-shot generalization. Adjusting prompts to control reasoning depth improves adaptability to diverse tasks. As a result, we introduce Multi-Paradigm Mathematical (MPM), a dataset comprising 167k reasoning paths, and propose Progressive Paradigm Training (PPT), a method enabling models to progressively master multiple reasoning paradigms, leading to CoR-Math-7B.

We evaluate CoR-Math-7B on five challenging mathematical reasoning benchmarks, covering both arithmetic computation and theorem proving. Our results show that LLMs equipped with CoR significantly surpass current SOTA baselines. In theorem proving (Fig. 1 (a)), CoR-Math-7B improves zero-shot performance over GPT-4o (Hurst et al., 2024) by 41.0%, outperforming all few-shot reasoners. For arithmetic tasks, CoR-Math-7B achieves a 24.2% absolute improvement over GPT-4 on MATH. Furthermore, as shown in Fig. 1 (b), CoR-Math-7B efficiently utilizes resources to surpass the optimal performance curve of single-paradigm approaches. Unlike mainstream methods that conduct extensive searches within a single paradigm, CoR enhances test-time inference in multiple paradigms. These results show that CoR can solve comprehensive mathematical problems through multi-paradigm reasoning, requiring less training data and lowering reasoning costs.

2 Related Work

Reasoning Paradigms in LLMs. Recent advancements in LLMs focus on single-paradigm reasoning, with each paradigm representing a distinct method for knowledge representation and logical inference. NLR uses human-readable language for commonsense reasoning and step-by-step deductions (Wei et al., 2022; Yao et al., 2023; Zhou et al., 2023; Besta et al., 2024; Sel et al., 2024), AR generates executable code for precise calculations (Chen et al., 2023; Rozière et al., 2023), and SR utilizes logical symbols for theorem proving (Xin et al., 2024). Some methods enhance reasoning by integrating external tools like calculators or code interpreters, yet they remain within a single paradigm (Gou et al., 2024). While effective in specific tasks, they struggle with cross-domain generalization and dynamic environments. To address these limitations, the CoR framework enables collaboration across reasoning paradigms, enabling zero-shot multitask generalization.

Mathematical Problem Solving with LLMs. Advanced research highlights the potential of LLMs in solving mathematical problems (Zhu et al., 2023; Lin et al., 2024b; Luo et al., 2025). Several studies have developed unified solvers by synthesizing mathematical data to address chal-

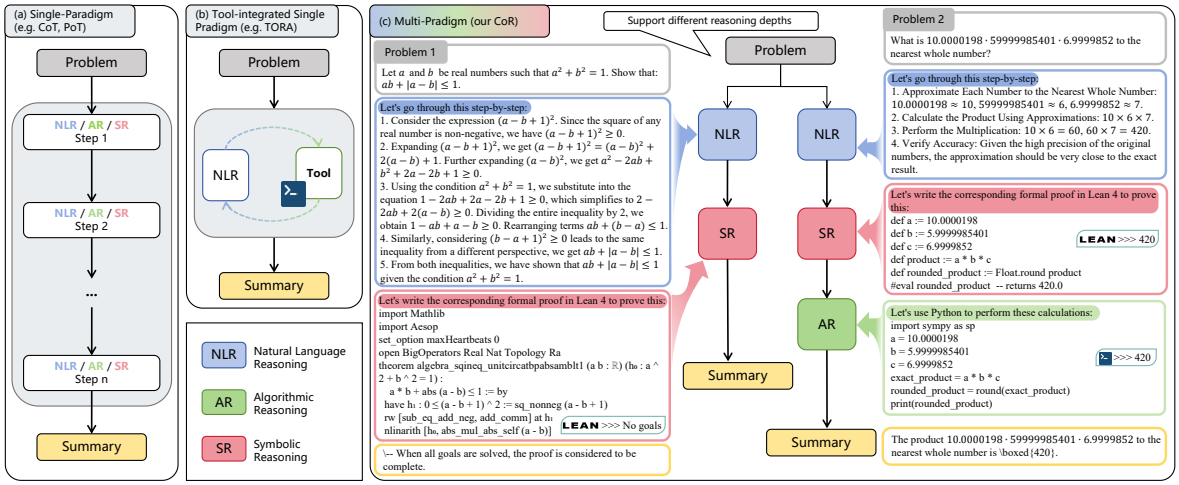

Figure 2: The reasoning process under different paradigms: (a) In single-paradigm reasoning, each reasoning step relies on the same knowledge medium, such as Natural Language (NL), algorithms, or symbols. (b) In tool-integrated single-paradigm, NL is used for reasoning, while code assists in solving specific sub-problems. After obtaining the execution results, the reasoning continues using NL. (c) The proposed CoR reasoning framework, along with several examples, shows that multi-paradigm reasoning allows for varying reasoning depths to address different types of problems.

lenges like arithmetic computation and theorem proving (Huang et al., 2024; Shao et al., 2024). However, most approaches focus on optimizing a single paradigm. For instance, in arithmetic computation, tools are integrated to assist natural language reasoning. (Gou et al., 2024). In theorem proving, models rely on specialized data like pre-training data and Reinforcement Learning (RL) reward data, and tree-based search methods to generate numerous possible solutions (Ying et al., 2024b; Xin et al., 2024). These approaches either neglect complete reasoning or depend on large-scale search within the solution space, limiting performance gains within a single paradigm. To address these, we introduce the CoR-Math-7B model with complete reasoning processes to explore an expanded multi-paradigm solution space.

3 Chain-of-Reasoning Framework

3.1 Overview

CoR aims to enable LLMs to perform a series of multi-paradigm reasoning on any type of mathematical problem, ultimately arriving at a solution. Specifically, given a mathematical problem x, LLMs ($\mathbb{P}$) can infer the result y by following multiple reasoning paradigms, where each reasoning paradigm τ includes n reasoning paths $\{rp_1, ..., rp_n\}$. We represent this single-paradigm scenario as $y \sim \mathbb{P}(y|x, \tau)$. To simplify the process for each reasoning paradigm, we set $n = 1$ as default (Details in Appendix A).

Inspired by recent works (Wei et al., 2022) in encouraging step-by-step reasoning, we introduce CoR, which extends from a single paradigm to three paradigms $\Gamma = (\tau_1, \tau_2, \tau_3)$. For generating multiple chained reasoning paradigms for a given problem, CoR follows the steps outlined below. The reasoning process begins with the problem x and the first reasoning paradigm τ_1. Subsequently, each paradigm τ_i in the sequence Γ is generated based on the problem x and the previously established paradigms $\tau_1, ..., \tau_{i-1}$, as represented by $\tau_i \sim \mathbb{P}(\tau_i|x, \tau_1, ..., \tau_{i-1})$. Finally, the outcomes of all the reasoning paradigms are aggregated to derive the final result y, expressed as $y \sim \mathbb{P}(y|x, \tau_{\text{NLR}}, \tau_{\text{SR}}, \tau_{\text{AR}})$, considering three paradigms: NLR, SR, and AR. In detail, τ_{SR} and τ_{AR} include both the complete reasoning paths and the interaction processes with tool outputs. In this study, τ_{SR} uses the Lean prover, and τ_{AR} applies a Python compiler to obtain reasoning results.

In our workflow (Fig. 3), the training pipeline includes: (a) collecting the MPM dataset with deep reasoning paths; (b) using progressive paradigm training to enhance reasoning across paradigms; and (c) applying the trained LLMs for zero-shot inference with sequential multi-paradigm sampling to explore diverse solutions.

3.2 Collecting Dataset

To train CoR model, we extend the single-paradigm datasets to incorporate multiple reason-

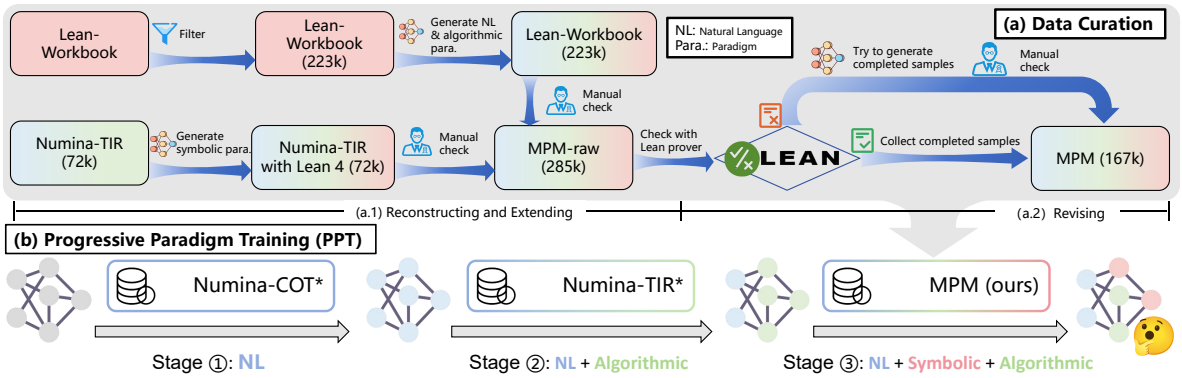

Figure 3: An overview of (a) the Multi-Paradigm Math (MPM) dataset construction process, involving reconstruction, extension, and theorem prover verification, and (b) the Progressive Paradigm Training (PPT) method, where the model is trained with increasing reasoning paradigms in stages.

ing paradigms, denoted as $< \mathsf{x}, \mathsf{NLR}, \mathsf{SR}, \mathsf{AR}, \mathsf{y} >$. As presented in Fig. 3 (a), the training data collection process involves two stages: (a.1) Reconstructing and Extending, and (a.2) Revising. In the first stage, we reconstruct high-quality open-source mathematical data as seed samples and synthesize additional reasoning paradigms to form the MPM-raw dataset. In the second stage, a theorem prover examines MPM-raw samples, and a mathematical reasoner revises those that fail, compiling all completed data into the final MPM dataset.

Stage 1: Reconstructing and Extending. We introduce a universal text template for multi-paradigm reasoning (details in Appendix B.1), which standardizes the placement and relationships of reasoning paradigms. It supports various reasoning depths and flexible combinations of different reasoning paradigms. As shown in Fig. 2 (c), Problem 1 shows an instance incorporating both NLR and SR paradigms, and Problem 2 integrates three reasoning paradigms. To ensure data integrity and prevent potential biases, we preprocess the Numina-TIR (LI et al., 2024) and Lean-Workbook (Ying et al., 2024a) datasets in two steps. First, samples without corresponding solutions are removed. Second, we further reconstruct and extend these datasets by leveraging powerful LLMs $\mathcal{G}$, such as GPT-4o (Hurst et al., 2024). These models generate missing reasoning paradigms τ_g and refine existing ones τ', ensuring comprehensive coverage and logical consistency. To effectively guide the processes of augmentation and refinement, we develop tailored prompts p_s for each seed dataset (examples in Appendix B.2). The procedure can be described in:

$$\tau_g \sim \mathbb{P}_{\mathcal{G}}(\tau_g \mid p_s \oplus x \oplus y \oplus \tau'), \tau' \in \{\tau_{\text{NLR}}, \tau_{\text{SR}}, \tau_{\text{AR}}\}, \quad (1)$$

where $\oplus$ means concatenation. After that, we conduct a manual review of all samples. Given that alternative approaches are readily verified through external tools, we focus on the accuracy of the NLR and AR. This approach considerably lowers the requisite skill level of annotators. To prevent data leakage, we compute the Levenshtein distance (Miller et al., 2009) between training and test problems, as detailed in Appendix C.1. As a result, this phase yields the MPM-raw dataset, comprising approximately $285,000$ synthetic samples.

Stage 2: Revising. The MPM-raw dataset interacts with the Lean prover to verify proof steps, guiding the filtering and modification of reasoning paths. The proof τ_{SR} is submitted to the prover, and if verification is successful without errors, the entire multi-paradigm reasoning path is collected to the MPM dataset. Otherwise, the error information ε returned by the prover is fed into a revising model $\mathbb{P}_R$. Furthermore, this model generates a revised proof $\tilde{\tau}_{\text{SR}}$ based on a prompt p_ε. The relationship can be expressed as:

$$\tilde{\tau}_{\text{SR}} \sim \mathbb{P}_R(\tilde{\tau}_{\text{SR}} \mid p_\varepsilon \oplus x \oplus y \oplus \tau_{\text{SR}}), \quad (2)$$

where the revised proof $\tilde{\tau}_{\text{SR}}$ is then resubmitted to the Lean prover for verification. This iterative process continues for up to 64 iterations or until $\tilde{\tau}_{\text{SR}}$ is verified as correct. In detail, we employ DeepSeek-Prover-V1.5 (Xin et al., 2024) as the $\mathbb{P}_R$.

Consequently, the MPM dataset comprises $82,770$ problems and $167,412$ multi-paradigm

reasoning solutions.

3.3 Training

Inspired by recent advances (LI et al., 2024), we introduce the Progressive Paradigm Training (PPT) strategy, enabling LLMs to gradually master diverse reasoning paradigms. As shown in Fig. 3 (b), this framework expands the model's reasoning abilities by sequentially introducing different paradigms at each stage. Each training stage uses a different combination of reasoning paradigms. In stage ①, given the dominance of NL in the pre-training data of language models (Dubey et al., 2024), we create Numina-CoT* as an initialized teaching stage. This is done by modifying the original Numina-CoT dataset (LI et al., 2024) according to our universal text template, enabling the model to learn to use NL to solve complex mathematical problems. Based on the question x, the model performs reasoning τ_{NLR} and generates the answer y. The generated sequence is $z = [x]\tau_{\text{NLR}}y$, where $[x]$ represents the inputs after tokenization. For simplicity, we first consider the loss function for a single sample:

$$\mathcal{L}_{\text{sample}} = -\sum_{t=1}^{|z|} \log \mathbb{P}_\theta(z_t \mid z_{<t}), \qquad (3)$$

where θ represents the model parameters, z_t is the t-th token in the generated sequence, and $z_{<t}$ indicates all tokens before the t-th token in the generated sequence. In stage ②, considering a certain proportion of code corpora in the pre-training data, we expand the training dataset to include two paradigms: NLR and AR. Similar to Numina-CoT*, we modify the original Numina-TIR to create Numina-TIR*. With this modified dataset, the generated sequence is $z = [x]\tau_{\text{NLR}}\tau_{\text{AR}}y$. After this stage, the model can handle problems that require precise answers. In stage ③, we further expand the training data to three reasoning paradigms by utilizing the MPM dataset, where z is $[x]\tau_{\text{NLR}}\tau_{\text{AR}}\tau_{\text{SR}}y$. After full stages, the trained CoR-Math-7B model not only masters NLR and AR but also can perform rigorous logical SR.

Unlike traditional curriculum and incremental learning, which sequencing tasks by increasing difficulty within a single paradigm or accumulating knowledge, PPT progressively integrates fundamentally distinct reasoning paradigms, transitioning from familiar to unfamiliar, while fostering synergy across paradigms.

3.4 Inference

We present an inference method combining variable reasoning depth and multi-paradigm sampling to solve comprehensive mathematical tasks.

Prompts with variable reasoning depth. In the zero-shot inference phase, CoR-Math-7B exhibits proficiency in multi-paradigm reasoning, enabling adjustable reasoning depths by modifying prompts based on task requirements. Initially, the model is prompted to conduct NLR, thereby activating a wide range of knowledge patterns associated with natural language from the pre-training corpus. Subsequently, the inference method is tailored to the specific problem type. For example, in theorem proving, the model switches to SR, which is more suitable for formal deduction. Since the SR output is structured in Lean 4, the relevant proof segment can be extracted as the final solution. In arithmetic computations, the model first employs NLR, followed by SR for logical coherence, and concludes with AR for precise calculations. A summary box is utilized to present the final results. The paradigm-switching behavior depends on the presence of prompts and can be categorized into instruction-followed reasoning and instruction-free reasoning. This section primarily presents results from the former. Instruction-free reasoning, which explores how the model autonomously switches paradigms without prompt guidance, will be demonstrated in Appendix D.1. This adaptable paradigm effectively captures the distinct reasoning patterns inherent in different paradigms and demonstrates flexibility in accommodating a variety of scenarios.

Sequential Multi-Paradigm Sampling (SMPS). Instead of token-level sampling based on tree structures (Qiu et al., 2024; Xiong et al., 2024) within single-paradigm reasoning paths, we purpose sequential paradigm-level sampling method, named SMPS. This allows the model to generate outputs by sampling across different paradigms. For instance, in a two-paradigm reasoning scenario, the model first instantiates J distinct paths for the initial reasoning paradigm τ_1.

$$\tau_{1j} \sim \mathbb{P}(\tau_{1j} \mid x), \quad \forall j \in 1, \ldots, J \qquad (4)$$

Subsequently, for each of these τ_1 paths, the model instantiates K possible paths for the secondary reasoning paradigm τ_2.

$$\tau_{2k} \sim \mathbb{P}(\tau_{2k} \mid x, \tau_{1j}), \quad \forall k \in 1, \ldots, K, \forall j \qquad (5)$$

This hierarchical sampling process yields a total of $J \times K$ potential responses, denoted as y.

$$y_{jk} \sim \mathbb{P}(y_{jk} \mid x, \tau_{1j}, \tau_{2k}), \quad \forall j, k \tag{6}$$

Overall, the SMPS method utilizes a combinatorial expansion of reasoning paths to explore a diverse paradigm-based solution space, enhancing the robustness and depth of the reasoning results.

4 Experimental Settings

4.1 Evaluation Setup

Datasets. We conduct extensive experiments to evaluate the model's mathematical reasoning abilities, including arithmetic computation and theorem proving. The arithmetic computation ability is evaluated on the GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), AMC2023 (AI-MO, 2024b) and AIME2024 (AI-MO, 2024a) datasets. Theorem proving ability is evaluated on the miniF2F test set (Zheng et al., 2022), which features mathematical problems of Olympiad difficulty. Further details are in Appendix B.4.

Metrics. Accuracy is the primary evaluation metric. For arithmetic tasks, we adopt widely-used CoT settings (Wei et al., 2022), rounding answers to the nearest integer. The SymPy library[1] is used for parsing and evaluation. To handle numerical representation variations, the model explicitly states the final answer (LI et al., 2024). For theorem proving, we follow the recent advances (Xin et al., 2024) adapting the miniF2F benchmark from Lean 3 to Lean 4. The pass@N metric evaluates proof correctness within N attempts. Our CoR-Math-7B uses SMPS with NLR and SR, where $N = N_{\text{NLR}} \times N_{\text{SR}}$. Model settings are detailed in Appendix B.5.

4.2 Implementation Details

We fine-tuned widely-used DeepSeekMath-Base 7B (Shao et al., 2024) and Llama-3.1 8B (Dubey et al., 2024) models, employing our PPT method on MPM dataset. Unless otherwise specified, CoR-Math-7B model is based on DeepSeekMath-Base 7B. The details are available in Appendix B.3.

4.3 Baselines

We examine three categories of baseline models, with results reported using CoT prompting.

[1]https://www.sympy.org/

General-purpose mathematical models. To evaluate CoR's generalization, we includes Mustard (Huang et al., 2024), DeepSeekMath (Shao et al., 2024), InternLM-Math (Ying et al., 2024b), Llama-3.1 (Dubey et al., 2024), Mistral (Jiang et al., 2023), and Llemma (Azerbayev et al., 2024).

Task-specific mathematical models. We consider several expert models on mathematical optimization for specific tasks, such as large-scale mathematical data and inference search improvements. The arithmetic experts encompass Qwen2.5-Math (Yang et al., 2024), WizardMath (Luo et al., 2023), MetaMath (Yu et al., 2024), DART-Math (Tong et al., 2024), InternLM-Math (Ying et al., 2024b), , DeepSeekMath-Instruct / RL (Shao et al., 2024), Xwin-Math (Li et al., 2024), ToRA (Gou et al., 2024), and NuminaMath (LI et al., 2024). The theorem proving experts include LLM-Step (Welleck and Saha, 2023), GPT-f (Polu and Sutskever, 2020), Lean-STaR (Lin et al., 2024a), Hypertree Proof Search (Lample et al., 2022), DeepSeek-Prover (Xin et al., 2024), and InternLM2.5-StepProver (Wu et al., 2024).

Foundation and proprietary models. We present open-source foundation models Llama-3.1 (Dubey et al., 2024), Mistral (Jiang et al., 2023), and Mixtral (Jiang et al., 2024), along with proprietary models o1-mini (OpenAI, 2024), GPT-4 (OpenAI, 2023), and GPT-4o (Hurst et al., 2024).

5 Main Results

5.1 Comparisons with General-purpose Mathematical Models

To evaluate the comprehensive mathematical reasoning abilities of CoR-Math-7B, we compare it with widely-used models and SOTA mathematical models, which are based on single reasoning paradigms. As shown in Table 1, CoR-Math-7B outperforms others across five challenging benchmarks in a zero-shot setting, highlighting its strong cross-task versatility. The main findings are: (1) arithmetic computation: CoR-Math-7B achieves the best results, outperforming InterLM2-Math-Plus-7B by an absolute 13.7% on MATH and 2.9% on GSM8K. Compared to Llama-3.1-8B-Instruct, it achieves an increase in correct answers on AMC2023 (34 vs 16) and AIME2024 (12 vs 5). (2) theorem proving: CoR-Math-7B sets a new zero-shot benchmark on miniF2F, even surpassing GPT-4o model by a 41.0% absolute increase in

Model	Model Types	Arithmetic Computation				Theorem Proving	
		MATH Pass@1	GSM8k Pass@1	AMC2023 Maj@64	AIME2024 Maj@64	miniF2F-test Sample Budget (N)	miniF2F-test Pass@N
o1-mini	Proprietary	$90.0^{\emptyset}$	$94.8^{\emptyset}$	$38/40^{\emptyset}$	$17/30^{\emptyset}$	1	13.2
GPT-4	Proprietary	$42.5^{\emptyset}$	$87.1^{\emptyset}$	$25/40^{\emptyset}$	$6/30^{\emptyset}$	128	24.6
GPT-4o	Proprietary	$76.6^{\emptyset}$	$90.5^{\emptyset}$	$24/40^{\emptyset}$	$3/30^{\emptyset}$	128	25.0
Llama-3.1-8B	Foundation	$4.2^{\emptyset}$	$6.2^{\emptyset}$	$1/40^{\emptyset}$	$0/30^{\emptyset}$	128	25.8
Llama-3.1-8B-Instruct	Foundation	$47.2^{\emptyset}$	$76.6^{\emptyset}$	$16/40^{\emptyset}$	$5/30^{\emptyset}$	128	23.4
Mistral-7B	Foundation	14.3	40.3	$5/40^{\emptyset}$	$0/30^{\emptyset}$	$1\times32\times100$	22.1
Mixtral-8x7B	Foundation	28.4	74.4	$8/40^{\emptyset}$	$0/30^{\emptyset}$	$1\times32\times100$	23.4
MUSTARD	GMM	$13.8^{\emptyset}$	$27.9^{\emptyset}$	-	-	1	7.8
Llemma-7B	GMM	18.6	41.0	$2/40^{\emptyset}$	$0/30^{\emptyset}$	$1\times32\times100$	26.2
DeepSeekMath-7B-Base	GMM	$11.8^{\emptyset}$	$22.2^{\emptyset}$	$3/40^{\emptyset}$	$0/30^{\emptyset}$	$1\times32\times100$	28.3
InternLM2-Math-7B-Base	GMM	21.5	49.2	$6/40^{\emptyset}$	$0/30^{\emptyset}$	$1\times32\times100$	30.3
InternLM2-Math-Plus-7B	GMM	$53.0^{\emptyset}$	$85.8^{\emptyset}$	$15/40^{\emptyset}$	$1/30^{\emptyset}$	$1\times32\times100$	43.3
CoR-Math-7B	GMM	$\mathbf{66.7}^{\emptyset}$	$\mathbf{88.7}^{\emptyset}$	$\mathbf{34/40}^{\emptyset}$	$\mathbf{12/30}^{\emptyset}$	128×1 32×100 128×128	$52.9^{\emptyset}$ $59.4^{\emptyset}$ $\mathbf{66.0}^{\emptyset}$

Table 1: A overall comparison of CoR-Math-7B with three types of general mathematical reasoners (Proprietary, Foundational, and General-purpose Mathematical Models (GMM)) on three mathematical benchmarks. Results are shown for zero-shot (denoted by $\emptyset$) or few-shot settings by default. For the miniF2F benchmark, we report the best results from the relevant literature with a specified sample budget. **Bolded** scores are the best performance among all models except for the proprietary one.

Model	Sample Budget N	miniF2F
LLMStep	$1\times32\times100$	27.9
GPT-f	$64\times8\times512$	36.6
Hypertree Proof Search	64×5000	41.0
Lean-STaR	$64\times1\times50$	46.3
DeepSeek-Prover-V1.5-Base	6400	42.2
DeepSeek-Prover-V1.5-SFT + RMaxTS	4×6400	56.3
DeepSeek-Prover-V1.5-SFT + RMaxTS	32×6400	60.2
DeepSeek-Prover-V1.5-RL + RMaxTS	4×6400	59.6
DeepSeek-Prover-V1.5-RL + RMaxTS	32×6400	63.5
InternLM2.5-StepProver	$64\times32\times100$	54.5
InternLM2.5-StepProver-BF	$1\times32\times600$	47.3
InternLM2.5-StepProver-BF	$256\times32\times600$	59.4
InternLM2.5-StepProver-BF+CG	$2\times32\times600$	50.7
InternLM2.5-StepProver-BF+CG	$256\times32\times600$	65.9
CoR-Math-7B	128×128	$\mathbf{66.0}^{\emptyset}$

Table 2: The zero-shot ($\emptyset$) performance of CoR-Math-7B and the few-shot performance of theorem proving optimized models on the miniF2F benchmark. The highest scores are in **bold**.

few-shot setting. (3) zero-shot vs few-shot: CoR-Math-7B's zero-shot performance exceeds all few-shot results from other models, with an absolute 37.7% improvement over DeepSeekMath-7B-Base on miniF2F. These findings indicate the generalization capability and comprehensive mathematical reasoning ability of the CoR framework. This suggests NLR descriptions and SR verification rehearse precise mathematical reasoning for AR. Additionally, we provide qualitative analysis of error cases in Appendix D.2. In summary, CoR effectively handles diverse reasoning challenges, with synergies across paradigms boosting its overall performance.

5.2 Comparisons with Theorem Proving Experts

Table 2 shows CoR-Math-7B's impressive zero-shot performance on miniF2F, achieving 66.0% accuracy without demonstrations, while competitors require few-shot examples. Its zero-shot performance is noteworthy when considering computational efficiency. For instance, under similar computational constraints, CoR-Math-7B surpasses DeepSeek-Prover-V1.5-RL + RMaxTS by 6.4% absolute points (66.0% vs 59.6% with 128×128 vs 4×6400) and InternLM2.5-StepProver-BF+CG by 15.3% absolute points (66.0% vs 50.7% with 128×128 vs $2 \times 32 \times 600$). Even with larger sample sizes, CoR-Math-7B remains superior, surpassing few-shot InternLM2.5-StepProver-BF+CG by 0.1%, despite considering 300 times more solutions. These results highlight the efficiency of multi-paradigm reasoning and CoR's ability to explore paradigm-based solution spaces.

5.3 Comparisons with Arithmetic Experts

Since several mathematical reasoners are primarily trained on arithmetic computation rather than formal theorem proving, and often struggle with the latter, we classify them as experts in arithmetic computation to enable a fair comparison. Table 3 highlights CoR-Math-7B's strong performance in arithmetic computation, surpassing expert models and achieving a 15% improvement over RL-based methods, with DeepSeekMath-RL-7B as the rep-

Model	MATH	GSM8k
	ZS Pass@1	ZS Pass@1
WizardMath	10.7	54.9
MetaMath-7B	19.8	66.5
MetaMath-Llemma-7B	30.0	69.2
MetaMath-Mistral-7B	28.2	77.7
ToRA	40.1	68.8
ToRA-CODE	44.6	72.6
NuminaMath-7B-CoT	54.4†	66.6†
Xwin-Math-7B	40.6	82.6
DeepSeekMath-Instruct-7B	46.8	73.6
NuminaMath-7B-TIR	55.3†	73.6†
DeepSeekMath-RL-7B	51.7	88.2
DART-Math-7B	53.6	86.6
Qwen2.5-Math-7B-Instruct	**83.6**	**95.2**
CoR-Math-7B	66.7	88.7

Table 3: Zero-shot (ZS) performance on the MATH and GSM8K benchmarks for the arithmetic task. † denotes our reported results with the open-sourced model weights. The best results are in **bold**, and the second-best results are underlined.

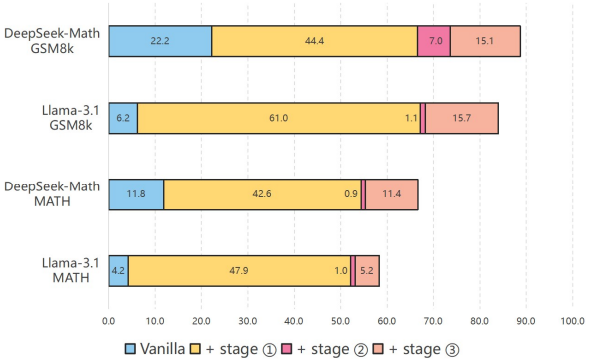

Figure 4: An evaluation of the effectiveness of the PPT strategy. We present the zero-shot Pass@1 results on the MATH and GSM8k benchmarks across three cumulative stages of the PPT strategy. The results highlight the PPT strategy's cumulative effectiveness, showing increased performance with each progressive stage.

Benchmark	NLR→ **AR** → **SR** → Summary	NLR→ **SR** → **AR** → Summary
MATH	49.9	**66.7**
GSM8K	84.2	**88.7**

Table 4: Zero-shot Pass@1 results for varying paradigm orders on the MATH and GSM8k benchmarks. The best results are in **bold**.

resentative, on the MATH benchmark for arithmetic tasks. Unlike methods such as ToRA and NuminaMath, which rely on code without complete reasoning, CoR-Math-7B leverages multi-paradigm reasoning to outperform them across all benchmarks. For example, on the MATH dataset, CoR-Math-7B achieves an 11.4% absolute improvement over the tool-integrated NuminaMath-7B-TIR, underscoring the effectiveness of complete code-based reasoning over fragmented interleaving of natural language and code. To further explore the relationship between resource utilization (training sample size) and performance, we fit a quadratic function to the current SOTA models and perform a Pareto frontier analysis (Branke et al., 2008), as shown in Fig. 1 (Details in Appendix B.6). CoR-Math-7B outperforms the optimal performance of single-paradigm methods with similar data volume, suggesting a new optimal curve for multi-paradigm reasoning that surpasses single-paradigm approaches.

6 Ablation Study

6.1 Impact of Stages in PPT Method

We evaluate the PPT method using two base models, Llama-3.1-8B and DeepseekMath-7B-Base, on the MATH and GSM8K benchmarks at different PPT stages. As shown in Fig. 4, the vanilla models exhibit limited performance. After stage ①, performance improves significantly, with Llama-3.1-Base gaining 47.9% on MATH and 61.0% on GSM8K, emphasizing the need for mathematical understanding and model warming. Stage ②, utilizing NLR and AR data, yields limited gains compared to stage ① due to pre-training data already containing relevant content. However, this does not imply that stage ② is unimportant, as it plays a critical role in CoR by reactivating and refining computational skills essential for collaborating with other reasoning paradigms. Both stages ① and ② focus on reactivating models for mathematical tasks. In stage ③, training with three paradigms further enhances performance, suggesting that rare or unseen paradigms improve reasoning abilities. The consistent performance gains with added paradigms in Fig. 4 robustly demonstrate the superiority of our multi-paradigm PPT approach over single- or fewer-paradigm fine-tuning of the same base model, yielding a more powerful, comprehensive mathematical reasoner.

6.2 Order of the Reasoning Paradigms

As shown in Table 4, we investigate the impact of varying the sequence of reasoning paradigms in CoR-Math-7B during zero-shot arithmetic tasks, modifying only the prompt structure. Given that NLR closely aligns with the pre-training of LLMs, it serves as the first paradigm. The results indicate

Model	GSM8k@1	MATH@1	AMC2023 (Maj@64)	AIME2024 (Maj@64)	miniF2F@128
DSM + NLR	33.9	27.8	14/40	1/30	-
DSM + AR	75.8	37.6	10/40	0/30	-
DSM + SR	-	-	-	-	44.3
DSM + CoR (ours)	88.7	66.7	34/40	12/30	52.9

Table 5: Zero-shot results of multi- and single-paradigm fine-tuning on DeepSeekMath-7B-Base (DSM) for mathematical Reasoning benchmarks. The best results are in **bold**.

Model	Size	MATH	GSM8k	miniF2F
Qwen2.5-Math (Base)	1.5B	34.0	39.3	0.0
	7B	51.8	**90.0**	0.0
Qwen2.5-Math (CoR)	1.5B	57.6	84.5	51.6
	7B	64.7	**90.0**	52.5
Llama-3.1 (Base)	8B	4.2	6.2	25.8
	70B	16.8	20.5	22.5
Llama-3.1 (CoR)	8B	58.2	84.0	53.3
	70B	**70.7**	**90.0**	**56.2**

Table 6: Zero-shot Pass@1 results on the mathematical benchmarks across different model scales. The best scores are in **bold**.

that SR before AR resulted in the highest accuracy, with 66.7% on MATH, compared to 49.9% for AR before SR. A possible explanation is that SR plays a crucial role in decomposing the problem into manageable sub-steps, thereby providing a structured foundation for subsequent AR. This demonstrates that paradigms in CoR are interdependent, not isolated, and performance hinges on how they are linked during reasoning, beyond merely mastering individual paradigms. Furthermore, positioning AR preceding the Summary boxes may facilitate a more coherent integration of the computational outcome into the final answer.

6.3 Impact of Model Scales

Table 6 evaluates base model performance under the CoR framework with varying parameters, such as Qwen2.5-Math models (1.5B / 7B) and Llama-3.1-8B (8B / 70B). The results show that CoR scales with model size. The 7B Qwen2.5-Math outperforms the 1.5B version by 7.1% on MATH and 5.5% on GSM8K. Similarly, Llama-3.1 70B gains 2.9% over its 8B version on minF2F. This supports that CoR exhibits parameter scalability.

6.4 Multi- vs. Single-Paradigm Reasoning

To further validate the superior performance of multi-paradigm reasoning, we conducted an ablation experiment comparing it to single-paradigm fine-tuning We fine-tuned the DeepSeekMath-7B-Base model separately using single-paradigm paths (NLR, AR, SR) extracted from the MPM dataset. All experiments employed identical base models, problem sets, and training data sizes ($10,000$ samples per paradigm). Experimental results, summarized in Table 5, clearly demonstrate that our CoR model significantly outperforms single-paradigm fine-tuned models across diverse tasks. Specifically, for GSM8k and MATH datasets, CoR achieves improvements of 12.9% and 29.1%, respectively, compared to the best-performing single-paradigm approaches. On AMC2023 and AIME2024, CoR solves 20 and 11 more problems, respectively, compared to single-paradigm approaches. Lastly, on minF2F, CoR achieves an accuracy improvement of 8.6%. Collectively, these improvements underline CoR's effectiveness in integrating multiple reasoning paradigms, enabling robust and generalized problem-solving capabilities.

7 Conclusions

This paper introduces CoR, a novel unified reasoning framework that enhances mathematical reasoning in LLMs by synergistically integrating natural language, algorithmic, and symbolic. CoR tackles the limitations of single-paradigm approaches with the PPT strategy and the MPM dataset, enabling LLMs to progressively master diverse reasoning paradigms for improved generalization and performance. Our CoR-Math-7B outperformed SOTA models on five challenging benchmarks, showcasing enhanced zero-shot generalization. Ablation studies validated the benefits of progressive training and paradigm sequence. Additionally, CoR offers a new perspective on test-time scaling through multi-paradigm reasoning to enhance both efficiency and performance.

Acknowledgments

This work was partly supported by the Shenzhen Science and Technology Program (JSGG20220831110203007) and the research grant No. CT20240905126002 of the Doubao Large Model Fund.

Limitations

This paper proposes a general zero-shot reasoning method that supports different paradigms to solve multiple tasks. Evaluating zero-shot performance is challenging, as many methods rely on a single evaluation metric, like zero-shot pass@1. Additionally, most approaches focus on few-shot settings, as seen in benchmarks like miniF2F, limiting the comprehensiveness of our evaluation with aligned settings. To address this, we include additional experiments in Appendix C.2 to examine the impact of different evaluation strategies. On the other term, CoR can explore a multi-paradigm solution space, covering up to three paradigms, such as in large-scale arithmetic tasks. However, due to resource constraints and the fact that other methods can only predict within a single paradigm, we did not further consider using SMPS on benchmarks such as MATH to ensure fairness.

References

AI-MO. 2024a. Aime 2023. https://huggingface.co/datasets/AI-MO/aimo-validation-aime/.

AI-MO. 2024b. Amc 2023. https://huggingface.co/datasets/AI-MO/aimo-validation-amc/.

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen Marcus McAleer, Albert Q. Jiang, Jia Deng, Stella Biderman, and Sean Welleck. 2024. Llemma: An open language model for mathematics. In *ICLR*. OpenReview.net.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *AAAI*, pages 17682–17690. AAAI Press.

Jürgen Branke, Kalyanmoy Deb, Kaisa Miettinen, and Roman Slowinski, editors. 2008. *Multiobjective Optimization, Interactive and Evolutionary Approaches [outcome of Dagstuhl seminars]*, volume 5252 of *Lecture Notes in Computer Science*. Springer.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. 2023. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Trans. Mach. Learn. Res.*, 2023.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

Tri Dao. 2024. Flashattention-2: Faster attention with better parallelism and work partitioning. In *ICLR*. OpenReview.net.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Edward A Feigenbaum, Julian Feldman, et al. 1963. *Computers and thought*, volume 37. New York McGraw-Hill.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. PAL: program-aided language models. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pages 10764–10799. PMLR.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Minlie Huang, Nan Duan, and Weizhu Chen. 2024. Tora: A tool-integrated reasoning agent for mathematical problem solving. In *ICLR*. OpenReview.net.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the MATH dataset. In *NeurIPS Datasets and Benchmarks*.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *EMNLP*, pages 523–533. ACL.

Yinya Huang, Xiaohan Lin, Zhengying Liu, Qingxing Cao, Huajian Xin, Haiming Wang, Zhenguo Li, Linqi Song, and Xiaodan Liang. 2024. MUSTARD: mastering uniform synthesis of theorem and proof data. In *ICLR*. OpenReview.net.

Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Madry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino, Alex Renzin, Alex Tachard Passos, Alexander Kirillov, Alexi Christakis, Alexis Conneau, Ali Kamali, Allan Jabri, Allison Moyer, Allison Tam, Amadou Crookes, Amin Tootoonchian, Ananya Kumar, Andrea Vallone, Andrej Karpathy, Andrew Braunstein, Andrew Cann, Andrew Codispoti, Andrew Galu, Andrew Kondrich, Andrew Tulloch, Andrey Mishchenko, Angela Baek, Angela Jiang, Antoine Pelisse, Antonia Woodford, Anuj Gosalia, Arka Dhar, Ashley Pantuliano, Avi Nayak, Avital Oliver, Barret Zoph, Behrooz Ghorbani, Ben Leimberger, Ben Rossen, Ben Sokolowsky, Ben Wang, Benjamin Zweig, Beth Hoover, Blake Samic, Bob McGrew, Bobby Spero, Bogo Giertler, Bowen Cheng, Brad Lightcap, Brandon Walkin, Brendan Quinn, Brian Guarraci, Brian Hsu, Bright Kellogg, Brydon Eastman, Camillo Lugaresi, Carroll L. Wainwright, Cary Bassin, Cary Hudson, Casey Chu, Chad Nelson, Chak Li, Chan Jun Shern, Channing Conger, Charlotte Barette, Chelsea Voss, Chen Ding, Cheng Lu, Chong Zhang, Chris Beaumont, Chris Hallacy, Chris Koch, Christian Gibson, Christina Kim, Christine Choi, Christine McLeavey, Christopher Hesse, Claudia Fischer, Clemens Winter, Coley Czarnecki, Colin Jarvis, Colin Wei, Constantin Koumouzelis, and Dane Sherburn. 2024. Gpt-4o system card. *CoRR*, abs/2410.21276.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *CoRR*, abs/2310.06825.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts. *CoRR*, abs/2401.04088.

Haein Kong, Yongsu Ahn, Sangyub Lee, and Yunho Maeng. 2024. Gender bias in llm-generated interview responses. *CoRR*, abs/2410.20739.

Guillaume Lample, Timothée Lacroix, Marie-Anne Lachaux, Aurélien Rodriguez, Amaury Hayat, Thibaut Lavril, Gabriel Ebner, and Xavier Martinet. 2022. Hypertree proof search for neural theorem proving. In *NeurIPS*.

Chen Li, Weiqi Wang, Jingcheng Hu, Yixuan Wei, Nanning Zheng, Han Hu, Zheng Zhang, and Houwen Peng. 2024. Common 7b language models already possess strong math capabilities. *CoRR*, abs/2403.04706.

Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. 2024. Numinamath. https://github.com/project-numina/aimo-progress-prize.

Haohan Lin, Zhiqing Sun, Yiming Yang, and Sean Welleck. 2024a. Lean-star: Learning to interleave thinking and proving. *CoRR*, abs/2407.10040.

Zicheng Lin, Tian Liang, Jiahao Xu, Xing Wang, Ruilin Luo, Chufan Shi, Siheng Li, Yujiu Yang, and Zhaopeng Tu. 2024b. Critical tokens matter: Token-level contrastive estimation enhence llm's reasoning capability. *arXiv preprint arXiv:2411.19943*.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *CoRR*, abs/2308.09583.

Ruilin Luo, Zhuofan Zheng, Yifan Wang, Yiyao Yu, Xinzhe Ni, Zicheng Lin, Jin Zeng, and Yujiu Yang. 2025. Ursa: Understanding and verifying chain-of-thought reasoning in multimodal mathematics. *arXiv preprint arXiv:2501.04686*.

Frederic P. Miller, Agnes F. Vandome, and John McBrewster. 2009. *Levenshtein Distance: Information theory, Computer science, String (computer science), String metric, Damerau?Levenshtein distance, Spell checker, Hamming distance*. Alpha Press.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

OpenAI. 2024. Openai o1 system card. *Preprint*, arXiv:2412.16720.

Stanislas Polu and Ilya Sutskever. 2020. Generative language modeling for automated theorem proving. *CoRR*, abs/2009.03393.

Jiahao Qiu, Yifu Lu, Yifan Zeng, Jiacheng Guo, Jiayi Geng, Huazheng Wang, Kaixuan Huang, Yue Wu, and Mengdi Wang. 2024. Treebon: Enhancing inference-time alignment with speculative tree-search and best-of-n sampling. *CoRR*, abs/2410.16033.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: memory optimizations toward training trillion parameter models. In *SC*, page 20. IEEE/ACM.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton-Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2023. Code llama: Open foundation models for code. *CoRR*, abs/2308.12950.

Bilgehan Sel, Ahmad Al-Tawaha, Vanshaj Khattar, Ruoxi Jia, and Ming Jin. 2024. Algorithm of thoughts: Enhancing exploration of ideas in large language models. In *ICML*. OpenReview.net.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300.

Yuxuan Tong, Xiwen Zhang, Rui Wang, Ruidong Wu, and Junxian He. 2024. Dart-math: Difficulty-aware rejection tuning for mathematical problem-solving. *CoRR*, abs/2407.13690.

Shaobo Wang, Xiangqi Jin, Ziming Wang, Jize Wang, Jiajun Zhang, Kaixin Li, Zichen Wen, Zhong Li, Conghui He, Xuming Hu, and Linfeng Zhang. 2025. Data whisperer: Efficient data selection for task-specific llm fine-tuning via few-shot in-context learning. *Annual Meeting of the Association for Computational Linguistics*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *ICLR*. OpenReview.net.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*.

Sean Welleck and Rahul Saha. 2023. LLMSTEP: LLM proofstep suggestions in lean. *CoRR*, abs/2310.18457.

Zijian Wu, Suozhi Huang, Zhejian Zhou, Huaiyuan Ying, Jiayu Wang, Dahua Lin, and Kai Chen. 2024. Internlm2.5-stepprover: Advancing automated theorem proving via expert iteration on large-scale LEAN problems. *CoRR*, abs/2410.15700.

Huajian Xin, Z. Z. Ren, Junxiao Song, Zhihong Shao, Wanjia Zhao, Haocheng Wang, Bo Liu, Liyue Zhang, Xuan Lu, Qiushi Du, Wenjun Gao, Qihao Zhu, Dejian Yang, Zhibin Gou, Z. F. Wu, Fuli Luo, and Chong Ruan. 2024. Deepseek-prover-v1.5: Harnessing proof assistant feedback for reinforcement learning and monte-carlo tree search. *CoRR*, abs/2408.08152.

Yunfan Xiong, Ruoyu Zhang, Yanzeng Li, Tianhao Wu, and Lei Zou. 2024. Dyspec: Faster speculative decoding with dynamic token tree structure. *CoRR*, abs/2410.11744.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *CoRR*, abs/2409.12122.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. In *NeurIPS*.

Huaiyuan Ying, Zijian Wu, Yihan Geng, Jiayu Wang, Dahua Lin, and Kai Chen. 2024a. Lean workbook: A large-scale lean problem set formalized from natural language math problems. *CoRR*, abs/2406.03847.

Huaiyuan Ying, Shuo Zhang, Linyang Li, Zhejian Zhou, Yunfan Shao, Zhaoye Fei, Yichuan Ma, Jiawei Hong, Kuikun Liu, Ziyi Wang, Yudong Wang, Zijian Wu, Shuaibin Li, Fengzhe Zhou, Hongwei Liu, Songyang Zhang, Wenwei Zhang, Hang Yan, Xipeng Qiu, Jiayu Wang, Kai Chen, and Dahua Lin. 2024b. Internlm-math: Open math large language models toward verifiable reasoning. *Preprint*, arXiv:2402.06332.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *ICLR*. OpenReview.net.

Yiyao Yu, Junjie Wang, Yuxiang Zhang, Lin Zhang, Yujiu Yang, and Tetsuya Sakai. 2023. EALM: introducing multidimensional ethical alignment in conversational information retrieval. In *SIGIR-AP*, pages 32–39. ACM.

Yu-Xiang Zhang, Junjie Wang, Xinyu Zhu, Tetsuya Sakai, and Hayato Yamana. 2024. SSR: solving named entity recognition problems via a single-stream reasoner. *ACM Trans. Inf. Syst.*, 42(5):138:1–138:28.

Jinman Zhao and Xueyan Zhang. 2024. Large language model is not a (multilingual) compositional relation reasoner. In *First Conference on Language Modeling*.

Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. 2022. minif2f: a cross-system benchmark for formal olympiad-level mathematics. In *ICLR*. OpenReview.net.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V. Le, and Ed H. Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. In *ICLR*. OpenReview.net.

Xinyu Zhu, Junjie Wang, Lin Zhang, Yuxiang Zhang, Yongfeng Huang, Ruyi Gan, Jiaxing Zhang, and Yujiu Yang. 2023. Solving math word problems via cooperative reasoning induced language models. In *ACL (1)*, pages 4471–4485. Association for Computational Linguistics.

Appendix

A Discussion on Reasoning Hierarchy: Paradigms, Paths, and Steps

This section attempts to explore the essence of reasoning, contrasts CoR with current methods, examines the factors contributing to its effectiveness, and provides a new perspective for future research.

As shown in Fig. 5, this paper posits that reasoning texts generated by LLMs exhibit a reasoning hierarchical structure, which consists of three levels: reasoning paradigms, reasoning paths, and reasoning steps.

- **Reasoning steps** represent the fundamental units, each comprising one or more tokens and encompassing an incomplete stage of the solution process.
- **Reasoning paths** consist of several reasoning steps, forming a complete line of reasoning that typically includes a final answer and the solution process.
- **Reasoning paradigms** comprise one or more reasoning paths. They often contain multiple potential final answers, thus necessitating a summarization method, such as a summary module, to extract the ultimate answer. Furthermore, a reasoning paradigm uses a single knowledge media, such as natural language.

Furthermore, current studies can be categorized based on their focus. As shown in Fig. 5 (a), contemporary work concentrates within a single paradigm, optimizing along two dimensions: depth (the number of reasoning steps) and width (the number of reasoning paths). For instance, regarding reasoning depth, the CoT method (Wei et al., 2022) employs prompts to increase intermediate steps within one reasoning path to achieve higher performance. Concerning reasoning width, some approaches (Wang et al., 2023) involve altering sampling techniques to generate multiple distinct reasoning paths. Random sampling exemplifies this. Other studies (Zhu et al., 2023) employ scoring mechanisms for reasoning steps, guiding LLMs to generate several more desirable reasoning paths. Monte Carlo search illustrates this. Building upon these generated reasoning paths, existing work proposes various integration strategies to obtain the final answer, such as Best-of-N and Self-consistency (Wang et al., 2023). Specifically, as shown in Fig. 6 (a), recent advanced studies

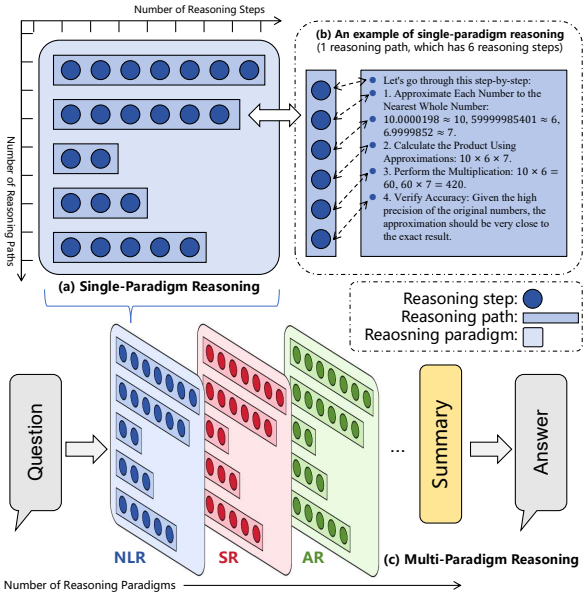

Figure 5: An overview of the reasoning hierarchical structure. (a) A single reasoning paradigm depicts multiple distinct reasoning paths. (b) An example of single-paradigm reasoning includes one reasoning path, which contains some reasoning steps. (c) Multi-paradigm reasoning includes several distinct reasoning paradigms.

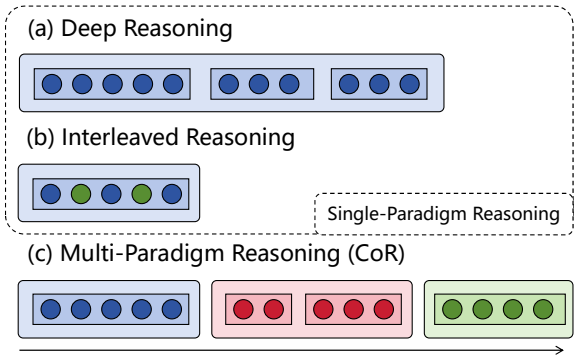

Figure 6: A comparison of reasoning methods in several advanced studies.

introduce the deep reasoning method, which focuses on generating serial concatenation of reasoning paths followed by summarization (like OpenAI o1 (OpenAI, 2024)). Since these methods are based on different optimization dimensions, we can combine them feasibly in applications.

Specifically, some approaches prioritize a dominant paradigm for reasoning yet hope to integrate other paradigms for guidance. For instance, as shown in Fig. 6 (b), the reasoning process of ToRA (Gou et al., 2024) presents an interleaved reasoning method with different knowledge media. It involves initial natural language generation, which is followed by code generation. Sub-

sequently, the process awaits the results of code execution before generating further natural language. InternLM2.5-StepProver (Wu et al., 2024) exhibits a similar pattern. It incorporates natural language annotations to support symbolic reasoning steps. However, these interleaved reasoning approaches do not constitute true multi-paradigm reasoning. The primary paradigm can independently achieve the final answer without the supplementary paradigms. Consequently, these methods imply a single-paradigm reasoning approach enhanced by other paradigms, rather than genuine multi-paradigm reasoning.

Current methods often *overlook the synergistic effects between paradigms and underestimate the importance of complete reasoning for achieving accurate results*. To address this challenge, as shown in Fig. 5 (c) and Fig. 6 (c), we introduce Chain-of-Reasoning (CoR), a unified reasoning framework capable of multi-paradigm reasoning. This framework embodies the following potential advantages:

Sequential reasoning dependency. As a type of multi-paradigm reasoning, CoR is not a mere accumulation of isolated steps; instead, it represents a coherent, interconnected process. The output from an earlier paradigm functions not only as input for a subsequent paradigm but also informs the foundational information for its reasoning. For instance, knowledge derived from a natural language paradigm guides subsequent algorithmic paradigms, thereby enhancing the efficiency and accuracy of code generation based on detailed natural language reasoning. This mechanism also provides substantial context from prior paradigms to later paradigms. It operates as a form of preliminary rehearsal where subsequent paradigms can follow correct reasoning steps or rectify incorrect ones.

A novel direction for test-time scaling emerges. We observe that the achievable improvements of single-paradigm reasoning are increasingly constrained by various search methods applied to the solution space. For instance, DeepSeek-Prover-V1.5-RL + RMaxTS requires 32×6400 reasoning paths to achieve 63.5% few-shot accuracy on the miniF2F benchmark. While extending the number of candidate solution paths seems a natural way to enhance the hit rate for ground truth solutions, the substantial computational effort involved underscores the inherent limitations of single-paradigm reasoning. This observation suggests that scaling up reasoning within a single paradigm during test time is inadequate for addressing complex tasks, such as mathematical problem solving. From a broader perspective, our CoR framework reexamines this challenge by shifting the focus of scaling efforts from reasoning paths to reasoning paradigms, thereby proposing a novel avenue for advancement.

Expanding the solution space. In multi-paradigm reasoning, the solution space is expanded by considering potential orthogonal relationships between reasoning paradigms. This allows generated solutions to be searched within different paradigms. Moreover, diverse solutions across paradigms can leverage chain relationships in reasoning to mutually inform each other. Therefore, our CoR framework enhances both intra-paradigm and inter-paradigm search capabilities within a large solution space. This diversity increases the likelihood of discovering optimal solutions.

Journey-based reasoning learning. CoR enhances models' learning trajectory, allowing them to perform deep reasoning both within specific paradigms and across multiple paradigms. This approach expands the "journey" by introducing the ability to navigate and collaborate across diverse reasoning paradigms.

Compatibility with existing methods. Since most current approaches are fundamentally based on a single paradigm, CoR can seamlessly integrate these methods within its specified paradigms, enabling them to work synergistically and leverage their strengths. For example, CoT can be incorporated as a specific implementation of the NLR paradigm in CoR. This implies that CoR serves as a platform that facilitates continuous integration and collaboration.

B Experiment Details

In this study, we use open-source tools, models, and datasets in compliance with their respective open-source licenses and solely for academic research purposes.

B.1 The detail of the Universal Text Template

As shown in Fig. 7, we apply the designed universal text template on all training samples.

B.2 Example Prompts for Dataset Enhancement

To guide LLMs in generating and refining reasoning paradigms during dataset enhancement, we de-

Universal Template Designed for Multi-paradigm Reasoning.

Problem:

[Statement of the problem]

Let's go through this step-by-step:

1. [Step 1 description]
2. [Step 2 description]
3. [Step 3 description]

$\cdots$ [Further steps as needed]

✓ [Verification or conclusion of the step-by-step reasoning]

Let's write the corresponding formal proof in Lean 4 to prove this:
Formal proof in Lean 4

```
[Lean 4 code for formal proof]
```

[Lean 4 Output]

Let's use Python to perform these calculations:
Code in Python

```
[Python code for calculation]
```

[Python Output]

Summary

[Summary of the solution and key takeaways]

Figure 7: The universal text template designed for multi-paradigm reasoning. The contents for each paradigm have been omitted.

Dataset	# of Test set	Avg. Length
MATH	5,000	30.7
GSM8K	1,319	46.3
AIME2024	30	59.2
AMC2023	40	47.2
MiniF2f	244	30.5

Table 7: The statistics of the evaluation datasets.

veloped specific prompts (p_s) tailored to each seed dataset. These prompts provide structured instructions for the models to follow, ensuring the generation of high-quality and logically consistent reasoning paradigms.

Fig. 8 and Fig. 9 provide examples of such prompts designed for augmenting and refining formal proofs within the Lean 4 theorem prover for the Numina-TIR and Lean-Workbook datasets, respectively. The prompt specifies the input and output format and includes placeholders for the problem statement, informal proof, and corresponding formal proof in Lean 4. It also incorporates N_{shot} few-shot examples to further guide the model. In our experiments, we follow the common settings (Jiang et al., 2023; Yang et al., 2024; Wang et al., 2025; Zhao and Zhang, 2024) and set $N_{\text{shot}} = 8$. This structured approach and setting are applied to all datasets to ensure comprehensive coverage and logical consistency across the enhanced dataset.

B.3 Details of Training Settings

In all stages of the PPT method, a learning rate of $2e-5$ and a warm-up ratio of 1% are implemented. To enhance computational efficiency, the training process is conducted using distributed optimization with DeepSpeed ZeRO Stage 3 (Rajbhandari et al., 2020) and is combined with Flash-Attention (Dao, 2024). Stage ① comprises 3 epochs, whereas the stage ② and ③ are conducted over 4 epochs. A sequence length of $2,048$ tokens is used in Stages ① and ②, which is increased to $4,096$ tokens in stage ③ to support more complex reasoning tasks. Furthermore, we employ an annealing strategy at the end of stage ③ with high-quality MPM samples, with the aim of further enhancing model accuracy in complex reasoning tasks.

B.4 Benchmarks

We report the statistics of the evaluation datasets in Table 7.

B.5 Details of Metrics

This study employs the pass@N metric on the miniF2F benchmark, with the evaluation grounded in a sample budget of N. To ensure a fair comparison of computational cost across different generation schemes, this paper defines the sample budget according to the following rules.

- For single-pass sampling methods, we define the sample budget N as the total number of proofs generated, with large values of N factorized for ease of comparison to tree search methods.
- For best-first-search methods, following the notation in Llemma (Azerbayev et al., 2024), we present $N = N_{\text{att}} \times N_{\text{tact}} \times N_{\text{iter}}$ where N_{att} denotes the number of best-first-search attempts, N_{tact} denotes the number of tactics generated for each expansion, and N_{iter} denotes the number of expansion iterations.
- For tree-based search methods, e.g., RMaxTS (Xin et al., 2024) and HTPS (Lample et al., 2022), we present $N = N_{\text{att}} \times N_{\text{ex}}$, where N_{att} denotes the number of tree search attempts, and N_{ex} denotes the number of model generations invoked in tree expansions.
- For our SMPS, we define the sample budget as $N = N_{\text{NLR}} \times N_{\text{SR}}$, where N_{NLR} denotes the number of initial semantic reasoning paths, and N_{SR} denotes the number of symbolic reasoning paths extended for each other reasoning paradigm.

B.6 Experimental Details of MATH and GSM8K benchmarks

For arithmetic-related results, we analyze the number of mathematical samples utilized by the baseline model following the pre-training phase, as illustrated in Fig. 1. Specifically, as shown in Table 8, we present the sample quantity along with the model performance.

C Additional Analysis

C.1 The Risk of Data Leakage

We measure the Levenshtein distance (Miller et al., 2009) between problems in the training dataset and those in the test dataset to mitigate the risk of data leakage. During the experiments, we set a similarity threshold of 0.7 and excluded cases from

Model	SFT Data Size (k)	MATH	GSM8k	Average
		ZS Pass@1	ZS Pass@1	
WizardMath	868	10.7	54.9	32.8
MetaMath-7B	2,790	19.8	66.5	43.2
MetaMath-Llemma-7B	2,790	30.0	69.2	49.6
MetaMath-Mistral-7B	2,790	28.2	77.7	53.0
ToRA	85	40.1	68.8	54.5
ToRA-CODE	85	44.6	72.6	58.6
NuminaMath-7B-CoT	859	54.4†	66.6†	60.5
Xwin-Math-7B	1,440	40.6	82.6	61.6
DeepSeekMath-Instruct-7B	776	46.8	73.6	64.2
NuminaMath-7B-TIR	931	55.3†	73.6†	64.5
DeepSeekMath-RL-7B	920	51.7	88.2	70.0
DART-Math-7B	1,175	53.6	86.6	70.1
Qwen2.5-Math-7B-Instruct	3,026	**83.6**	**95.2**	**89.4**
CoR-Math-7B	1,098	66.7	88.7	77.7

Table 8: Zero-shot (ZS) performance on the MATH and GSM8K benchmarks for the arithmetic task, with data size in the Supervised Fine-Tuning (SFT) stage. † denotes our reported results with the open-sourced model weights. The best results are in **bold**, and the second-best results are underlined.

Example Prompts for TIR Dataset Enhancement

System Prompt:
You are an expert in Lean 4. Please respond to a math problem by translating the provided informal proof into Lean 4 code. Follow the format provided in the prompt. Please note that the informal proof and the formal proof need to be identical. Follow the format provided in the prompt.
User Input:
Now please translate the formal solution in Lean 4 following the instructions below. Please write the corresponding solution in Lean 4 (indicated by "Formal proof in Lean 4: ") given the "# Problem: " and "# Informal proof: ", filling in the "# Formal proof in Lean 4: " section.

You must respond in the following format:
Problem: ...
Informal proof: ...
Formal proof in Lean 4:

```lean4
(lean 4 code for proving)
...
```

Here are examples you may refer to:

—
`N few-shot examples`
—
Problem: `{problem}`
Informal proof: `{informal_proof}`
Formal proof in Lean 4:

```lean4
...
```

Figure 8: Example prompts for Numina-TIR dataset enhancement. The few-shot examples provided for the LLM have been omitted.

Example Prompts for Lean-Workbook Dataset Enhancement

System Prompt:
You are an expert in Lean 4 and formal mathematics. Your task is to explain how to construct formal proofs using Lean 4 syntax and tactics. Focus on the Lean 4 approach to theorem proving, rather than traditional mathematical reasoning.
User Input:
Please follow the instructions below to convert the Lean 4 code (indicated by "Formal proof in Lean 4: ") into its informal proof, using the informal problem (indicated by "Problem: ") as a guide. Please write the corresponding informal solution in natural language (indicated by "Informal proof: ") given the "# Problem: " and "# Formal proof in Lean 4: ", filling in the "# Informal proof: " section.
<Instruction>
Analyze the given mathematical theorem and the corresponding Lean 4 code. Provide a detailed explanation of the proof process, adhering to the following guidelines:
1. Theorem structure: Clearly state the theorem, including its assumptions and conclusion.
2. Proof strategy: Explain the overall strategy employed in the proof, focusing on logical reasoning and mathematical deduction rather than calculation.
3. Step-by-step reasoning: Provide a detailed, step-by-step explanation of the proof process, ensuring that each step corresponds to an element in the Lean 4 code.
4. Logical deduction: Emphasize how each step of the proof follows logically from the previous steps or from the given assumptions.
5. Mathematical concepts: Discuss any specific mathematical concepts, notations, or definitions used in the proof, such as divisibility or exponentiation.
6. Abstraction: Present the proof in a general, abstract form that could be applied to similar problems, rather than focusing on specific numerical calculations.
7. Correspondence to code: Ensure that the logical flow of your proof explanation aligns with the structure and tactics used in the Lean 4 code, without explicitly mentioning Lean-specific terminology.
Avoid using syntax or terminology specific to formal proof systems. Instead, focus on presenting a rigorous mathematical argument using general logical principles and mathematical language. The proof should follow the reasoning path implied by the Lean 4 code but be accessible to readers unfamiliar with formal proof assistants.
</Instruction>
You must respond in the following format:
Problem: ...
Tags: ...
Formal proof in Lean 4:

```lean4
(lean 4 code for proving)
```

```
# Informal proof:
(Informal reasoning path for proving the problem)
```

Here are examples you may refer to:
—
`N few-shot examples`
—
Problem: `{problem}`
Tags: `{tag}`
Formal proof in Lean 4:

```lean4
{lean_workbook code}
```

```
# Informal proof: ...
```

Figure 9: Example pompts for Lean-Workbook dataset enhancement. The few-shot examples provided for the LLM have been omitted.

Model	MATH		GSM8k	
	Pass@1	Maj@8	Pass@1	Maj@8
Llama-3.1-8B	58.18	59.46	84.00	87.26
DeepSeekMath-Base-7B	66.74	71.70	88.70	91.43

Table 9: Zero-shot results of different evaluation strategies across MATH and GSM8k benchmarks.

the training dataset that exceeded this threshold. As a result, approximately $1,000$ cases, accounting for less than 0.6% of our training dataset, were removed to prevent potential leakage.

C.2 Different Evaluation Strategies on Arithmetic Benchmarks

Table 9 explores the impact of various evaluation strategies on the CoR framework. The majority vote strategy benefits the CoR framework. Specifically, a majority vote strategy with 8 candidate samples on GSM8K can exceed GPT-4o's Pass@1 performance (90.5%).

C.3 Ethical Considerations

In this work, we present a method that achieves robust zero-shot multi-paradigm reasoning capabilities. However, this comes with potential risks, which include two primary social and ethical concerns: 1) the risk of tool misuse, and 2) pre-existing risks embedded in the backbone model parameters. The CoR framework supports various reasoning paradigms, some of which are deeply integrated with specific tools. For instance, the AR paradigm may involve unvetted Python code libraries. Therefore, we recommend performing pre-deployment audits on tools that may be utilized by the model. Furthermore, our method is compatible with existing LLMs, and several studies have highlighted the presence of biased behaviors in models like LLaMa (Kong et al., 2024). To mitigate ethical risks, we encourage the use of risk-free language models during deployment, which can reduce potential harm. Recent research suggests incorporating ethical alignment processes, such as filtering training data or generated content, to minimize unnecessary risks (Yu et al., 2023). Overall, we advocate for open discussions regarding the use of our framework, as a transparent and public environment can reduce the likelihood of misuse.

D Case Studies

D.1 Cases of Instruction-free Reasoning

In the absence of explicit instructions, as illustrated in Fig. 10, the model generally adopts a paradigm sequence that aligns with the task types present in the training data, as described by MPM. The model transitions between paradigms autonomously. For example, during a computational task, the model initiates reasoning with the paradigm NLR, proceeds to SR for logical verification, and ultimately employs AR to achieve enhanced computational performance.

D.2 Qualitative Analysis of Error Cases

We now present a preliminary error pattern analysis for CoR-Math-7B across two tasks: arithmetic computation and theorem proving.

In arithmetic computation, an analysis of 200 randomly sampled error cases reveals four primary types of errors: comprehension errors or incorrect premise assumptions (33 cases), logical fallacies (26 cases), computational errors (56 cases), and incomplete proofs or code syntax errors, which were most prevalent in the AR/SR paradigms (85 cases). Notably, cross-paradigm self-correction behaviors were observed. For example, AR rectified erroneous calculations initially produced by NLR, and SR supplemented incomplete logical steps that were missing in NLR outputs. In theorem proving, a separate analysis of 50 randomly sampled error cases revealed the same four error types, with the following distribution: 2 comprehension errors, 6 logical fallacies, 9 computational errors, and 33 incomplete proofs or syntax errors. Cross-paradigm self-correction behaviors were again consistently observed. As shown in Fig. 11 and Fig. 12, we provide an example of error cases and an example of self-correction behavior.

We attribute these errors to the following potential causes. Despite MPMs systematic integration of multi-paradigm reasoning paths and PPTs paradigm-progressive training as discussed in Sec. 3.2 and Sec. 3.4 to mitigate distribution shifts, the overwhelming prevalence of NLR examples in the pretraining data —particularly the scarcity of SR-specific instances —has resulted in persistent competency gaps across paradigms. Additionally, CoR inherits the base LLMs fixed-context window, which introduces the risk of truncating long reasoning chains or omitting key steps, leading to incomplete outputs. Fur-

thermore, the observed self-correction capabilities can be attributed to the complementarity of reasoning paradigms within CoR, where one reasoning paradigm can rectify errors made by another, thus validating the effectiveness of our multi-paradigm approach. Additionally, feedback from multi-step reasoning validation allows subsequent paradigms to verify earlier outputs, facilitating error detection and correction throughout the reasoning process.

Examples of Instruction-free Reasoning Generated by CoR

Problem:
Cities A and B are 45 miles apart. Alicia lives in A and Beth lives in B. Alicia bikes towards B at 18 miles per hour. Leaving at the same time, Beth bikes toward A at 12 miles per hour. How many miles from City A will they be when they meet?

Solution:
Let's go through this step-by-step in natural language. To determine how many miles from City A Alicia and Beth will meet, we can use the concept of relative speed and the distance between their homes.
1. Calculate the relative speed between Alicia and Beth:
- Alicia's speed: 18 miles per hour
- Beth's speed: 12 miles per hour
- Relative speed = 18 + 12 miles per hour
2. Calculate the time it takes for them to meet:
- Distance between City A and City B: 45 miles
- Time to meet = Distance ÷ Relative speed
3. Calculate the distance each person covers before meeting:
- Distance covered by Alicia = Speed of Alicia × Time to meet
- Distance covered by Beth = Speed of Beth × Time to meet.

Next, let's write the corresponding formal proof in Lean 4 to prove this.
Formal proof in Lean 4:

```lean4
def distance_AB := 45  -- miles
def speed_Alicia := 18  -- miles per hour
def speed_Beth := 12  -- miles per hour
-- Calculate the relative speed
def relative_speed := speed_Alicia + speed_Beth  -- miles per hour
-- Calculate the time to meet
def time_to_meet := distance_AB.toFloat / relative_speed.toFloat  -- in hours
-- Calculate the distance each person covers before meeting
def distance_covered_by_Alicia := speed_Alicia.toFloat * time_to_meet
def distance_covered_by_Beth := speed_Beth.toFloat * time_to_meet
-- Calculate how many miles from City A they will be when they meet
def distance_from_A_when_meet := distance_covered_by_Alicia
#eval distance_from_A_when_meet  -- returns 27.0
```
```lean_output
27.000000
```
Let's use Python code to perform these calculations.
Code in Python:

```python
import sympy as sp
# Given values
distance_AB = 45  # miles
speed_Alicia = 18  # miles per hour
speed_Beth = 12  # miles per hour
# Calculate the relative speed
relative_speed = speed_Alicia + speed_Beth  # miles per hour
# Calculate the time to meet
time_to_meet = distance_AB / relative_speed  # in hours
# Calculate the distance each person covers before meeting
distance_covered_by_Alicia = speed_Alicia * time_to_meet
distance_covered_by_Beth = speed_Beth * time_to_meet
# Calculate how many miles from City A they will be when they meet
distance_from_A_when_meet = distance_covered_by_Alicia
print(distance_from_A_when_meet)
```
```output
27.0
```
Alicia and Beth will meet boxed27 miles from City A.

Figure 10: A case of the reasoning trajectory produced by CoR in an instruction-free setting.

Examples of Incomplete Proofs or Code Syntax Errors

Problem:
What is the value of $\sqrt{1{,}000{,}000} - \sqrt[3]{1{,}000{,}000}$? Show that it is 900.

Solution:
Let's go through this step-by-step in natural language.
We need to prove the following theorem:

```
theorem mathd_algebra_208 : Real.sqrt 1000000 - 1000000 ^ ((1 : ) / 3) = 900 := by
```

We have the following:

$$\begin{aligned}\sqrt{1,000,000} - \sqrt[3]{1,000,000} &= \sqrt{10^6} - \sqrt[3]{10^6} \\ &= (10^6)^{\frac{1}{2}} - (10^6)^{\frac{1}{3}} \\ &= 10^{6\cdot\frac{1}{2}} - 10^{6\cdot\frac{1}{3}} \\ &= 10^3 - 10^2 = 1000 - 100 = 900.\end{aligned}$$

So, the value of $\sqrt{1,000,000} - \sqrt[3]{1,000,000}$ is indeed 900.
Next, let's write the corresponding formal proof in Lean 4 to prove this.
Formal proof in Lean 4:

```lean4
theorem mathd_algebra_208 : Real.sqrt 1000000 - 1000000 ^ ((1 : ) / 3) = 900 := by
norm_num [sqrt_eq_iff_mul_self_eq_of_pos, pow_one]
```

```lean_output
Execution Failed.
```

The original code has a syntax error. The proof block after := by is incomplete. Proof failed.

Figure 11: A proof task failure case caused by incomplete proofs or code syntax errors.

Examples of Instruction-free Reasoning Generated by CoR

Problem:
Let $b \geq 2$ be an integer. Call a positive integer n *b-eautiful* if it has exactly two digits when expressed in base b, and these two digits sum to $\sqrt{n}$.
For example, 81 is 13-*eautiful* because $81 = \underline{6}\,\underline{3}_{13}$ and $6 + 3 = \sqrt{81}$.
Find the least integer $b \geq 2$ for which there are more than ten b-*eautiful* integers.

Solution:
Let's go through this step-by-step in natural language. To solve this problem, we need to find the smallest integer $b \geq 2$ such that there are more than ten b-beautiful integers.
1. Understanding the Problem:
- We need to identify b-beautiful integers, which have exactly two digits in base b.
- These integers n must satisfy the condition that the sum of their digits equals $\sqrt{n}$.
...
6. Conclusion:
- The least integer $b \geq 2$ for which there are more than ten b-beautiful integers is $b = 13$.
Therefore, the smallest b such that there are more than ten b-beautiful integers is $b = 13$.

Next, let's write the corresponding formal proof in Lean 4 to prove this.
Formal proof in Lean 4:

```lean4
theorem lean_workbook_plus_1377 (b : ) (hb : 2  b)
(h : ( n, (Nat.sqrt n = n  Nat.digits b n = [n / b, n % b]))
   ň ( n, (Nat.sqrt n = n  Nat.digits b n = [n / b, n % b]))) : 13  b :=
  by
  contrapose! h
  interval_cases b <;> norm_num [Nat.digits_eq_of_eq_mul_left, Nat.digits_eq_of_eq_mul_right]
```
```lean_output
27.000000
```
Let's use Python code to perform these calculations.
Code in Python:

```python
import sympy as sp
import math
def is_b_beautiful(n, b):
\"\"\"Check if a number n is b-beautiful.\"\"\"
digits = [n // b, n % b]
return sum(digits) == math.sqrt(n)
def count_b_beautiful_numbers(b):
\"\"\"Count the number of b-beautiful numbers for a given b.\"\"\"
count = 0
for n in range(1, b * b):
if is_b_beautiful(n, b):
count += 1
return count
def find_smallest_b_with_more_than_ten_beautiful_numbers():
\"\"\"Find the smallest b >= 2 such that there are more than ten b-beautiful numbers.\"\"\"
b = 2
while True:
if count_b_beautiful_numbers(b) > 10:
return b
b += 1
# Find the smallest b
smallest_b = find_smallest_b_with_more_than_ten_beautiful_numbers()
print(smallest_b)
```
```output
211
```
The smallest integer $b \geq 2$ for which there are more than ten b-beautiful integers is $b = 211$. Thus, the answer is $\boxed{211}$.

Figure 12: A case of self-correction behaviors. Some lengthy content is omitted with "..." for brevity.