War of Thoughts: Competition Stimulates Stronger Reasoning in Large Language Models

Yibin Chen[1*], **Jinyi Liu**[2*], **Yan Zheng**[1†], **Yifu Yuan**[2], **Jianye Hao**[2]
[1]School of New Media and Communication, Tianjin University, Tianjin, China
[2]College of Intelligence and Computing, Tianjin University, Tianjin, China
{yibin_chen,jyliu,yanzheng,yuanyf,jianye.hao}@tju.edu.cn

Abstract

Recent advances in Large Language Models (LLMs) have reshaped the landscape of reasoning tasks, particularly through test-time scaling (TTS) to enhance LLM reasoning. Prior research has used structures such as trees or graphs to guide LLMs in searching for optimal solutions. These methods are time-consuming and require a strong reward model (RM) to support effective solution space exploration. Tournament-style approaches eliminate the reliance on RMs through comparative evaluation but suffer from transitivity dilemmas, leading to unstable ordering. To address these issues, we propose *War of Thoughts* (**WoT**), a novel post-hoc method that enhances reasoning without finetuning. WoT comprises two distinct stages: (1) *Exploration*, in which diverse and meaningful candidate solutions are generated through contrastive demonstrations and multi-granularity reasoning specifications; and (2) *Competition*, where these candidate solutions are subjected to multiple rounds of matchups within a competitive arena. Throughout this iterative process, the solutions are optimized and improved, with the optimal solution being determined based on Elo ratings. Extensive experiments across various LLMs demonstrate the superiority of WoT, surpassing baselines by 10–30%. WoT can effectively stimulate stronger reasoning abilities, achieving impressive TTS performance in both generation budget and model size. It shows higher scalability efficiency compared to the baseline within the same budget. Notably, WoT exhibits excellent scalability with model size, even outperforming a 72B model despite using a 7B model.

1 Introduction

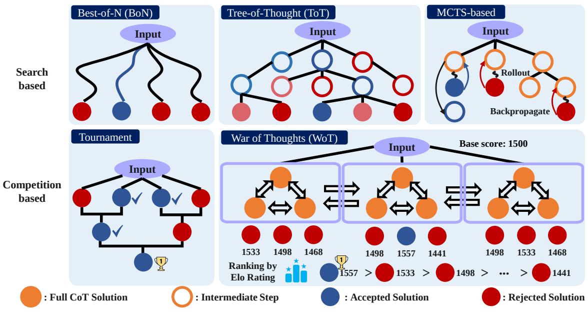


Figure 1: Comparison of different reasoning enhancement strategies at test time. (1) Search-based methods are time-consuming and rely on a strong reward model. (2) Tournament-style approaches assume transitivity in comparative evaluations, which may not always hold. (3) Our WoT enhances diversity of solutions in the *Exploration* stage, and introduces Elo ratings to effectively select the optimal solution in the *Competition* stage.

Large language models (LLMs) (Brown et al., 2020; Chowdhery et al., 2023; Zeng et al., 2023; Touvron et al., 2023; Achiam et al., 2023) and their applications (Zheng et al., 2019; Wu et al., 2023; Chen et al., 2024c; Shen et al., 2024) have shown remarkable capabilities across various fields, particularly when integrated with Chain-of-Thought (CoT) (Wei et al., 2022), ReAct (Yao et al., 2023) Tree-of-Thought (ToT) (Yao et al., 2024) and other prompting techniques (Wang et al., 2023c,a; Besta et al., 2024; Ding et al., 2024). However, LLMs still face significant challenges in complex reasoning (Valmeekam et al., 2022; Weng et al., 2023; Mirzadeh et al., 2024). Recent progress has been made in enhancing the advanced reasoning capabilities of LLMs. One paradigm is to internalize reasoning skills by fine-tuning LLMs, for instance, by using large amounts of high-quality reasoning data for Supervised Fine-tuning (SFT) (Yu et al., 2024; Tong et al., 2024; Li et al., 2024a). Additionally, LLMs can serve as both generators and verifiers, bootstrapping reasoning paths by Reinforcement Learning (RL) (Trung et al., 2024; Cheng et al., 2024; Havrilla et al., 2024; Guo et al., 2025). However, this paradigm requires substantial training resources and incurs high data collection costs. An-

[*]Both authors contributed equally to this research
[†]Corresponding to: Yan Zheng (yanzheng@tju.edu.cn)

Findings of the Association for Computational Linguistics: ACL 2025, pages 21716–21737
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

other paradigm involves scaling the computational resources during inference to improve LLM reasoning, namely Test Time Scaling (TTS). Empirically, applying more computation at test time can enhance LLM performance beyond training levels (Snell et al., 2024). Thus, this work focuses on improving LLMs' complicated reasoning through TTS via effective design.

Existing post-hoc strategies for improving the reasoning capabilities of LLMs can be mainly divided into two categories: (1) **search-based** methods and (2) **competition-based** methods, as illustrated in Figure 1. Search-based methods, such as Best-of-N (BoN) (Stiennon et al., 2020), involve sampling N responses from an LLM and choosing the best response based on the evaluation of RM. ToT, GoT, and MCTS-based approaches (Zhou et al., 2024; Qi et al., 2024; Zhang et al., 2024b) decompose problem-solving into multiple steps. Intermediate steps are assessed via PRM or specified scoring strategies, and the optimal solution is chosen as the final answer. However, these approaches often fail to fully explore the solution space, getting trapped in low-quality reasoning steps due to limited reflection mechanisms. For instance, increasing sample size in BoN may lead to diminishing returns and even negative yield. Besides, a superior RM matters; without accurate rewards, ToT and MCTS-based methods struggle to determine the promising solution. However, a powerful RM requires deliberate annotation (Lightman et al., 2024), which consumes expensive resources. Moreover, when employing heuristics for reward assignment, even the strongest LLMs can suffer from systematic biases (Wang et al., 2023b; Thakur et al., 2024) and overconfidence (Xiong et al., 2024).

An intuitive solution is to replace scoring with comparison to lessen the burden on LLMs when handling complicated scoring criteria. By directly comparing the pros and cons of different solutions, LLMs can make more intuitive judgments. This focuses on assessing relative differences, which avoids the complexity of assigning absolute scores and reduces systematic biases. Son et al., 2024 proposed a single-elimination tournament framework to rank different LLM outputs. Such design inherently assumes transitivity—if A beats B and B beats C, then A is expected to beat C. Nevertheless, this assumption does not always hold, especially when one model significantly outperforms a second, which is closely matched with a third. Therefore, This approach is unstable for determining the best solution from a single LLM.

To address the aforementioned issues, we introduce War of Thoughts (WoT), a novel TTS approach that improves reasoning without finetuning. WoT consists of two main stages: **(1) Generate Diverse and Meaningful Grouped Kickoff Solutions** (§ 3.1) and **(2) Select the Optimal Solution via Iterative Competition** (§ 3.2). In stage 1, we identify common types of errors in problem-solving and construct contrastive CoT demonstrations to prevent the LLM from repeating similar mistakes during reasoning. Meanwhile, we define multi-granularity reasoning specifications to maximize the reasoning boundaries of LLMs. Through these two levels of guidance, we hope to enable the LLM to generate multiple meaningful candidate responses, with each response focusing on a diverse path of reasoning. In stage 2, we group multiple candidates and have them compete in the arena, using multi-round matchups to alleviate positional bias in comparative evaluations. Through matchup reviews, the solutions are iteratively refined and improved. Moreover, we incorporate Elo rating, which transforms unstable partial orders into more reliable, quantifiable rankings.

Extensive experiments (§ 4) across different LLMs and varying reasoning-intensive benchmarks demonstrate the superiority of WoT, achieving a significant improvement of 10-30% over baselines such as BoN, ToT, RAP, and rStar. We also provide additional analysis to comfirm the impressive scalability of WoT (§ 5), revealing that WoT scales more effectively under the same generation budget. Furthermore, the 7B model with WoT outperforms the 72B model despite a 10× more parameter discrepancy, highlighting WoT's high scalability with respect to model size.

2 Related Work

Reasoning with LLMs. Recent years have seen a rapid improvement in LLM performance on reasoning tasks (Lewkowycz et al., 2022; Lightman et al., 2024; Achiam et al., 2023; Shao et al., 2024; Team et al., 2024). Several key factors have contributed to these advancements: (1) running continuous pretraining on large corpus of reasoning tasks (Shao et al., 2024; Team et al., 2024); (2) utilizing a well-pretrained LLM to synthesis data and progressively enhancing performance via SFT (Wan et al., 2024; Chen et al., 2024a; Zhang et al., 2024a; Huang et al., 2023; Xu et al., 2024); (3) leveraging ad-

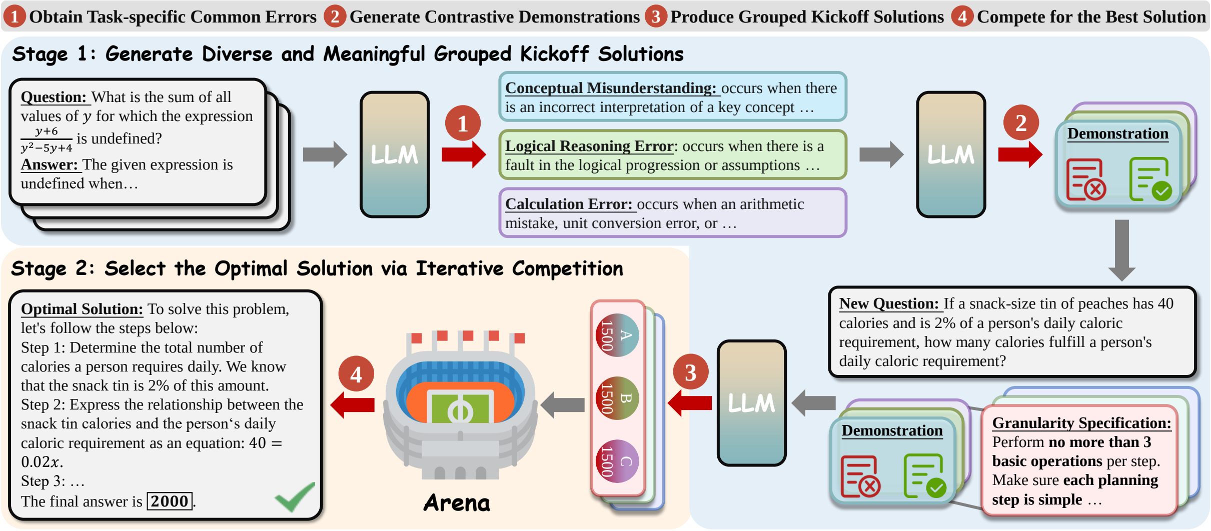

1 Obtain Task-specific Common Errors
2 Generate Contrastive Demonstrations
3 Produce Grouped Kickoff Solutions
4 Compete for the Best Solution
Stage 1: Generate Diverse and Meaningful Grouped Kickoff Solutions
Question: What is the sum of all values of y for which the expression $\frac{y+6}{y^2-5y+4}$ is undefined?
Answer: The given expression is undefined when…
LLM
1
Conceptual Misunderstanding: occurs when there is an incorrect interpretation of a key concept …
Logical Reasoning Error: occurs when there is a fault in the logical progression or assumptions …
Calculation Error: occurs when an arithmetic mistake, unit conversion error, or …
LLM
2
Demonstration
Stage 2: Select the Optimal Solution via Iterative Competition
New Question: If a snack-size tin of peaches has 40 calories and is 2% of a person's daily caloric requirement, how many calories fulfill a person's daily caloric requirement?
Optimal Solution: To solve this problem, let's follow the steps below:
Step 1: Determine the total number of calories a person requires daily. We know that the snack tin is 2% of this amount.
Step 2: Express the relationship between the snack tin calories and the person's daily caloric requirement as an equation: 40 = 0.02x.
Step 3: …
The final answer is $\boxed{2000}$.
4
Arena
A 1500
B 1500
C 1500
3
LLM
Demonstration
Granularity Specification: Perform no more than 3 basic operations per step. Make sure each planning step is simple …

Figure 2: Overview of WoT. First, the LLM identifies common error types using a sub-dataset and creates contrastive demonstrations accordingly. In stage 2, multiple candidates engage in intra-group competition, and the candidate with the highest Elo score is selected as the winner.

vanced prompting techniques, such as ToT (Yao et al., 2024), RAP (Hao et al., 2023), and rStar (Qi et al., 2024), which improve performance through self-exploration at inference time. However, these methods require large amounts of high-quality data or rely on a powerful RM, limiting their generalizability. Our approach introduces an iterative competition framework, which bypasses RMs through comparative evaluation and more effectively enhances LLM reasoning at test time.

Answer Selection. Majority voting (Wang et al., 2023c) is a widely used method for selecting the correct reasoning trajectory. Several works (Wang et al., 2024b; Chen et al., 2024a) train RMs for verification, but these require additional annotations and suffer from limited generalizability. Other research (Li et al., 2025; Xie et al., 2024) designs criteria and prompts LLMs to score, which can lead to overconfidence (Xiong et al., 2024; Zhang et al., 2024c). Comparative evaluation has been employed to address this issue (Son et al., 2024; Raina et al., 2025). However, it neglects positional bias (Wang et al., 2023b; Thakur et al., 2024) and the lack of transitivity (Zheng et al., 2023). Our method introduce an Elo-based rating system and use multi-round matchups to mitigate these issues.

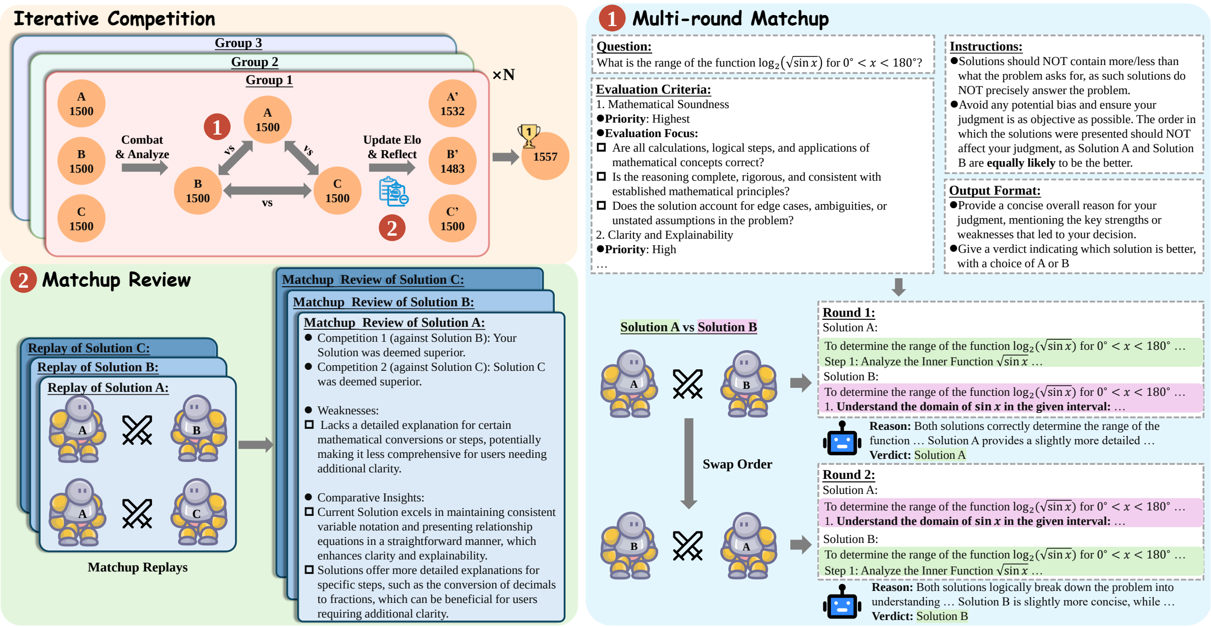
3 WoT

As illustrated in Figure 2, WoT consists of two main stages: **(1) Generate Diverse and Meaningful Grouped Kickoff Solutions** and **(2) Select the Optimal Solution via Iterative Competition**. In stage 1, we generate kickoff solutions by constructing various task-specific contrastive demonstrations and defining different reasoning granularity specifications. This enhances the diversity and significance of the solutions from the level of problem-solving perspective and granularity. In stage 2, we introduce the Elo rating system to build a reliable ranking system. Grouped candidates are engaged in pairwise matchups, update their Elo scores through comparative evaluation, and improve the solution quality through reflection in an iterative process. The solution with the highest Elo score is finally selected as the optimal solution.

3.1 Generate Diverse and Meaningful Grouped Kickoff Solutions

For BoN/CoT-SC, as the number of samples increases, the solutions tend to become more homogeneous, leading to diminishing returns. To more efficiently enhance reasoning abilities, in this stage, we generate more diverse and meaningful initial solutions through contrastive demonstrations and multi-granularity reasoning specifications.

Contrastive Demonstrations LLMs make various errors when solving problems, and we aim for different solutions to avoid these errors, thereby broaden the problem-solving perspectives of LLMs and enhancing diversity. To this end, we introduce contrastive demonstrations. Let $\mathcal{D} = \{(Q_i, A_i)\}_{i=1}^{n}$ represent a sub-dataset containing n samples, where each sample consists of a question Q_i and its corresponding answer A_i. We aim for



Matchup Review of Solution A:

- Competition 1 (against Solution B): Your Solution was deemed superior.
- Competition 2 (against Solution C): Solution C was deemed superior.

- Weaknesses:

□ Lacks a detailed explanation for certain mathematical conversions or steps, potentially making it less comprehensive for users needing additional clarity.

- Comparative Insights:

□ Current Solution excels in maintaining consistent variable notation and presenting relationship equations in a straightforward manner, which enhances clarity and explainability.

□ Solutions offer more detailed explanations for specific steps, such as the conversion of decimals to fractions, which can be beneficial for users requiring additional clarity.

Question:
What is the range of the function $\log_2(\sqrt{\sin x})$ for $0° < x < 180°$?

Evaluation Criteria:
1. Mathematical Soundness
●**Priority**: Highest
●**Evaluation Focus:**
□ Are all calculations, logical steps, and applications of mathematical concepts correct?
□ Is the reasoning complete, rigorous, and consistent with established mathematical principles?
□ Does the solution account for edge cases, ambiguities, or unstated assumptions in the problem?
2. Clarity and Explainability
●**Priority**: High
…

Instructions:
●Solutions should NOT contain more/less than what the problem asks for, as such solutions do NOT precisely answer the problem.
●Avoid any potential bias and ensure your judgment is as objective as possible. The order in which the solutions were presented should NOT affect your judgment, as Solution A and Solution B are **equally likely** to be the better.

Output Format:
●Provide a concise overall reason for your judgment, mentioning the key strengths or weaknesses that led to your decision.
●Give a verdict indicating which solution is better, with a choice of A or B

Round 1:
Solution A:
To determine the range of the function $\log_2(\sqrt{\sin x})$ for $0° < x < 180°$ …
Step 1: Analyze the Inner Function $\sqrt{\sin x}$ …
Solution B:
To determine the range of the function $\log_2(\sqrt{\sin x})$ for $0° < x < 180°$ …
1. **Understand the domain of sin x in the given interval:** …

Reason: Both solutions correctly determine the range of the function … Solution A provides a slightly more detailed …
Verdict: Solution A

Round 2:
Solution A:
To determine the range of the function $\log_2(\sqrt{\sin x})$ for $0° < x < 180°$ …
1. **Understand the domain of sin x in the given interval:** …
Solution B:
To determine the range of the function $\log_2(\sqrt{\sin x})$ for $0° < x < 180°$ …
Step 1: Analyze the Inner Function $\sqrt{\sin x}$ …

Reason: Both solutions logically break down the problem into understanding … Solution B is slightly more concise, while …
Verdict: Solution B

Figure 3: An illustration of iterative competition. In each group, candidates compete in pairwise multi-round matchups. The judge LLM provides a verdict with reason for each round. Candidates reflect on their performance and make improvements via mathcup reviews, while their Elo scores are updated accordingly. The candidate with the highest Elo score is selected, with its solution deemed the optimal one.

the LLM $\mathcal{M}$ to identify the top-k most common errors in problem-solving, denoted as $\mathcal{E} = \mathcal{M}(\mathcal{D})$, where $|\mathcal{E}| = k$. Each error $e_i \in \mathcal{E}$ includes a description of the corresponding mistake. Figure 2 illustrates this process using the MATH training set as an example, identifying three common errors. Once task-specific common errors $\mathcal{E}$ are obtained, for a given problem Q', we prompt the LLM to generate a problem $\hat{Q}$ similar to Q'. We then create k contrastive CoT demonstration pairs $\mathcal{S} = \left\{(s_i^+, s_i^-)\right\}_{i=1}^k = \mathcal{M}(\hat{Q}, \mathcal{E})$. Each negative demonstration s_i^- incorporates an error pattern e_i, potentially leading to an incorrect solution. These demonstration pairs are created both to increase the diversity of solutions and to encourage the LLM to avoid making similar errors during reasoning.

Multi-granularity Reasoning Specifications We further enhance the diversity and significance of the initial solutions by considering another dimension–the reasoning granularity. This involves introducing the concept of Reasoning Boundary (RB) (Chen et al., 2024b). The *Combination Law of RB* reveals the collaboration of different capabilities through CoT enhances LLM performance. For example, in mathematical reasoning, CoT may involve step planning and step calculation (Tan, 2023; Xiao and Liu, 2024) with individual RBs. By balancing planning and calculation, computational efficiency at each step can be maximized. We design different specifications $\mathcal{P} = \{p_i\}_{i=1}^t$ to explicitly control the granularity of planning and calculation in the CoT process, thereby increasing the diversity and significance of initial solutions from the perspective of reasoning granularity.

Based on the specifications $\mathcal{P}$ and demonstrations $\mathcal{S}$, t groups are curated, denoted as $\mathcal{G}$. In each group, we set up k LLM candidates, The initial solution for each candidate is given by $g_{ij} = \mathcal{M}_{ij}(p_i, e_j)$, where $i = 1, \ldots, t$ and $j = 1, \ldots, k$.

3.2 Select the Optimal Solution via Iterative Competition

After dividing the groups and generating diverse kickoff solutions, we allow these candidates to enter the arena for iterative competition to determine the best one. Figure 3 illustrates the process of this iterative competition. For simplicity, we assume each group g_i consists of three candidates, namely A, B, and C. A performance ranking system relies on *transitivity* (Zheng et al., 2023), which means if A beats B and B beats C, then A should beat C. However, this condition may not hold in LLM evaluations (Boubdir et al., 2023). To alleviate this issue, all candidates within each group engage in pairwise matchups. Additionally, we incorporate Elo rating to convert this unstable partial order into reliable and quantifiable rankings.

❶ **Multi-round Matchup** We first introduce a judge LLM, denoted as $\mathcal{J}$, to assess which candidate is superior. A set of evaluation criteria is defined based on the category of problem being solved. We require the judge LLM to provide its verdict and the corresponding reason, which highlights the key strengths or weaknesses that led to the decision. Multiple rounds of competition are then held to alleviate positional bias in comparative evaluations (Li et al., 2024b; Wang et al., 2024a). Specifically, we first set the order of candidates such that A's solution is placed before B's, generating the first-round verdict $v^1 = \mathcal{J}(A, B)$. Then, we swap the order of A and B to generate the second-round verdict $v^2 = \mathcal{J}(B, A)$. If v^1 and v^2 conflict (e.g., $v^1 = A \succ B$ and $v^2 = A \prec B$), a third-round matchup is introduced. In this third, the judge gives the final verdict v^3 based on the previous two verdicts and their respective reasons. According to the verdict, we update each candidate's Elo score. We specify this process in appendix B.

❷ **Matchup Review** After the matchups, we extract each candidate's records from the pairwise competitions as matchup replays and instruct the LLM to generate a matchup review for each candidate, with the candidate as the main subject. This review clearly outlines the win/loss outcome, any weaknesses observed, and insights gained from the competitions. Each candidate is then instructed to improve their solution based on their review, generating a new solution that incorporates lessons learned from the matches and the experiences of other candidates. This process is repeated for a predefined number of iterations. The candidate with the highest Elo score across all groupis selected, and its solution is considered the optimal solution.

4 Experiments

4.1 Settings

Datasets and Metrics We evaluate our method on several widely-used and challenging benchmarks, including MATH (Hendrycks et al., 2021), GPQA-Diamond (Rein et al., 2023), and LiveBench (White et al., 2024). Following (Wang et al., 2024b; Qi et al., 2024), we adopt MATH-500, a subset of representative problems from MATH, to expedite the evaluation process. For LiveBench, we select the latest publicly available `0831` branch. We focus on two reasoning-oriented subsets, namely LiveBench-Math (`AMPS_Hard`, `Competition`, and `Olympiad`) and LiveBench-Reasoning (`Spatial`, `Web_of_lies_v2`, and `Zebra_Puzzles`). For `Olympiad` tasks, we adopt this metric: $1 - \text{dist}(\hat{y}, y)/\max(\text{len}(\hat{y}), \text{len}(y)) \in [0, 1]$ and scale it to $[0, 100]$ to measure the discrepancy between the predicted and ground truth answers, where $\text{dist}(\cdot)$ represents the edit distance between the prediction $\hat{y}$ and the ground truth y. For all other datasets, we report the Pass@1 accuracy. We describe these datasets in detail in Appendix C.

Baselines We compare WoT with the following baselines: **(1)** ***Standard CoT Prompting***, including Zero-shot CoT (Kojima et al., 2022), self-consistency (SC) (Wang et al., 2023c) and BoN (Stiennon et al., 2020; Nakano et al., 2021); **(2)** ***Reflection Prompting***, namely Self-Refine (Madaan et al., 2023); **(3)** ***Tree Search-based Prompting***, containing ToT (Yao et al., 2024), RAP (Hao et al., 2023), and rStar (Qi et al., 2024). For SC and BoN, we sample 8/64/128 times. We refer to Appendix D for more implementation details.

LLMs and Prompts We conduct experiments using open-source LLMs, specifically Qwen-2.5-7B-Instruct and Llama-3.1-8B-Instruct, alongside the proprietary LLM GPT-4o-mini. For baselines, we use the original prompts and adapt new ones as needed. For our algorithm, we design a variety of prompts to accommodate different datasets. Please refer to Appendix F for full prompts.

4.2 Main Results

We evaluate WoT on various challenging reasoning benchmarks to verify its effectiveness. To demonstrate its generality, we conduct experiments using Qwen-2.5-7B-Instruct, Llama-3.1-8B-Instruct, and GPT-4o-mini as backbones. In this setup, all modules of WoT adopt the same LLM. The main results are presented in Table 1. We observe that: **(1) WoT provides consistent improvements over other baselines.** Across all reasoning tasks, WoT (evaluated on Qwen-2.5) yields an average improvement of 15.65 compared to Zero-shot CoT, and shows significant and stable enhancements over other methods as well. In contrast, some methods like ToT and RAP experience a decrease in performance on certain tasks. This demonstrates our improvements are pronounced and consistent. **(2) WoT exhibits better generality across different LLMs and tasks.** Whether using open-sourced LLMs like Qwen-2.5 and Llama-3.1 or proprietary ones like GPT-4o-mini, and across various tasks ranging from scientific reasoning to spatial and logical reasoning, WoT consistently shows substan-

Model	Method	MATH-500	GPQA-Diamond	LiveBench-Math			LiveBench-Reasoning		
				{AMPS_Hard	Competitions	Olympiad}	{Spatial	Web_of_Lies_v2	Zebra_Puzzles}
Qwen-2.5-7B-Instruct	Zero-shot CoT	68.80	29.80	19.00	35.79	17.70	26.00	34.00	26.00
	BoN@8/64/128	72.40 ↑ 3.60	35.35 ↑ 5.55	23.00 ↑ 4.00	36.84 ↑ 1.05	14.30 ↓ 3.40	26.00 ↑ 0.00	34.00 ↑ 18.00	32.00 ↑ 6.00
		75.80 ↑ 7.00	38.89 ↑ 9.09	24.00 ↑ 5.00	37.89 ↑ 2.10	11.60 ↓ 6.10	30.00 ↑ 4.00	58.00 ↑ 24.00	36.00 ↑ 10.00
		76.20 ↑ 7.40	37.37 ↑ 7.57	26.00 ↑ 7.00	40.00 ↑ 4.21	10.90 ↓ 6.8	32.00 ↑ 6.00	60.00 ↑ 26.00	32.00 ↑ 6.00
	SC@8/64/128	72.20 ↑ 3.40	35.35 ↑ 5.55	22.00 ↑ 3.00	36.84 ↑ 1.05	17.00 ↓ 0.7	28.00 ↑ 2.00	48.00 ↑ 14.00	34.00 ↑ 8.00
		76.20 ↑ 7.40	37.37 ↑ 7.57	24.00 ↑ 5.00	38.95 ↑ 3.16	11.90 ↓ 5.80	28.00 ↑ 2.00	48.00 ↑ 18.00	34.00 ↑ 8.00
		75.80 ↑ 7.00	38.89 ↑ 9.09	25.00 ↑ 6.00	41.05 ↑ 5.26	11.40 ↓ 6.30	32.00 ↑ 6.00	62.00 ↑ 28.00	32.00 ↑ 6.00
	Self-Refine	72.20 ↑ 3.40	33.84 ↑ 4.04	25.00 ↑ 6.00	37.89 ↑ 2.10	18.20 ↑ 0.50	30.00 ↑ 4.00	44.00 ↑ 10.00	28.00 ↑ 2.00
	ToT	64.40 ↓ 4.40	32.32 ↑ 2.52	21.00 ↑ 2.00	34.74 ↓ 1.05	17.30 ↓ 0.40	28.00 ↑ 2.00	46.00 ↑ 12.00	24.00 ↓ 2.00
	RAP	62.60 ↓ 6.20	34.34 ↑ 4.54	22.00 ↑ 3.00	32.63 ↓ 3.16	17.90 ↑ 0.20	26.00 ↑ 0.00	44.00 ↑ 10.00	32.00 ↑ 6.00
	rStar	74.80 ↑ 6.00	40.40 ↑ 10.60	29.00 ↑ 10.00	42.11 ↑ 6.32	23.90 ↑ 6.20	32.00 ↑ 6.00	54.00 ↑ 20.00	36.00 ↑ 10.00
	WoT (ours)	**79.80 ↑ 11.00**	**43.43 ↑ 13.63**	**35.00 ↑ 16.00**	**47.37 ↑ 11.58**	**34.70 ↑ 17.00**	**38.00 ↑ 12.00**	**66.00 ↑ 32.00**	**38.00 ↑ 12.00**
Llama-3.1-8B-Instruct	Zero-shot CoT	45.00	29.29	12.00	23.16	10.80	10.00	28.00	22.00
	BoN@8/64/128	52.40 ↑ 7.40	32.83 ↑ 3.54	13.00 ↑ 1.00	25.26 ↑ 2.10	9.50 ↓ 1.30	16.00 ↑ 6.00	32.00 ↑ 4.00	30.00 ↑ 8.00
		55.60 ↑ 10.60	33.33 ↑ 4.04	16.00 ↑ 4.00	27.37 ↑ 4.21	6.70 ↓ 4.10	24.00 ↑ 14.00	52.00 ↑ 24.00	**34.00 ↑ 12.00**
		58.00 ↑ 13.00	33.84 ↑ 4.55	17.00 ↑ 5.00	28.42 ↑ 5.26	7.90 ↓ 2.90	30.00 ↑ 20.00	**54.00 ↑ 26.00**	**34.00 ↑ 12.00**
	SC@8/64/128	53.80 ↑ 8.80	33.33 ↑ 4.04	16.00 ↑ 4.00	24.21 ↑ 1.05	9.20 ↓ 1.60	14.00 ↑ 4.00	34.00 ↑ 6.00	28.00 ↑ 6.00
		56.00 ↑ 11.00	33.33 ↑ 4.04	16.00 ↑ 4.00	26.32 ↑ 3.16	6.60 ↓ 4.20	24.00 ↑ 14.00	52.00 ↑ 24.00	28.00 ↑ 6.00
		58.60 ↑ 13.60	35.86 ↑ 6.57	16.00 ↑ 4.00	28.42 ↑ 5.26	6.60 ↓ 4.20	26.00 ↑ 16.00	**54.00 ↑ 26.00**	31.00 ↑ 9.00
	Self-Refine	48.00 ↑ 3.00	31.31 ↑ 2.02	17.00 ↑ 5.00	25.26 ↑ 2.10	13.20 ↑ 2.40	16.00 ↑ 6.00	36.00 ↑ 8.00	22.00 ↑ 0.00
	ToT	47.40 ↑ 2.40	29.80 ↑ 0.51	20.00 ↑ 8.00	21.05 ↓ 2.11	12.50 ↑ 1.70	12.00 ↑ 2.00	32.00 ↑ 4.00	22.00 ↑ 0.00
	RAP	48.20 ↑ 3.20	29.29 ↑ 0.00	19.00 ↑ 7.00	23.16 ↑ 0.00	11.70 ↑ 0.90	12.00 ↑ 2.00	28.00 ↑ 0.00	24.00 ↑ 2.00
	rStar	55.40 ↑ 10.40	35.86 ↑ 6.57	25.00 ↑ 13.00	30.53 ↑ 7.37	15.00 ↑ 4.20	26.00 ↑ 16.00	48.00 ↑ 20.00	**34.00 ↑ 12.00**
	WoT (ours)	**62.60 ↑ 17.60**	**39.90 ↑ 10.61**	**29.00 ↑ 17.00**	**34.74 ↑ 11.58**	**21.90 ↑ 11.10**	**32.00 ↑ 22.00**	**54.00 ↑ 26.00**	**34.00 ↑ 12.00**
GPT-4o-mini	Zero-shot CoT	71.80	40.40	22.00	44.21	20.30	36.00	24.00	32.00
	BoN@8/64/128	74.80 ↑ 3.00	42.93 ↑ 2.53	23.00 ↑ 1.00	48.42 ↑ 4.21	19.70 ↓ 0.60	38.00 ↑ 2.00	44.00 ↑ 20.00	36.00 ↑ 4.00
		76.00 ↑ 4.20	43.43 ↑ 3.03	24.00 ↑ 2.00	49.47 ↑ 5.26	16.90 ↓ 3.40	34.00 ↓ 2.00	50.00 ↑ 26.00	42.00 ↑ 10.00
		76.80 ↑ 5.00	43.43 ↑ 3.03	23.00 ↑ 1.00	49.47 ↑ 5.26	16.90 ↓ 5.90	32.00 ↓ 4.00	**56.00 ↑ 32.00**	36.00 ↑ 4.00
	SC@8/64/128	74.60 ↑ 2.80	42.42 ↑ 2.02	22.00 ↑ 0.00	47.37 ↑ 3.16	20.10 ↓ 0.20	36.00 ↑ 0.00	36.00 ↑ 12.00	32.00 ↑ 0.00
		75.60 ↑ 3.80	43.43 ↑ 3.03	23.00 ↑ 1.00	48.42 ↑ 4.21	16.60 ↓ 3.70	32.00 ↓ 4.00	44.00 ↑ 20.00	40.00 ↑ 8.00
		76.40 ↑ 4.60	43.43 ↑ 3.03	23.00 ↑ 1.00	48.42 ↑ 4.21	15.00 ↓ 5.30	34.00 ↓ 2.00	48.00 ↑ 24.00	36.00 ↑ 4.00
	Self-Refine	71.80 ↑ 0.00	40.91 ↑ 0.51	23.00 ↑ 1.00	46.32 ↑ 2.11	21.00 ↑ 0.7	34.00 ↓ 2.00	32.00 ↑ 8.00	32.00 ↑ 0.00
	ToT	67.00 ↓ 4.80	36.87 ↓ 3.53	24.00 ↑ 2.00	43.16 ↓ 1.05	19.50 ↓ 0.80	30.00 ↓ 6.00	32.00 ↑ 8.00	34.00 ↑ 2.00
	RAP	66.20 ↓ 5.60	40.40 ↓ 5.05	19.00 ↓ 3.00	46.32 ↑ 2.11	21.20 ↑ 0.90	36.00 ↑ 0.00	36.00 ↑ 12.00	34.00 ↑ 2.00
	rStar	77.20 ↑ 5.40	43.43 ↑ 3.03	26.00 ↑ 4.00	49.47 ↑ 5.26	26.30 ↑ 6.00	38.00 ↑ 2.00	52.00 ↑ 28.00	40.00 ↑ 8.00
	WoT (ours)	**81.40 ↑ 9.60**	**46.97 ↑ 6.57**	**31.00 ↑ 9.00**	**53.68 ↑ 9.47**	**39.50 ↑ 19.20**	**44.00 ↑ 8.00**	**56.00 ↑ 32.00**	**48.00 ↑ 16.00**

Table 1: The performance on different reasoning tasks. Best results are **bolded** and suboptimal results are underlined. WoT improves accuracy consistently across various tasks and LLM backbones.

tial improvements. Specifically, from the view of LLMs, WoT evaluated on Qwen-2.5-7B-Instruct, Llama-3.1-8B-Instruct, and GPT-4o-mini achieve average improvements of 15.65, 15.97, and 13.73, respectively. On the task level, WoT applies to various task categories, attributing to the competition-based evaluation. In contrast, BoN and tree search-based methods depend on reward models, limiting their universality. **(3) SC and BoN remain strong baselines.** With increased sampling, SC and BoN achieve impressive results, often surpassing Self-Refine, ToT, and RAP on various tasks, and even approaching rStar on certain tasks. However, they underperform on the `Olympiad` task due to reduced consistency in sampled results as the answer space expands. WoT does not suffer from this issue because it incorporates granularity specification and contrastive demonstrations to enhance sampling diversity, and improves through competition, thereby stimulating reasoning capabilities while maintaining consistency, leading to exceptional performance.

4.3 Ablation Study

Components Ablation To investigate the efficacy of each component in WoT, we conduct ablation studies using Llama-3.1-8B-Instruct as the backbone on the MATH-500 and GPQA-Diamond datasets. The results are shown in Table 2. **i) Reasoning specifications and contrastive demonstrations boosts correctness.** Individually adding contrastive demonstrations improves performance across all tasks (c vs. d), as LLMs learn specific error patterns from both positive and negative examples, avoiding similar mistakes. With different reasoning granularity specifications, LLMs address problems from various perspectives. This enhances diversity and maximizes the activation of reason-

Index	Components *Granularity Spec.*	 *Contrastive Demos.*	 *Multi-round Matchup*	 *Matchup Review*	 *Competition*	**MATH-500**	**GPQA-Diamond**
a						52.80↓ 9.80	32.32↓ 7.58
b	✓		✓	✓	✓	60.20↓ 2.40	37.37↓ 2.53
c		✓	✓	✓	✓	57.60↓ 5.00	35.35↓ 4.55
d			✓	✓	✓	56.40↓ 6.20	33.84↓ 6.06
e	✓	✓				54.20↓ 8.40	33.33↓ 6.57
f	✓	✓		✓	✓	60.80↓ 1.80	37.88↓ 2.02
g	✓	✓	✓		✓	58.80↓ 3.80	35.35↓ 4.55
h	✓	✓	✓	✓	✓	**62.60**	**39.90**

Table 2: Ablation study of different components in WoT. Evaluated with Llama-3.1-8B-Instruct.

Model	Judge LLM	MATH-500	GPQA-Diamond
Qwen-2.5-7B-Instruct	Qwen-2.5-7B-Instruct	79.80	43.43
	Llama-3.1-8B-Instruct	79.20	42.42
	GPT-4o-mini	79.60	43.43
	GPT-4o	**81.00**	45.96
	Claude-3.5-Sonnet	80.60	**47.98**
Llama-3.1-8B-Instruct	Qwen-2.5-7B-Instruct	61.80	37.88
	Llama-3.1-8B-Instruct	62.60	39.90
	GPT-4o-mini	62.20	39.39
	GPT-4o	**64.80**	42.93
	Claude-3.5-Sonnet	64.00	**44.44**

Table 3: Ablation study on judge LLM selection.

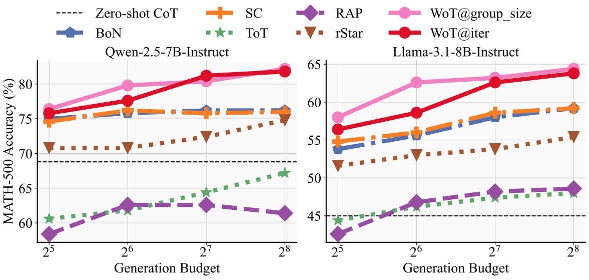


Figure 4: Performance of different methods by TTS. WoT can be scaled more efficiently.

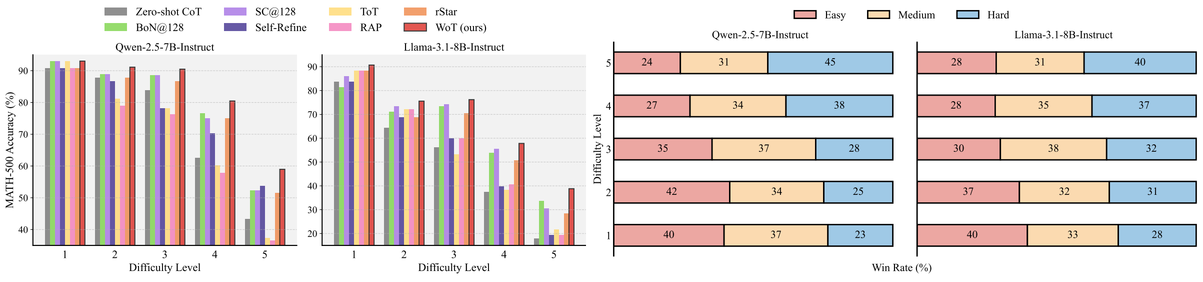
ing abilities within different reasoning boundaries, thus improving reasoning accuracy (b vs. d). The combination of both components further improves performance (h vs. d and e vs. a). **ii) Multi-round matchup reduce positional sensitivity in comparative evaluations.** Replacing multi-round matchup with a fixed-order single-round matchup results in decreases of 1.80 and 2.02 on MATH-500 and GPQA-Diamond respectively (h vs. f). This indicates that swapping solution orders in prompts and then conducting multi-round assessment can mitigate positional bias in comparative evaluations to some extent. **iii) Matchup review is more effective than self-reflection.** Replacing the matchup review with self-reflection led to larger performance drops of 3.80 and 4.55 on the two tasks (h vs. g). This is intuitive since it allows learning from other solutions to identify where the differences lie, which factors contributed to potential inconsistent results, and what insights can be gained for improvement. **iv) Competition Enhances the Reasoning Ability of LLMs.** We remove the competition mechanism in WoT, and instead allow different solutions to undergo multiple rounds of reflection, after which a single LLM acts as a judge to score the solutions and select the one with the highest score. We observe that after eliminating this core component, the accuracy on MATH-500 and GPQA-Diamond decreases by 8.40 and 6.57 (h vs. e). This demonstrates that competition boosts the reasoning ability of LLMs, and Elo-based evaluation is more effective and precise.

Judge LLM Ablation We study the impact of judge LLM selection. We test a range of judge LLMs, both weaker and stronger. The results are presented in Table 3. We observe that the choice of judge LLM generally does not affect the effectiveness of the competition mechanism and the Elo-based selection system in WoT. Notably, using more powerful models such as GPT-4o and Claude-3.5-Sonnet only brings slight improvements compared to smaller models (e.g., 81.00, 80.60 vs. 79.80). This demonstrates that WoT is robust and adaptable across models of different scales.

5 Analysis

5.1 Test-Time Scaling

We compare the performance of different methods by TTS, as shown in Figure 4. We define various generation budgets and adjust certain parameters of each approach while staying within the budgets. For WoT@group_size/iter, we modify the number of solutions per group and the number of iterations. The scaling strategies for other methods are detailed in Appendix D. Across all budgets, WoT consistently outperforms others. As the budget increases, WoT 's performance continues to improve, while methods like BoN, SC, and ToT exhibit diminishing returns, and RAP even declines. Though rStar's performance also improves with increased



(a) (b)

Figure 5: Statistical results on the MATH-500 dataset evaluated with Qwen-2.5-7B-Instruct and Llama-3.1-8B-Instruct. (a) Performance of WoT and other baselines binned by difficulty level. (b) Win rates (%) of different groups in WoT across varying difficulty levels.

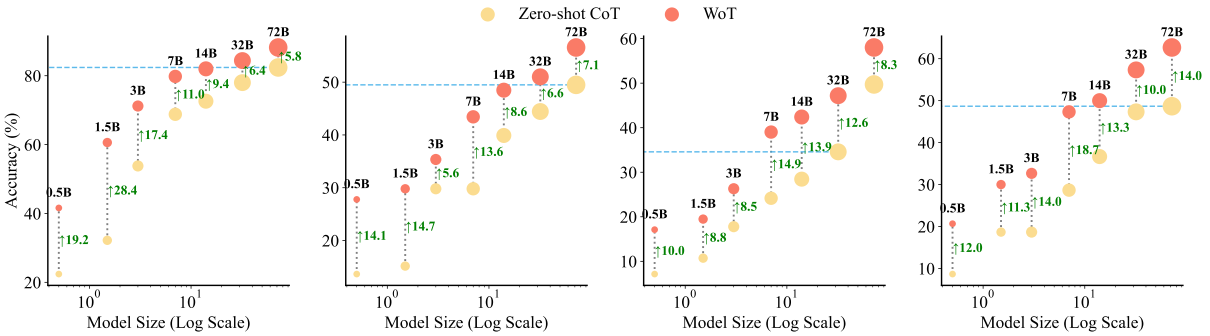


Figure 6: Results of WoT and Zero-shot CoT on all benchmarks evaluated with Qwen-2.5-Instruct models.

compute, its gains are less pronounced than WoT's. This suggests that WoT more efficiently leverages TTS. Additionally, scaling the number of solutions per group is a better strategy at the same budget.

5.2 Difficulty-Aware Design and Effectiveness

In the main experiments, we establish three groups, each characterized by a distinct reasoning granularity. We assume this division, ranging from coarse to fine, can be designed to address tasks of varying complexity, thus enhancing overall performance. To verify this, we categorize the problems in MATH-500 based on difficulty and compare the accuracy of WoT against other baselines under various difficulty levels. The results presented in Figure 5a indicate that WoT consistently outperforms the baselines, with the performance gains becoming more pronounced as the difficulty increases. This validates the rationale behind the proposed reasoning granularity division.

Furthermore, we examine the win rates of the different groups under varying difficulty levels, as illustrated in Figure 5b. For simpler problems, the group designed for easy tasks exhibited a higher win rate, while for more complex problems, the group optimized for hard tasks demonstrates superior performance. This highlights the adaptability of the reasoning granularity framework in addressing problems with various complexity, leading to overall performance improvements. Combined with the results in Table 1, it becomes evident that WoT effectively selects optimal solutions through competitive evaluation, further demonstrating the efficacy of the proposed design.

5.3 Model Size Scaling

We apply WoT across all datasets using Qwen-2.5-Instruct models of varying sizes (ranging from 0.5B to 72B) and compare the results against those obtained with Zero-shot CoT. As shown in Figure 6, WoT exhibits a clear upward trend in performance as model size increases, with consistent improvements observed at each model scale. Notably, the application of WoT to smaller models yields results that sometimes surpass those of larger models. For example, on LiveBench-Reasoning, despite a 10× more parameter discrepancy, the 7B model with WoT outperforms the 72B model. These findings demonstrate that WoT scales effectively with model size, highlighting its ability to enhance performance even for smaller models.

6 Conclusion

We propose WoT, a novel method that enhances LLM reasoning during inference without finetuning. We group LLMs as candidates and have them compete in pairwise matchups. Futhermore, we introduce Elo ratings to transform unstable partial orders into reliable and quantifiable rankings. Extensive experiments across various LLM backbones and reasoning tasks demonstrate the superiority of WoT. Additional analysis highlights its scalability, rationality, and shows that competition effectively stimulates stronger reasoning in LLMs.

Limitations

Although we employ multi-round matchup, it is still possible for positional bias to occur during evaluation. For smaller-scale LLMs, their limited instruction following capabilities may hinder accurate comparisons and effective reflection. The issue of forgetting in LLMs also poses a challenge to the scalability of WoT. Besides, determining a more reasonable and efficient way to define reasoning granularity based on the specific task in order to expand the LLMs' reasoning capabilities remains an open question. Future research is needed to surmount these challenges more effectively.

Ethics Statement

All the datasets and baselines used in this work are publicly available, and our proposed method does not raise any potential ethical concerns. Our work adheres to the ACL Ethics Policy.

Acknowledgements

This work is supported by the National Key Research and Development Program of China (Grant No. 2024YFE0210900), the National Natural Science Foundation of China (Grant Nos. 62422605, 92370132) and the Xiaomi Young Talents Program of Xiaomi Foundation.

References

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

Edward Beeching, Lewis Tunstall, and Sasha Rush. 2024. Scaling test-time compute with open models.

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

Meriem Boubdir, Edward Kim, Beyza Ermis, Sara Hooker, and Marzieh Fadaee. 2023. Elo uncovered: Robustness and best practices in language model evaluation. In *Proceedings of the Third Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 339–352.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. 2024a. Alphamath almost zero: process supervision without process. *arXiv preprint arXiv:2405.03553*.

Qiguang Chen, Libo Qin, Jiaqi Wang, Jinxuan Zhou, and Wanxiang Che. 2024b. Unlocking the capabilities of thought: A reasoning boundary framework to quantify and optimize chain-of-thought. *arXiv preprint arXiv:2410.05695*.

Yibin Chen, Yifu Yuan, Zeyu Zhang, Yan Zheng, Jinyi Liu, Fei Ni, and HAO Jianye. 2024c. Sheetagent: A generalist agent for spreadsheet reasoning and manipulation via large language models. In *ICML 2024 Workshop on LLMs and Cognition*.

Pengyu Cheng, Tianhao Hu, Han Xu, Zhisong Zhang, Yong Dai, Lei Han, Xiaolong Li, et al. 2024. Self-playing adversarial language game enhances llm reasoning. *Advances in Neural Information Processing Systems*, 37:126515–126543.

Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Banghua Zhu, Hao Zhang, Michael Jordan, Joseph E Gonzalez, et al. 2024. Chatbot arena: An open platform for evaluating llms by human preference. In *Forty-first International Conference on Machine Learning*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Ruomeng Ding, Chaoyun Zhang, Lu Wang, Yong Xu, Minghua Ma, Wei Zhang, Si Qin, Saravan Rajmohan, Qingwei Lin, and Dongmei Zhang. 2024. Everything of thoughts: Defying the law of penrose triangle for thought generation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 1638–1662. Association for Computational Linguistics.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Arpad E Elo and Sam Sloan. 1978. The rating of chessplayers: Past and present. *Arco Pub.*

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

Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8154–8173.

Alex Havrilla, Yuqing Du, Sharath Chandra Raparthy, Christoforos Nalmpantis, Jane Dwivedi-Yu, Maksym Zhuravinskyi, Eric Hambro, Sainbayar Sukhbaatar, and Roberta Raileanu. 2024. Teaching large language models to reason with reinforcement learning. *arXiv preprint arXiv:2403.04642*.

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 *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2023. Large language models can self-improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

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

Dawei Li, Zhen Tan, and Huan Liu. 2025. Exploring large language models for feature selection: A data-centric perspective. *ACM SIGKDD Explorations Newsletter*, 26(2):44–53.

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Omer Levy, Luke Zettlemoyer, Jason E Weston, and Mike Lewis. 2024a. Self-alignment with instruction backtranslation. In *The Twelfth International Conference on Learning Representations*.

Zongjie Li, Chaozheng Wang, Pingchuan Ma, Daoyuan Wu, Shuai Wang, Cuiyun Gao, and Yang Liu. 2024b. Split and merge: Aligning position biases in llm-based evaluators. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11084–11108.

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

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems*, 36.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2024. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. *arXiv preprint arXiv:2410.05229*.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

Zhenting Qi, Mingyuan Ma, Jiahang Xu, Li Lyna Zhang, Fan Yang, and Mao Yang. 2024. Mutual reasoning makes smaller llms stronger problem-solvers. *arXiv preprint arXiv:2408.06195*.

Shanghaoran Quan, Jiaxi Yang, Bowen Yu, Bo Zheng, Dayiheng Liu, An Yang, Xuancheng Ren, Bofei Gao, Yibo Miao, Yunlong Feng, et al. 2025. Codeelo: Benchmarking competition-level code generation of llms with human-comparable elo ratings. *arXiv preprint arXiv:2501.01257*.

Vatsal Raina, Adian Liusie, and Mark Gales. 2025. Finetuning LLMs for comparative assessment tasks. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 3345–3352, Abu Dhabi, UAE. Association for Computational Linguistics.

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

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

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2024. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. *Advances in Neural Information Processing Systems*, 36.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

Seonil Son, Ju-Min Oh, Heegon Jin, Cheolhun Jang, Jeongbeom Jeong, and Kuntae Kim. 2024. Varco arena: A tournament approach to reference-free benchmarking large language models. *arXiv preprint arXiv:2411.01281*.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

Juanhe (TJ) Tan. 2023. Causal abstraction for chain-of-thought reasoning in arithmetic word problems. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 155–168, Singapore. Association for Computational Linguistics.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Aman Singh Thakur, Kartik Choudhary, Venkat Srinik Ramayapally, Sankaran Vaidyanathan, and Dieuwke Hupkes. 2024. Judging the judges: Evaluating alignment and vulnerabilities in llms-as-judges. *arXiv preprint arXiv:2406.12624*.

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

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Luong Trung, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. 2024. ReFT: Reasoning with reinforced fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7601–7614, Bangkok, Thailand. Association for Computational Linguistics.

Karthik Valmeekam, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. 2022. Large language models still can't plan (a benchmark for llms on planning and reasoning about change). In *NeurIPS 2022 Foundation Models for Decision Making Workshop*.

Ziyu Wan, Xidong Feng, Muning Wen, Stephen Marcus Mcaleer, Ying Wen, Weinan Zhang, and Jun Wang. 2024. AlphaZero-like tree-search can guide large language model decoding and training. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 49890–49920. PMLR.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023a. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2609–2634.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. 2024a. Large language models are not fair evaluators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9440–9450, Bangkok, Thailand. Association for Computational Linguistics.

Peiyi Wang, Lei Li, Liang Chen, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. 2023b. Large language models are not fair evaluators. In *Annual Meeting of the Association for Computational Linguistics*.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024b. 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)*, pages 9426–9439.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023c. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

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

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. 2023. Large language models are better reasoners with self-verification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2550–2575.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Ben Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Siddartha Naidu, et al. 2024. Livebench: A challenging, contamination-free llm benchmark. *arXiv preprint arXiv:2406.19314*.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. 2023. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*.

Changnan Xiao and Bing Liu. 2024. A theory for length generalization in learning to reason. *arXiv preprint arXiv:2404.00560*.

Tinghao Xie, Xiangyu Qi, Yi Zeng, Yangsibo Huang, Udari Madhushani Sehwag, Kaixuan Huang, Luxi He, Boyi Wei, Dacheng Li, Ying Sheng, et al. 2024. Sorry-bench: Systematically evaluating large language model safety refusal behaviors. *arXiv preprint arXiv:2406.14598*.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, YIFEI LI, Jie Fu, Junxian He, and Bryan Hooi. 2024. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms. In *The Twelfth International Conference on Learning Representations*.

Yifan Xu, Xiao Liu, Xinghan Liu, Zhenyu Hou, Yueyan Li, Xiaohan Zhang, Zihan Wang, Aohan Zeng, Zhengxiao Du, Wenyi Zhao, et al. 2024. Chatglm-math: Improving math problem-solving in large language models with a self-critique pipeline. *arXiv preprint arXiv:2404.02893*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*.

Longhui Yu, Weisen Jiang, Han Shi, YU Jincheng, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations*.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. 2023. Glm-130b: An open bilingual pre-trained model. In *The Eleventh International Conference on Learning Representations*.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. 2024a. Rest-mcts*: Llm self-training via process reward guided tree search. *arXiv preprint arXiv:2406.03816*.

Di Zhang, Xiaoshui Huang, Dongzhan Zhou, Yuqiang Li, and Wanli Ouyang. 2024b. Accessing gpt-4 level mathematical olympiad solutions via monte carlo tree self-refine with llama-3 8b. *arXiv preprint arXiv:2406.07394*.

Wenqi Zhang, Yongliang Shen, Linjuan Wu, Qiuying Peng, Jun Wang, Yueting Zhuang, and Weiming Lu. 2024c. Self-contrast: Better reflection through inconsistent solving perspectives. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3602–3622, Bangkok, Thailand. Association for Computational Linguistics.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

Yan Zheng, Xiaofei Xie, Ting Su, Lei Ma, Jianye Hao, Zhaopeng Meng, Yang Liu, Ruimin Shen, Yingfeng Chen, and Changjie Fan. 2019. Wuji: Automatic online combat game testing using evolutionary deep reinforcement learning. In *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 772–784. IEEE.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. 2024. Language agent tree search unifies reasoning, acting, and planning in language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 62138–62160. PMLR.

A Reasoning Boundaries of LLMs

Mathematically, RB for a model m and task t is defined as $\mathcal{B}_{\mathrm{Acc}=K_1}(t \mid m) = \sup_d \{d \mid \mathrm{Acc}(t \mid d, m) = K_1\}$, where d is the task difficulty.

In complex reasoning tasks, LLMs often require the integration of multiple capabilities. The *Combination Law of RB* defines how the cooperation of different abilities enhances LLM performance through CoT. It is given by the weighted harmonic average of individual RBs:

$$\mathcal{B}_{\mathrm{Acc}=K_1}(t_1, t_2, \ldots, t_n \mid m) \approx \frac{1}{(n-1)\sum_{i=1}^{n} \frac{N_i}{\mathcal{B}_{\mathrm{Acc}=K_1}(t_i|m)-b_i}},$$

where N_i and b_i are task-specific scaling factors. For example, in mathematical reasoning, the CoT may involve step planning and step calculation (Tan, 2023; Xiao and Liu, 2024), with individual RBs $\mathcal{B}(p)$ and $\mathcal{B}(c)$, respectively. The combined RB is given by $\mathcal{B}^{\mathrm{CoT}}(c,p) = 1/\left(\frac{N_1}{(\mathcal{B}(c)-b_1)} + \frac{N_2}{(\mathcal{B}(p)-b_2)}\right)$.

RB sets a limit on the LLM's performance. To optimize CoT, we aim to adjust the reasoning path such that the difficulty aligns with the optimal RB ($d^* = \mathcal{B}_{\mathrm{Acc}=K_1}$), rather than the original RB ($d = \mathcal{B}_{\mathrm{Acc}=K_2}$) where $K_1 > K_2$. In mathematical reasoning, this requires balancing planning and calculation to maximize computational efficiency at each step. Our approach involves designing prompts with varying reasoning granularities, explicitly defining planning and calculation boundaries to achieve optimal performance.

B Elo Rating

The Elo rating system (Elo and Sloan, 1978) is widely used to evaluate LLMs' performance (Zheng et al., 2023; Chiang et al., 2024; Son et al., 2024; Quan et al., 2025). For a zero-sum matchup between two models A and B, with pre-match ratings R_A and R_B, the expected scores E_A and E_B are given by:

$$E_A = \frac{1}{1+10^{(R_B-R_A)/400}}, \quad E_B = \frac{1}{1+10^{(R_A-R_B)/400}}. \tag{1}$$

After each match, the Elo scores are updated based on the actual win-loss result:

$$R'_A = R_A + K(S_A - E_A), \quad R'_B = R_B + K(S_B - E_B), \tag{2}$$

where S_A and S_B are the match outcomes (1 for a win, 0 for a loss), and K controls the sensitivity of the rating change.

Elo rating-based performance ranking system rely on *transitivity* and *reliability* (Zheng et al., 2023). Transitivity means if A beats B, and B beats C, then A should beat C. However, this condition may not hold in LLM evaluations (Boubdir et al., 2023). Moreover, the outcomes of matches can be sensitive to the order of prompts during evaluation (Wang et al., 2024a), which affects the reliability of the Elo rating.

C Dataset Details

We declare that our use of all involved datasets is consistent with their intended purposes and complies with their respective licenses. The datasets we used are as follows:

MATH[1] (Hendrycks et al., 2021) is a dataset containing 12,500 challenging competition-style math problems, proposed by Hendrycks et al., 2021. Each problem in MATH comes with a step-by-step solution and a corresponding difficulty level. In this work, following the approach of (Wang et al., 2024b; Qi et al., 2024), we use MATH-500[2], a subset of MATH containing 500 representative problems, to accelerate our evaluation. The dataset is licensed under the MIT License.

GPQA[3] (Rein et al., 2023) is a dataset comprising of 448 multiple-choice questions annotated by experts from the fields of biology, physics, and chemistry, proposed by Rein et al., 2023. We evaluate using the highest-quality subset, GPQA-Diamond, which includes 198 questions that were correctly answered by experts but mostly misanswered by non-experts. The dataset is licensed under the MIT License.

LiveBench[4] (White et al., 2024) is a collection of challenging tasks spanning areas such as math, coding, reasoning, language, instruction following, and data analysis, created by White et al., 2024. LiveBench claims to regularly update its problems. We used the latest publicly available branch, `LiveBench-2024-08-31`. The dataset is licensed under the Apache License 2.0. From this dataset, we select two

[1] https://github.com/hendrycks/math

[2] https://huggingface.co/datasets/HuggingFaceH4/MATH-500

[3] https://huggingface.co/datasets/Idavidrein/gpqa

[4] https://huggingface.co/livebench

reasoning-intensive categories: LiveBench-Math and LiveBench-Reasoning. LiveBench-Math includes the following tasks:

- `AMPS_Hard`: This task generates harder problems by drawing random primitives, using a larger and more challenging distribution than AMPS across 10 of the hardest tasks within AMPS (Hendrycks et al., 2021), with 100 questions in total.

- `Competitions`: This task includes questions from AMC12 2023, SMC 2023, and AIME 2024, with modified prose and answer order, consisting of 95 questions.

- `Olympiad`: This task includes questions based on USAMO 2024 and IMO 2024, where the goal is to rearrange masked-out equations from the solution into the correct order, with 36 questions.

LiveBench-Reasoning includes the following tasks, each with 50 questions:

- `Spatial`: A spatial reasoning task designed to test the model's ability to infer intersections and directions of common 2D and 3D shapes.

- `Web_of_Lies_v2`: An improved version of similar tasks found in BigBench and Big-Bench-Hard. This task represents a random Boolean function as a natural language word problem and asks for its truth value. This version adds additional deductive components and several types of red herrings, significantly increasing the difficulty.

- `Zebra_Puzzles`: Tests the ability of the model to follow a set of statements that establish constraints and then logically deduce the requested information.

D Implementation Details

D.1 Details of Baselines

- **Zero-shot CoT:** We adopt the implementation from Kojima et al., 2022, appending "Let's think step by step" at the end of the prompt. We perform inference using a zero-shot approach and employ greedy decoding (temperature $T = 0$).

- **SC:** Except for the temperature $T = 0.8$, we use the same settings as Zero-shot CoT, sampling N times and selecting the most frequent answer through majority voting as the final answer.

- **BoN:** We refer to the publicly available implementation from Beeching et al., 2024[5] and adapt it for the new tasks. We instruct the LLM to run step-by-step reasoning and introduce a PRM to score each step. Specifically, we use the weighted BoN to select the best solution, which prioritizes high-quality answers by giving higher scores to more frequently occurring answers. Mathematically, the weighting across answers a_i is performed as follows:

$$a_{\text{weighted}} = \arg\max_{a} \sum_{i=1}^{N} \mathbb{I}\left(a_i = a\right) \cdot \mathrm{RM}\left(p, s_i\right),$$

where $\mathrm{RM}(p, s_i)$ is the score of the i-th solution s_i for the question p. Here, a solution s_i refers to the concatenation of all steps, and a_i represents the final answer, typically extracted from a special form. In this work, we use the score of the last step as the score for a solution. This score encapsulates the cumulative information from all prior steps, effectively treating the PRM as an ORM that can score partial solutions. Regarding the choice of PRM, for mathematical reasoning tasks (MATH-500 and LiveBench-Math), we use `math-shepherd-mistral-7b-prm`[6] to provide process rewards. For other reasoning tasks (GPQA-Diamond and LiveBench-Reasoning), we apply `Llama3.1-8B-PRM-Deepseek-Data`[7]. We also set the temperature $T = 0.8$.

- **Self-Refine:** We use the same implementation as in the original paper[8]. We set the temperature $T = 0.8$ and the maximum iteration $t = 5$.

- **ToT:** We refer to the publicly available implementation[9] and use the BFS algorithm as the search strategy. We define the branching width as $b = 5$ and the maximum depth as $d = 10$. For different tasks, we construct corresponding evaluative criteria and instruct another LLM to score each tree node. We select the trajectory with the highest score as the final answer. The temperature is also defined as $T = 0.8$.

[5] `https://github.com/huggingface/search-and-learn`

[6] `https://huggingface.co/peiyi9979/math-shepherd-mistral-7b-prm`

[7] `https://huggingface.co/RLHFlow/Llama3.1-8B-PRM-Deepseek-Data`

[8] `https://github.com/madaan/self-refine`

[9] `https://github.com/princeton-nlp/tree-of-thought-llm`

- **RAP:** We refer to the publicly available implementation[10]. We use MCTS for tree search and perform 16 rollouts on all tasks, with each node expanding 4 new child nodes. The temperature is set to $T = 0.8$. We calculate the node reward using majority voting, with 8 sampling times.

- **rStar:** We refer to the official implementation[11]. Due to the high cost of this approach, we run 16 rollouts during the trajectory self-generation stage. All tasks have a depth of $d = 5$. Actions A_1 and A_3 have a maximum of 5 nodes per depth, while other actions have a default node count of 1. Due to limited computational resources, we discard the trajectory discrimination stage.

D.2 Details of WoT

In the first stage, we use a small dataset to generate contrastive CoT demonstrations. The data used for all tasks is not included in the evaluation set. In the main experiment, we create 3 common errors for each task and define 3 levels of reasoning granularity. Specifically, the tasks are divided into 3 groups, with each group containing 3 candidates. In the second stage, we set the number of competition iterations to 3. Each participant starts with an initial Elo rating of 1,500, and the sensitivity parameter K for rating changes is set to 32. It is important to note that the LLM backbone is the same across all modules. Notably, in the iterative competition, WoT employs a parallelized implementation to make the experimental procedure more efficient.

D.3 LLM Backbones

In the main experiments, we use three different LLMs, namely Qwen-2.5-7B-Instruct[12] (Yang et al., 2024), Llama-3.1-8B-Instruct[13] (Dubey et al., 2024), and GPT-4o-mini[14]. For Qwen-2.5 and Llama-3.1, we deploy them using vLLM (Kwon et al., 2023) with version 0.6.3[15]. For GPT-4o-mini, we utilize the official API platform[16].

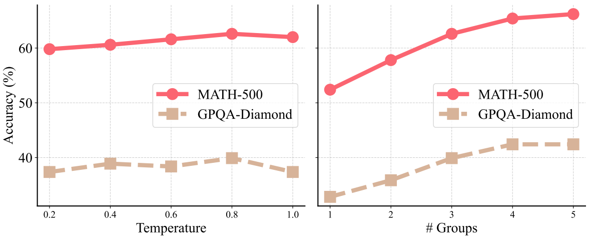


Figure 7: Parameter sensitivity of WoT on temperature (Left) and number of groups (Right).

D.4 Computing Power

All the results in our experiments are obtained by running the code on a server equipped with an Intel(R) Xeon(R) Gold 6338 CPU @ 2.00GHz and 2*NVIDIA A800.

E Additional Analysis

E.1 Parameter Sensitivity

We evaluate the impact of different parameters on the performance of WoT using Llama-3.1-8B-Instruct on MATH-500 and GPQA-Diamond. First, we conduct experiments under various temperature settings. As shown in Figure 7 (Left), the accuracy remains generally stable with minor fluctuations, demonstrating the robustness of WoT. Furthermore, we adjust the number of groups in WoT, which also affects the common errors and corresponding contrastive demonstrations. The results presented in Figure 7 (Right) show that accuracy improves as the number of groups increases. This highlights group scaling as an effective strategy for expanding test-time compute and further activating the reasoning potential of LLMs.

F Prompts

In this section, we present all the prompt templates used in WoT. We note that we have made slight adjustments to the prompt templates to adapt to different tasks. For brevity, we display the prompts used when evaluating MATH-500.

F.1 Prompt for Stage 1

Obtain Task-specific Common Errors

```
You are an expert in mathematical
↪ reasoning. Your task is to analyze
↪ several complex mathematical
↪ reasoning problems and identify {n}
↪ most common mistakes that people might
↪ make when attempting to solve it. For
↪ each mistake, provide the following:
```

[10] https://github.com/maitrix-org/llm-reasoners

[11] https://github.com/zhentingqi/rStar

[12] https://huggingface.co/Qwen/Qwen2.5-7B-Instruct

[13] https://huggingface.co/meta-llama/Llama-3.1-8B

[14] https://platform.openai.com/docs/models/gpt-4o-mini

[15] https://github.com/vllm-project/vllm/releases/tag/v0.6.3

[16] https://openai.com/api

```
1. A clear and concise description of the
↪  mistake.
2. An explanation of why this mistake is
↪  likely to occur in this context.

Your analysis should focus on reasoning
↪  mistakes such as incorrect
↪  assumptions, flawed logic, or
↪  misinterpretation of information,
↪  rather than trivial mistakes like
↪  typos or calculation mistakes. Be as
↪  specific as possible, and avoid overly
↪  generic responses.

Provide the output in valid JSON format
↪  with the following structure:
[
    {
        "category": "A clear and concise
        ↪  description of the mistake",
        "explanation": "An explanation of
        ↪  why this mistake is likely to
        ↪  occur in this context",
    },
    ...
]

{qa_pairs}
```

Samples of QA Pairs

```
Q: Let \\[f(x) =
↪  \\left\\{\n\\begin{array}{cl} ax+3,
↪  &\\text{ if }x>2, \\\\\nx-5 &\\text{
↪  if } -2 \\le x \\le 2, \\\\\n2x-b
↪  &\\text{ if } x
↪  <-2.\n\\end{array}\n\\right.\\]Find
↪  $a+b$ if the piecewise function is
↪  continuous (which means that its graph
↪  can be drawn without lifting your
↪  pencil from the paper).
A: For the piecewise function to be
↪  continuous, the cases must \"meet\" at
↪  $2$ and $-2$. For example, $ax+3$ and
↪  $x-5$ must be equal when $x=2$. This
↪  implies $a(2)+3=2-5$, which we solve
↪  to get $2a=-6 \\Rightarrow a=-3$.
↪  Similarly, $x-5$ and $2x-b$ must be
↪  equal when $x=-2$. Substituting, we
↪  get $-2-5=2(-2)-b$, which implies
↪  $b=3$. So $a+b=-3+3=\\boxed{0}$.

Q: What is the value of $9^3 + 3(9^2) +
↪  3(9) + 1$?
A: The given expression is the expansion
↪  of $(9+1)^3$.  In general, the cube of
↪  $(x+y)^3$ is
↪  \\[(x+y)^3=1x^3+3x^2y+3xy^2+1y^3.\\]
↪  The first and last terms in the given
↪  expression are cubes and the middle
↪  two terms both have coefficient 3,
↪  giving us a clue that this is a cube
↪  of a binomial and can be written in
↪  the form \\[(x+y)^3\\]In this case,
↪  $x=9$ and $y=1$, so our answer
↪  is\\[(9+1)^3\\] = 10^3 =
↪  \\boxed{1000}\\]

Q: Find the remainder when the sum
↪  \\[75+76+77+78+79+80+81+82\\]is
↪  divided by 16.
A: We notice that 16 divides $78+82$ as
↪  well as $79+81$ and also 80.
↪  Therefore the sum is congruent to
↪  \\[75+76+77\\pmod{16}.\\]Since these
↪  numbers are congruent to $-5$, $-4$,
↪  and
↪  $-3$ modulo 16, this can be computed as
↪  \\[-5-4-3\\equiv-12\\pmod{16}.\\]Finally,
↪  since $-12\\equiv4\\pmod{16}$ the
↪  remainder we seek is $\\boxed{4}$.

Q: Let $S$ be a region in the plane with
↪  area 10.  When we apply the
↪  matrix\n\\[\\begin{pmatrix} 2 & 1 \\\\
↪  7 & -3 \\end{pmatrix}\\]to $S,$ we
↪  obtain the region $S'.$  Find the area
↪  of $S'.$
A: Note that\n\\[\\begin{vmatrix} 2 & 1
↪  \\\\ 7 & -3 \\end{vmatrix} = (2)(-3) -
↪  (1)(7) = -13,\\]so the matrix scales
↪  the area of any region by a factor of
↪  $|-13| = 13.$  In particular, the area
↪  of $S'$ is $13 \\cdot 10 =
↪  \\boxed{130}.$

[omit for brevity]
```

Rephrase Question

```
Given the original problem, create a new
↪  problem that is within the same domain
↪  as the original. The new problem
↪  should:
1. Maintain a logical connection to the
↪  original problem and align with its
↪  type and level of complexity.
2. Introduce variations in phrasing,
↪  constraints, or focus to ensure it is
↪  not identical to the original.
3. Avoid creating problems that are too
↪  generic or unrelated.


Provide the output in valid JSON format
↪  using the following structure:
{
    "new_problem": "Your newly created
    ↪  problem."
}

Original Problem:
{problem}

Please output in valid JSON format without
↪  any additional comments.
```

Generate Contrastive Demonstrations–Conceptual Misunderstanding

```
You are an expert in mathematical
↪  reasoning. Given a complex
↪  mathematical problem, your task is to
↪  provide a correct solution and an
↪  incorrect solution that specifically
↪  involves a **conceptual
↪  misunderstanding**.

!!!Instructions:
1. Conceptual misunderstandings occur when
↪  there is an incorrect interpretation
↪  or application of a key concept,
↪  formula, or rule in the problem. These
↪  misunderstandings can arise from
↪  confusing definitions, misapplying
↪  principles, or overlooking key
↪  distinctions in the problem.
2. For both the correct and incorrect
↪  solutions, give step by step reasoning
↪  before you answer, and when you're
↪  ready to answer, please use the format
↪  "The final answer is \\boxed{answer}."
3. For the incorrect solution, Naturally
↪  connect to the given mistake as if it
↪  were a valid solution process. Avoid
↪  explicitly pointing out or analyzing
↪  the mistake in the process. Present
↪  the reasoning as if it were a genuine
↪  attempt to solve the problem.
4. Finally, include an explanation that
↪  identifies and points out where the
↪  conceptual misunderstandings
↪  occurred.

Example of a Conceptual Misunderstanding:
If asked to solve a geometry problem
↪  involving the area of a circle, a
↪  common misunderstanding could involve
↪  confusing the formula for
↪  circumference with the area formula.

Output Format:
Provide the output in valid JSON format
↪  with the following structure:
{
    "correct_solution": "step-by-step
    ↪  correct solution",
    "incorrect_solution": "step-by-step
    ↪  incorrect solution",
    "explanation": "briefly explain where
    ↪  the conceptual misunderstanding
    ↪  occurred in the incorrect
    ↪  solution."
}

{question}
```

Generate Contrastive Demonstrations–Logical Reasoning Error

```
You are an expert in mathematical
↪  reasoning. Given a complex
↪  mathematical problem, your task is to
↪  provide a correct solution and an
↪  incorrect solution that specifically
↪  involves a **logical reasoning
↪  error**.

!!!Instructions:
1. Logical reasoning errors occur when
↪  there is a fault in the logical
↪  progression of steps or assumptions.
↪  This could involve using invalid
↪  assumptions, ignoring certain
↪  conditions in the question, or making
↪  incorrect inferences based on
↪  available information.
2. For both the correct and incorrect
↪  solutions, give step by step reasoning
↪  before you answer, and when you're
↪  ready to answer, please use the format
↪  "The final answer is \\boxed{answer}."
3. For the incorrect solution, Naturally
↪  connect to the given mistake as if it
↪  were a valid solution process. Avoid
↪  explicitly pointing out or analyzing
↪  the mistake in the process. Present
↪  the reasoning as if it were a genuine
↪  attempt to solve the problem.
4. Finally, include an explanation that
↪  identifies and points out where the
↪  logical reasoning errors occurred.

Example of a Logical Reasoning error:
In a probability question, a student might
↪  incorrectly assume events are
↪  independent when they are actually
↪  dependent, leading to an incorrect
↪  final answer.

Output Format:
Provide the output in valid JSON format
↪  with the following structure:
{
    "correct_solution": "step-by-step
    ↪  correct solution",
    "incorrect_solution": "step-by-step
    ↪  incorrect solution",
    "explanation": "briefly explain where
    ↪  the logical reasoning error
    ↪  occurred in the incorrect
    ↪  solution."
}

{question}
```

Generate Contrastive Demonstrations–Calculation Error

```
You are an expert in mathematical
↪  reasoning. Given a complex
↪  mathematical problem, your task is to
↪  provide a correct solution and an
↪  incorrect solution that specifically
↪  involves a **calculation error**.

!!!Instructions:
1. Calculation errors occur when an
↪  arithmetic mistake, unit conversion
↪  error, or other computational
↪  inaccuracy leads to an incorrect
↪  result. This type of error does not
↪  arise from a misunderstanding of the
↪  concept, but from a misstep in
↪  executing the steps correctly.
2. For both the correct and incorrect
↪  solutions, give step by step reasoning
↪  before you answer, and when you're
↪  ready to answer, please use the format
↪  "The final answer is \\boxed{answer}."
3. For the incorrect solution, Naturally
↪  connect to the given mistake as if it
↪  were a valid solution process. Avoid
↪  explicitly pointing out or analyzing
↪  the mistake in the process. Present
↪  the reasoning as if it were a genuine
↪  attempt to solve the problem.
4. Finally, include an explanation that
↪  identifies and points out where the
↪  calculation errors occurred.

Example of a Calculation Error:
In solving a physics problem, a student
↪  might correctly set up the equation
↪  but make an arithmetic mistake while
↪  simplifying, resulting in a final
↪  answer that does not match the correct
↪  one.

Output Format:
Provide the output in valid JSON format
↪  with the following structure:
{
    "correct_solution": "step-by-step
    ↪  correct solution",
    "incorrect_solution": "step-by-step
    ↪  incorrect solution",
    "explanation": "briefly explain where
    ↪  the calculation error occurred in
    ↪  the incorrect solution."
}

{question}
```

Produce Grouped Kickoff Solutions–Granularity 1

```
You are a helpful assistant capable of
↪  solving mathematical problems through
↪  step-by-step reasoning. You need to
↪  solve the problem by breaking it into
↪  several simple and manageable steps.

You will be provided with:
1. A problem with its contrastive
↪  solution, e.g., correct vs. incorrect
↪  solution. (The incorrect solution
↪  provided contains a **{error_type}**.
↪  {error_description})
2. A new problem that needs to be solved
↪  correctly.

Your task is to:
1. Integrate insights gained from
↪  analyzing both the correct and
↪  incorrect solutions, highlighting
↪  strong reasoning patterns while
↪  avoiding similar pitfalls.
2. Solve the problem step-by-step,
↪  performing no more than 3 basic
↪  operations per step. Ensure each step
↪  is easy to understand and verify. The
↪  operations should be small and easy to
↪  understand, similar to what a beginner
↪  might perform when solving a basic
↪  arithmetic problem.
3. Stay within the completely feasible
↪  reasoning boundary, ensuring high
↪  accuracy and comprehensibility. Make
↪  sure each planning step is simple,
↪  requiring minimal mental effort, and
↪  each calculation involves basic
↪  arithmetic to stay well within the
↪  feasible boundaries of both planning
↪  and calculation.

Problem:
{old_problem}

Correct Solution:
{correct_solution}

Incorrect Solution:
{incorrect_solution}
({explanation})

New Problem:
{new_problem}

Given the new problem above, provide step
↪  by step reasoning before you answer,
↪  and when you're ready to answer,
↪  please use the format "The final
↪  answer is \\boxed{answer}."
```

Produce Grouped Kickoff Solutions–Granularity 2

```
You are a capable assistant skilled in
↪  solving moderately complex
↪  mathematical problems. You need to
↪  solve the problem by planning multiple
↪  intermediate steps.

You will be provided with:
1. A problem with its contrastive
↪  solution, e.g., correct vs. incorrect
↪  solution. (The incorrect solution
↪  provided contains a **{error_type}**.
↪  {error_description})
2. A new problem that needs to be solved
↪  correctly.

Your task is to:
1. Integrate insights gained from
↪  analyzing both the correct and
↪  incorrect solutions, highlighting
↪  strong reasoning patterns while
↪  avoiding similar pitfalls.
2. Solve the problem step-by-step,
↪  performing around 5-7 operations per
↪  step. Each step should be efficient
↪  but not overly complex. The operations
↪  should be straightforward enough for
↪  someone with basic mathematical
↪  knowledge to follow.
3. Balance step complexity and clarity,
↪  ensuring calculations are
↪  understandable while keeping reasoning
↪  efficient. Combine calculations where
↪  possible to reduce the total number of
↪  steps but avoid exceeding 7 operations
↪  per step to maintain clarity.
4. Stay within the partially feasible
↪  reasoning boundary, ensuring each
↪  planning step reduces the overall
↪  problem complexity without becoming
↪  overly burdensome.

Problem:
{old_problem}

Correct Solution:
{correct_solution}

Incorrect Solution:
{incorrect_solution}
({explanation})

New Problem:
{new_problem}

Given the new problem above, provide step
↪  by step reasoning before you answer,
↪  and when you're ready to answer,
↪  please use the format "The final
↪  answer is \\boxed{answer}."
```

Produce Grouped Kickoff Solutions–Granularity 3

```
You are an advanced assistant capable of
↪  solving complex mathematical problems
↪  efficiently. You need to solve the
↪  problem by utilizing a highly
↪  efficient multi-step reasoning
↪  approach.

You will be provided with:
1. A problem with its contrastive
↪  solution, e.g., correct vs. incorrect
↪  solution. (The incorrect solution
↪  provided contains a **{error_type}**.
↪  {error_description})
2. A new problem that needs to be solved
↪  correctly.

Your task is to:
1. Integrate insights gained from
↪  analyzing both the correct and
↪  incorrect solutions, highlighting
↪  strong reasoning patterns while
↪  avoiding similar pitfalls.
2. Solve the problem step-by-step,
↪  performing as many basic operations as
↪  possible without exceeding
↪  computational complexity that risks
↪  mistakes or confusion. Aim for around
↪  10-15 operations per step, ensuring
↪  each step is a major logical
↪  progression toward solving the
↪  problem.
3. Combine calculations strategically to
↪  minimize the total number of global
↪  steps while maintaining clarity and
↪  logical progression. Ensure each step
↪  is computationally sound and logically
↪  advanced.
4. Stay within the extended reasoning
↪  boundary, pushing the limits of
↪  feasible reasoning and calculation
↪  while avoiding critical mistakes.

Problem:
{old_problem}

Correct Solution:
{correct_solution}

Incorrect Solution:
{incorrect_solution}
({explanation})

New Problem:
{new_problem}

Given the new problem above, provide step
↪  by step reasoning before you answer,
↪  and when you're ready to answer,
↪  please use the format "The final
↪  answer is \\boxed{answer}."
```

F.2 Prompt for Stage 2

The First Two Rounds of Judging

```
You are an impartial judge who can
↪  evaluate the quality of the solutions
↪  provided by two AI assistants to the
↪  problem. Your job is to assess which
↪  assistant's solution is better.

Evaluation Criteria:
1. Mathematical Soundness
- **Priority**: Highest
- **Evaluation Focus**:
  - Are all calculations, logical steps,
  ↪  and applications of mathematical
  ↪  concepts correct?
  - Is the reasoning complete, rigorous,
  ↪  and consistent with established
  ↪  mathematical principles?
  - Does the solution account for edge
  ↪  cases, ambiguities, or unstated
  ↪  assumptions in the problem?

2. Clarity and Explainability
- **Priority**: High
- **Evaluation Focus**:
  - Is the solution well-structured and
  ↪  easy to follow for the intended
  ↪  audience?
  - Are steps presented in a logical
  ↪  sequence with sufficient
  ↪  explanation?
  - Does it use examples, diagrams, or
  ↪  other aids to clarify complex
  ↪  reasoning where needed?

3. Elegance and Simplicity
- **Priority**: Medium
- **Evaluation Focus**:
  - Does the solution achieve the result
  ↪  in a clear and efficient way without
  ↪  unnecessary steps?
  - Does it showcase mathematical elegance
  ↪  by simplifying or illuminating the
  ↪  problem effectively?
  - Is there a balance between conciseness
  ↪  and thoroughness?

4. Mathematical Creativity and Depth
- **Priority**: Medium
- **Evaluation Focus**:
  - Does the solution use innovative or
  ↪  insightful techniques to address
  ↪  the problem?
  - Does it demonstrate a deeper
  ↪  understanding by connecting the
  ↪  solution to broader mathematical
  ↪  principles?
  - Can the method be generalized to other
  ↪  problems or provide additional
  ↪  insights?

Prioritization:
- **Essential Criteria**: Mathematical
↪  Soundness, Clarity and Explainability.
- **Differentiating Factors**: Elegance
↪  and Simplicity, Mathematical
↪  Creativity and Depth.

Instructions:
- Solutions should NOT contain more/less
↪  than what the problem asks for, as
↪  such solutions do NOT precisely answer
↪  the problem.
- Avoid any potential bias and ensure your
↪  judgment is as objective as possible.
↪  The order in which the solutions were
↪  presented should NOT affect your
↪  judgment, as Solution A and Solution B
↪  are **equally likely** to be the
↪  better.

Output Format:
- Provide a concise overall reason for
↪  your judgment, mentioning the key
↪  strengths or weaknesses that led to
↪  your decision.
- Give a verdict indicating which solution
↪  is better, with a choice of A or B.

Provide the output in valid JSON format
↪  with the following structure:
{
    "reason": "your reason here",
    "verdict": "A or B"
}

Problem:
{problem}

Solution A:
{solution1}

Solution B:
{solution2}

Please respond in the specified format.
```

The Final Judging

```
You are an impartial senior judge tasked
↪  with evaluating the quality of
↪  solutions provided by two AI
↪  assistants to a given problem. When
↪  two junior judges have reached
↪  conflicting evaluations, your
↪  intervention is required. You will be
↪  presented with their respective
↪  explanations and assessment results,
↪  and your job is to deliver the final
↪  judgment.

Evaluation Criteria:
1. Mathematical Soundness
- **Priority**: Highest
- **Evaluation Focus**:
  - Are all calculations, logical steps,
  ↪  and applications of mathematical
  ↪  concepts correct?
  - Is the reasoning complete, rigorous,
  ↪  and consistent with established
  ↪  mathematical principles?
  - Does the solution account for edge
  ↪  cases, ambiguities, or unstated
  ↪  assumptions in the problem?

2. Clarity and Explainability
- **Priority**: High
- **Evaluation Focus**:
  - Is the solution well-structured and
  ↪  easy to follow for the intended
  ↪  audience?
  - Are steps presented in a logical
  ↪  sequence with sufficient
  ↪  explanation?
  - Does it use examples, diagrams, or
  ↪  other aids to clarify complex
  ↪  reasoning where needed?

3. Elegance and Simplicity
- **Priority**: Medium
- **Evaluation Focus**:
  - Does the solution achieve the result
  ↪  in a clear and efficient way without
  ↪  unnecessary steps?
  - Does it showcase mathematical elegance
  ↪  by simplifying or illuminating the
  ↪  problem effectively?
  - Is there a balance between conciseness
  ↪  and thoroughness?

4. Mathematical Creativity and Depth
- **Priority**: Medium
- **Evaluation Focus**:
  - Does the solution use innovative or
  ↪  insightful techniques to address
  ↪  the problem?
  - Does it demonstrate a deeper
  ↪  understanding by connecting the
  ↪  solution to broader mathematical
  ↪  principles?
  - Can the method be generalized to other
  ↪  problems or provide additional
  ↪  insights?

Prioritization:
- **Essential Criteria**: Mathematical
↪  Soundness, Clarity and Explainability.
- **Differentiating Factors**: Elegance
↪  and Simplicity, Mathematical
↪  Creativity and Depth.

Output Format:
- Provide a concise overall reason for
↪  final your judgment, mentioning the
↪  key strengths or weaknesses that led
↪  to your decision.
- Give a final verdict indicating which
↪  solution is better, with a choice of A
↪  or B.

Provide the output in valid JSON format
↪  with the following structure:
{
    "reason": "your reason here",
    "verdict": "A or B"
}

Problem:
{problem}

Solution A:
{solution1}

Solution B:
{solution2}

Judge 1 thinks Solution {better1} is
↪  better with the following reason:
{reason1}

Judge 2 thinks Solution {better2} is
↪  better with the following reason:
{reason2}

Please respond in the specified format.
```

Generate Matchup Replay

```
You are an analysis expert. You will be
↪  provided with pairwise comparisons (PK
↪  information) between a specific
↪  solution and two other solutions. Each
↪  PK segment includes the verdicts and
↪  reasons provided by two primary
↪  judges. If their opinions conflict, a
↪  senior judge is introduced, and its
↪  verdict and reasons take precedence.
↪  Your task is to analyze and integrate
↪  these information to identify the
↪  weaknesses of the **current
↪  solution**, as well as the useful
↪  insights learned from comparisons, to
↪  help improve the solution effectively.

Instructions:
- When the opinions of the two primary
↪  judges conflict, prioritize the senior
↪  judge's verdict and reasons as the
↪  authoritative source. Carefully
↪  identify conflicting points in the
↪  primary judges' opinions and exclude
↪  these conflicts from the final
↪  analysis.
- Focus on extracting only the most
↪  relevant and impactful points from the
↪  information, avoiding unnecessary
↪  details or repetition.
- Ensure that insights are concise,
↪  meaningful, and framed at a strategic
↪  or high level to provide actionable
↪  guidance.
- Ensure that all significant
↪  observations, whether explicit or
↪  implicit, are captured and accurately
↪  represented.

Provide the output in valid JSON format
↪  with the following structure:
{
    "weaknesses": ["weakness 1", "weakness
    ↪  2", ...],
    "comparative_insights": ["insight 1",
    ↪  "insight 2", ...],
}

PK Information:
```

```
1. Current Solution vs. Rival Solution 1:
{pk_info1}

2. Current Solution vs. Rival Solution 2:
{pk_info2}

Please respond in the specified format.
```

Reflection

```
Your solution was compared against two alternative solutions (Rival Solution 1 and Rival Solution 2). The outcomes are as follows:

- **Comparison 1 (against Rival Solution 1)**: {winner1} was deemed superior.
- **Comparison 2 (against Rival Solution 2)**: {winner2} was deemed superior.

Based on these comparisons, here is the feedback:
- **Weaknesses**:
{weaknesses}
- **Comparative Insights**:
{insights}

Instructions:
1. Reflect on the feedback and revise your solution to:
    - Address weaknesses if you agree with them.
    - Leverage useful insights.
    - Enhance its overall quality.
2. Only provide the revised solution, without any additional text.
3. Conclude with "The final answer is \\boxed{answer}."

Give your revised solution:
```