CoMAT: Chain of Mathematically Annotated Thought Improves Mathematical Reasoning

Anonymous ACL submission

Abstract

Mathematical reasoning remains a significant challenge for large language models (LLMs), despite progress in prompting techniques such as Chain-of-Thought (CoT). We present **Chain of Mathematically Annotated Thought (CoMAT)**, which enhances reasoning through two stages: *Symbolic Conversion* (converting natural language queries into symbolic form) and *Reasoning Execution* (deriving answers from symbolic representations). CoMAT operates entirely with a single LLM and without external solvers. Across four LLMs, CoMAT outperforms traditional CoT on six out of seven benchmarks, achieving gains of 4.48% on MMLU-Redux (MATH) and 4.58% on GaoKao MCQ. In addition to improved performance, CoMAT ensures faithfulness and verifiability, offering a transparent reasoning process for complex mathematical tasks[1].

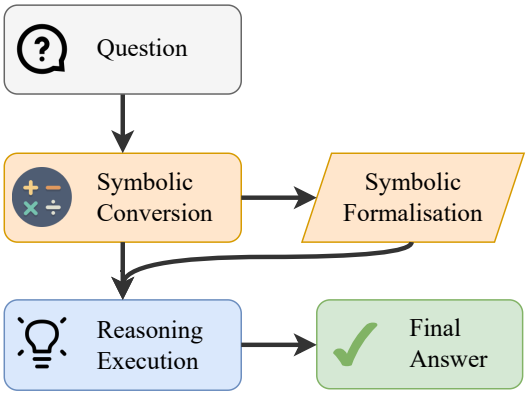


Figure 1: An overview of our CoMAT framework. CoMAT divides complex reasoning tasks into two stages: *Symbolic Conversion*, where queries are translated into structured symbolic reasoning chains (Figure 2a), and *Reasoning Execution*, where step-by-step calculations are performed to derive the final answer (Figure 2b).

1 Introduction

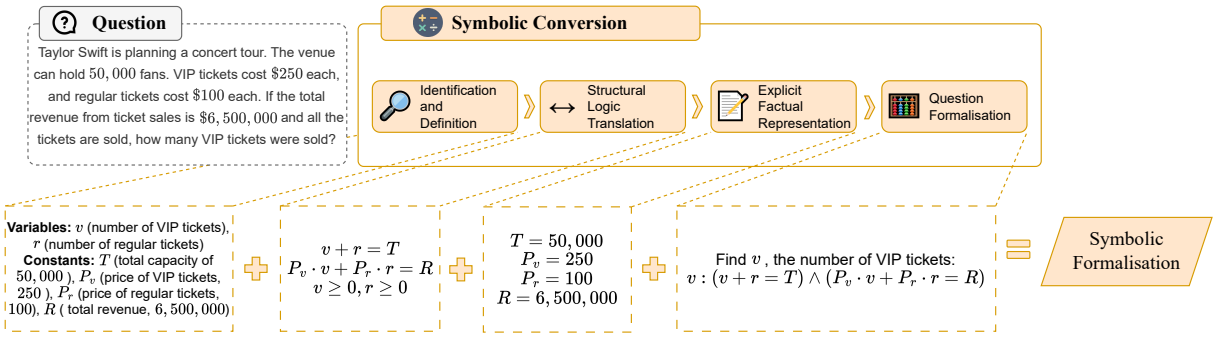
Complex mathematical reasoning remains a significant challenge for large language models (LLMs; Luo et al., 2024; Li et al., 2023; Meadows and Freitas, 2023). Techniques like Chain-of-Thought (CoT) prompting (Wei et al., 2022; Kojima et al., 2022; Wang et al., 2022) have improved LLM performance by encouraging the generation of intermediate reasoning steps. However, CoT explanations are not always faithful to the actual reasoning process of the model (Bentham et al., 2024; Turpin et al., 2024; Yee et al., 2024); with final answers that may not logically follow from the reasoning chain, suggesting that LLMs can fabricate reasoning paths (Lyu et al., 2023).

Recent efforts to improve reasoning faithfulness have relied on integrating external solvers (Lyu et al., 2023; He-Yueya et al., 2023; Jiang et al., 2024a). However, these approaches introduce new challenges: LLMs often generate erroneous or syntactically invalid code when translating natural language into formal statements, causing solvers to fail (Wen et al., 2024).

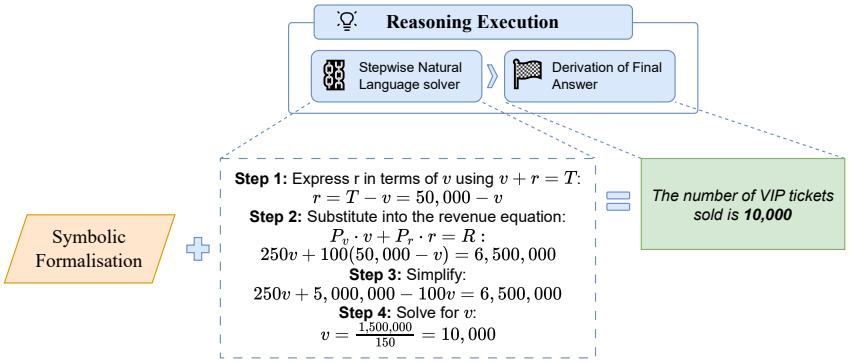
To address these issues, we propose **Chain of Mathematically Annotated Thought (CoMAT)**, a novel approach that leverages symbolic reasoning entirely within LLMs. By eliminating external solvers, CoMAT avoids issues related to code generation failures, offering a more robust solution for a broad range of mathematical tasks. Notably, CoMAT performs well where there are difficulties with code-based methods, such as multilingual datasets and Olympiad-level problems. As shown in Appendix A, code-based methods often struggle with Olympiad-level mathematics, while CoMAT demonstrates greater adaptability. To our knowledge, CoMAT is the first method to apply symbolic reasoning across diverse mathematical reasoning tasks, including multilingual datasets, Olympiad-level problems, and common benchmarks.

Empirically, CoMAT consistently outperforms

[1]Code is available at https://anonymous.4open.science/r/CoMAT-FDFB/



(a) An overview of the Symbolic Conversion stage of CoMAT, which includes *Indentification and Definition*, *Structural Logic Translation*, *Explicit Factual Representation*, and *Question Formalisation*.



(b) An overview of the Reasoning Execution stage of CoMAT, which performs 'step by step' reasoning based on the symbolic representation to provide the final answer.

Figure 2: An overview of CoMAT divided into two main stages: Symbolic Conversion and Reasoning Execution

existing baselines. When evaluated on datasets ranging from standard mathematical problems to complex Olympiad-level challenges in both English and Mandarin, CoMAT showed significant improvements. When applied to Gemini-1.5-Pro, we achieved an overall exact match boost of 3.54% on AQUA and an 8.18% increase in accuracy on the Olympiad Bench (English) compared to CoT. Similarly, using Qwen2-72B as a baseline, we observed a 2.55% improvement averaged across mathematical subjects of the MMLU-Redux benchmark. These results highlight CoMAT's effectiveness in enhancing accuracy across diverse and challenging tasks.

To summarise, our key contributions are:

- We introduce CoMAT, a novel approach that leverages symbolic reasoning entirely within LLMs, eliminating the need for external tools or verifiers. This ensures both accuracy and verifiability through a transparent and structured reasoning process.
- Our symbolic prompts are standardised to tackle an array of mathematical reasoning tasks, achieving state-of-the-art performance across diverse datasets of varying complexity and languages.
- We perform a comprehensive ablation study demonstrating the generalisability of CoMAT across multiple LLMs, datasets, prompt designs, and languages.

2 CoMAT: Chain of Mathematically Annotated Thought

Traditional CoT methods in mathematical reasoning rely heavily on natural language explanations, which can introduce ambiguity and inconsistencies (Lu et al., 2024). These methods may lead models to fabricate reasoning paths or misinterpret variable relationships, compromising accuracy in complex tasks. CoMAT addresses these limitations by adopting a structured, symbolic approach that enforces mathematical consistency and reduces ambiguity (Tam et al., 2024). By using well-defined symbolic logic, CoMAT ensures that each reasoning step adheres to sound mathematical principles, enhancing the model's ability to solve problems accurately (Zhong et al., 2024).

CoMAT extends traditional CoT by incorporating symbolic transformations as the core compo-

nent of the reasoning process. While CoT typically follows a (Q, R, A) structure — where Q is the question, R is the reasoning, and A is the final answer — CoMAT introduces a more structured process: (Q, S, R, A), where $S = (s_1, s_2, s_3, s_4)$ represents the four steps in the symbolic reasoning pipeline, designed to break down complex problems into formal, interpretable sequences of logical operations. This structured decomposition enhances transparency and allows for systematic verification of each step. Examples of manual annotation for verifiability can be found in Appendix B.

2.1 CoMAT's Two-Stage Process

CoMAT consists of two main stages: **Symbolic Conversion** and **Reasoning Execution**. Each stage is critical to ensuring the accuracy and faithfulness of the reasoning process.

Symbolic Conversion. In this stage, the LLM transforms a natural language query Q into a symbolic representation, $S = (s_1, s_2, s_3, s_4)$. This involves *identification and definition*, *structural logic translation*, *explicit factual representation*, and *question formalisation*, all carried out by the LLM. This stage acts as the foundation for accurate reasoning by converting ambiguous natural language into well-structured mathematical logic.

Reasoning Execution. Once the problem is translated into its symbolic form (S), the model applies logical reasoning to derive the solution. The logical reasoning is followed by stepwise reasoning, similar to Kojima et al. (2022) –*i.e.,* prompting the model with the phrase "Let's think step-by-step". By grounding the reasoning in the symbolic structure, CoMAT ensures that each step aligns with mathematical logic, reducing the risk of errors or illogical steps. The final answer, A, is then generated based on this rigorous reasoning process.

2.2 Case Study

To demonstrate how CoMAT works in practice, consider the following math problem related to ticket sales, based on Figure 2:

> *Taylor Swift is planning a concert tour. The venue can hold 50,000 fans. VIP tickets cost $250 each, and regular tickets cost $100 each. If the total revenue from ticket sales is $6,500,000 and all tickets are sold, how many VIP tickets were sold?*

1. **Identification and Definition.** CoMAT first identifies and defines the relevant variables and constants to ensure precision in the symbolic reasoning process. For instance:

> - **Variables**: v (number of VIP tickets), r (number of regular tickets)
> - **Constants**: T (total capacity of 50,000), P_v (price of VIP tickets, 250), P_r (price of regular tickets, 100), R (total revenue, $6,500,000$)

2. **Structural Logic Translation**: Next, CoMAT extracts the key variables and translates the problem into formal rules that define their relationships, ensuring the reasoning process is grounded in well-defined constraints:

> - $v + r = T$
> - $P_v \cdot v + P_r \cdot r = R$
> - $v \geq 0, r \geq 0$ (non-negative constraints)

3. **Explicit Factual Representation**: CoMAT then integrates all relevant facts into the logical structure to avoid omitting key information:

> - $T = 50,000$
> - $P_v = 250$
> - $P_r = 100$
> - $R = 6,500,000$

4. **Question Formalisation**: CoMAT formalises the question into symbolic expression to ensure that the reasoning process remains objective and free of bias from answer options. However, the model may inherently choose to parse the question in natural language rather than a symbolic representation. In this case:

> In our example, we are tasked with finding v, the number of VIP tickets:
> Find $v : (v + r = T) \wedge (P_v \cdot v + P_r \cdot r = R)$

5. **Reasoning Execution**: The problem is then solved step-by-step using the symbolic representation, as demonstrated:

Step 1: Express r in terms of v using $v + r = T$:

$$r = T - v = 50,000 - v$$

Step 2: Substitute into the revenue equation $P_v \cdot v + P_r \cdot r = R$:

$$250v + 100(50,000 - v) = 6,500,000$$

Step 3: Simplify:

$$250v + 5,000,000 - 100v = 6,500,000$$

Step 4: Solve for v:

$$v = \frac{1,500,000}{150} = 10,000$$

6. **Derivation of Final Answer**: The final answer is then derived solely based on the logical reasoning applied. In this case:

> The number of VIP tickets sold is 10,000.

7. (optional) **Answer Matching**: In multiple-choice QA tasks, CoMAT match the final answer to the most similar one from the provided options, without considering the order or labelling of the options. This final step ensures that the symbolic reasoning remains unbiased by the format of the answer choices.

This step-by-step symbolic reasoning ensures that each operation is transparent and verifiable, reducing the potential for errors often introduced in purely natural language reasoning. CoMAT's structured methodology facilitates easier error tracing, as shown in Figure 3. In cases where the final answer is incorrect, each step can be individually examined to identify mistakes — an advantage over traditional CoT, where explanations may not align with the actual reasoning steps, making the reasoning process less precise or harder to verify (Turpin et al., 2024; Li et al., 2024).

3 Experimental Setup

Datasets. We selected a total of seven datasets, including four in English: AQUA (Ling et al., 2017) and GSM8K (Cobbe et al., 2021), which were extracted from Math Word Problem (MWP) datasets,

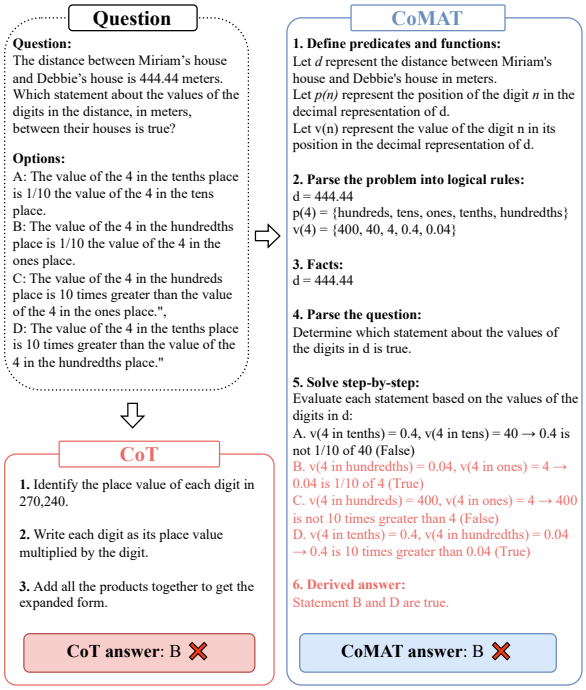


Figure 3: An example question from the MMLU Redux Elementary Mathematics dataset, comparing CoT and CoMAT. CoT follows a generic "step-by-step" approach without further guidance. In contrast, CoMAT enhances interpretability and verifiability by clearly pinpointing the error, which in this case arises from Step 5. Traditional CoT, by comparison, lacks the ability to identify specific errors directly.

MMLU-Redux (Gema et al. 2024); covering various mathematics subjects, including abstract algebra, college-level mathematics, high school mathematics, and elementary mathematics), Olympiad Bench (English; OlympBench-EN; He et al. 2024), focused on text-based olympiad-mathematics problems, and another two Mandarin datasets: GaoKao (MCQ; Zhang et al. 2023), Olympiad Bench (Chinese; OlympBench-CN; He et al. 2024). These datasets include multiple-choice questions (AQUA, MMLU-Redux, GaoKao MCQ) and string-valued answers (GSM8K, Olympiad Bench). The inclusion of complex datasets like GaoKao and Olympiad allows us to assess model performance on academically challenging, non-standard, questions, such as those in Olympiad-level exams and specialised academic tasks.

Evaluation Metrics. For multiple-choice datasets, we used exact match metrics, requiring the predicted answer to fully match the correct one, not just the option code (e.g., *A*, *B*). This ensures evaluation based on complete responses, addressing clarity concerns in datasets like MMLU-Redux.

For string-valued answers, we also used exact match in GSM8K. On the Olympiad Bench, we used GPT-4o-mini as a benchmark to evaluate how well the model's answers aligned with the ground truth. Further details about GPT-4o-mini evaluation can be found in Appendix C.

Language Models. We conducted our experiments using GPT-4o (Achiam et al., 2023), Gemini (Team et al., 2023), and Qwen2 (Yang et al., 2024). This variety allows us to examine performance across different model architectures, sizes, and language proficiencies. For Faithful CoT, we used GPT-4, following the baselines from Lyu et al. (2023), as GPT-4o tends to produce more invalid outputs compared to GPT-4.

Baselines. We compared our method against two baselines using the same decoding strategies: greedy decoding, where the most probable next token is selected (temperature = 0.0). The baselines include: (1) Standard Prompting, using a similar prompt to that in Holistic Evaluation of Language Models (Lee et al., 2024); (2) CoT prompting, famous "Let's think step by step" (Kojima et al., 2022). For common benchmarks, we included Faithful CoT for additional comparison. In cases where results were not available for specific tasks, we reproduced the baseline results to ensure consistency and fairness in comparison.

4 Results

Table 1 compares CoMAT, CoT (Kojima et al., 2022), Faithful CoT (Lyu et al., 2023) and baseline models across various benchmarks. CoMAT consistently outperforms traditional CoT and Faithful CoT in most datasets, particularly on tasks requiring advanced mathematical reasoning.

For open-source models, **Qwen2-7b** shows significant improvements with CoMAT. Under CoT, it often struggle to provide reasoning on OlympiadBench dataset. With CoMAT, its performance increases on English OlympiadBench (5.19% → 20.92%), and Chinese OlympiadBench (8.09% → 13.24%). **Qwen2-72b** also benefits from CoMAT, especially on OlympiadBench and GaoKao, with English OlympiadBench accuracy increasing (27.74% → 32.17%), highlighting CoMAT's ability to improve reasoning in models that initially underperform.

Among closed-source models, **Gemini** shows notable gains with CoMAT, with an 8.18% improvement on English OlympiadBench. Similarly, **GPT-4o** shows substantial gains on Mandarin datasets, with performance on GaoKao rising from 63.27% (CoT) to 71.43% (CoMAT).

However, there are some cases where CoMAT does not outperform CoT. For instance, **Qwen2-72b** and **GPT-4o** show declines on AQUA (79.13% → 72.44% and 84.25% → 83.46%, respectively). These decreases suggest that on simpler tasks where models already perform well, the added complexity of symbolic reasoning may not yield significant benefits.

When comparing CoMAT with Faithful CoT on MWP datasets, Faithful CoT shows a minor gain on GSM8K (93.70% → 95.0%). However, CoMAT demonstrates a significant 9.86% improvement on AQUA, highlighting its ability to outperform Faithful CoT without relying on external solvers. CoMAT also surpasses Faithful CoT on MMLU-Redux. We excluded Faithful CoT from comparisons on GaoKao and OlympiadBench due to its inability to execute most questions using external solvers, as outlined in Appendix A. CoMAT effectively mitigates these limitations, proving its capability to handle complex tasks without reliance on external solvers.

Generally, we observe the most significant gains on the challenging datasets (GaoKao and OlympiadBench), which require advanced reasoning. For instance, CoMAT improves average performance on the English OlympiadBench (23.68% → 30.74%), and GaoKao (56.13% → 60.70%). This suggests that CoMAT is particularly effective at enhancing reasoning capabilities for complex tasks. On simpler datasets like AQUA, where the base models already perform well, the average gains are smaller but still present, such as a 4.48% increase at MMLU-Redux. The average results for each model are illustrated in Figure 4.

Overall, integrating symbolic reasoning into the Chain-of-Thought process significantly enhances language models' ability to tackle complex mathematical reasoning. The improvements are more substantial on advanced tasks, aligning with CoMAT's design to handle intricate reasoning steps through structured symbolic representations.

4.1 Step Contribution Analysis

We conducted an ablation study to assess the impact of each step in the CoMAT prompt on model performance. We tested 16 variations by removing Steps 1, 2, 3, or 4 individually, as well as combi-

Model	MWP		English			Chinese		
	AQUA	GSM8K	MMLU-Redux	OlympBench	Average (EN+MWP)	GaoKao	OlympBench	Average (CN)
Open Source Models								
Qwen2-7b	34.25%	19.29%	56.52%	5.19%	28.81%	**48.98%**	8.09%	28.54%
+ CoT	33.07%	**81.78%**	64.80%	3.86%	45.88%	40.82%	4.17%	22.50%
+ CoMAT (Ours)	**42.13%**	79.80%	**79.80%**	**20.92%**	**55.66%**	44.90%	**13.24%**	**29.07%**
Qwen2-72b	51.97%	37.43%	66.53%	10.39%	41.58%	53.06%	11.27%	32.17%
+ CoT	**79.13%**	82.76%	79.17%	27.74%	67.20%	55.10%	**19.66%**	37.38%
+ CoMAT (Ours)	72.44%	**83.90%**	**81.72%**	**32.17%**	**67.56%**	**59.18%**	18.87%	**39.03%**
Closed Source Models								
Gemini	48.03%	45.71%	68.00%	5.50%	41.92%	43.00%	14.95%	28.98%
+ CoT	75.20%	**90.51%**	79.55%	21.28%	66.64%	65.31%	13.27%	39.29%
+ CoMAT (Ours)	**78.74%**	90.43%	**79.71%**	**29.46%**	**69.58%**	**67.30%**	**15.95%**	**41.63%**
GPT-4o	44.49%	56.72%	59.70%	9.94%	42.71%	36.73%	8.82%	22.78%
+ CoT	**84.25%**	**94.46%**	88.10%	**41.84%**	**77.17%**	63.27%	23.53%	43.40%
+ CoMAT (Ours)	83.46%	93.70%	**88.30%**	40.42%	76.48%	**71.43%**	**26.47%**	**48.95%**
Additional Benchmarks								
Faithful CoT	73.6%	**95.0%**	76.88%	-	-	-	-	-

Table 1: Performance comparison of models across multiple benchmarks, separated by language (EN, CN). While methods like Faithful CoT focus on MWP tasks, we compare CoMAT with Faithful CoT (Lyu et al., 2023) common benchmarks. However, as Faithful CoT fails to solve most questions in *OlympiadBench* and *GaoKao*, generating invalid answers, we exclude it from those datasets and instead use CoT (Kojima et al., 2022) as main comparison.

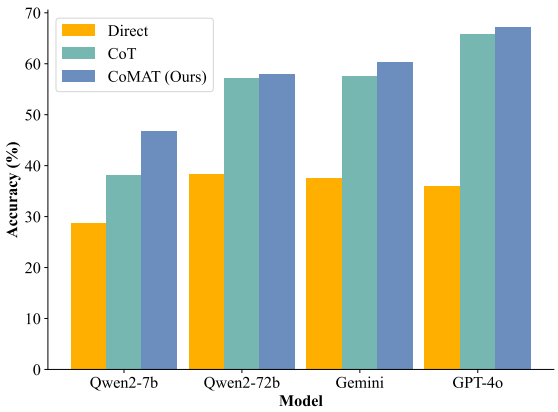


Figure 4: Average performance across all datasets for each model.

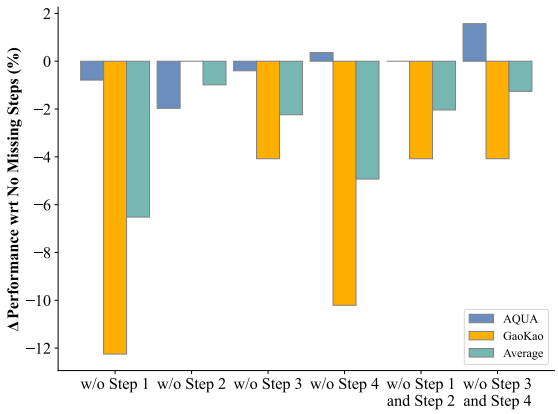
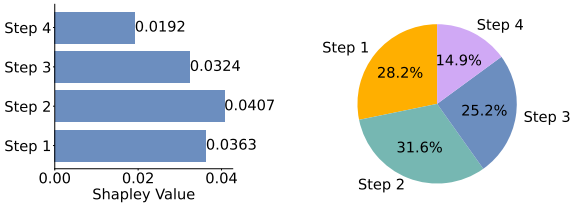


Figure 5: Performance change (Δ) for each configuration with missing steps. Detailed results for all complete variants are provided in Appendix D.1.

nations (*i.e.,* removing steps 1 and 2 or removing steps 3 and 4).

Figure 5 shows the change in accuracy compared to the full CoMAT prompt. We found that each individual step plays a crucial role in maintaining the accuracy of CoMAT. Removing Step 1 leads to the most significant drop in accuracy (6.91%). This suggests that Step 1 is fundamental to CoMAT, potentially serving as the foundation upon which other steps build; its absence may disrupt critical initial processes or data preparation necessary for accurate performance. Omitting Step 2 or Step 3 results in smaller declines (1.38% and 2.63% respectively), indicating these steps are important but less critical than Step 1. Interestingly, removing both Steps 1 and 2 results in a smaller performance drop (2.43%) than removing Step 1 alone, suggesting overlapping functionalities where the model compensates using remaining steps. Overall, the full CoMAT prompt achieves the highest average accuracy of 77.45%, reaffirming the importance of retaining all steps for optimal performance.

We further quantified each step's contribution using Shapley-value (Shapley, 1953) analysis (Figure 6). The analysis shows that all steps positively impact performance, with Steps 1 and 2 having the greatest influence, aligning with the expectation



(a) Shapley values for each step.

(b) Percentage impact of steps.

Figure 6: (a) and (b) represent the Shapley values in bar and pie chart form, respectively.

Model	BN	SW	TE	TH	Average
GPT-4o Models					
non-CoT	52.80	52.40	51.60	58.00	53.70%
+CoT	92.00	88.80	86.00	91.20	**89.50%**
+CoMAT	89.60	88.00	84.80	90.80	88.30%
Gemini Models					
non-CoT	40.00	46.00	41.60	42.00	42.40%
+CoT	79.20	70.80	62.40	84.00	74.10%
+CoMAT	82.40	83.20	74.00	79.20	**79.70%**

Table 2: Performance comparison of GPT-4o and Gemini models across low-resource MGSM benchmarks (BN, SW, TE, TH).

that initial steps lay the foundation for subsequent reasoning. For a detailed breakdown of the Shapley analysis calculations, refer to Appendix E.

These results highlight that every step in CoMAT's pipeline is essential, not only for achieving high accuracy but also for enhancing interpretability and verifiability. Each step builds upon the previous ones, ensuring that the reasoning process remains transparent and coherent. This reinforces CoMAT's strength in balancing empirical performance with faithful, interpretable reasoning, making it a robust and reliable framework across diverse mathematical reasoning tasks.

4.2 Multilingual Analysis

To assess the impact of low-resource languages on CoMAT's performance, we evaluate the models on MGSM dataset (Shi et al., 2022), focusing on Swahili (SW), Bengali (BN), Thai (TH), and Telugu (TE). While CoMAT has shown strong performance in high-resource languages like English and Mandarin, our analysis reveals mixed results in low-resource language settings relative to CoT. This suggests that while CoMAT is effective, further optimisation may be needed to fully adapt CoMAT to low-resource languages.

As shown in Table 2, Gemini-1.5-Pro demonstrated notable improvements in low-resource contexts, with an average performance increase of 5.60% after applying CoMAT. Significant gains were observed in Bengali (BN) and Swahili (SW), where accuracy increased to 82.40% and 83.20%, respectively. These improvements indicate that CoMAT can enhance reasoning in lower-resource settings when applied to certain models.

However, the performance of GPT-4o model slightly declines, from 89.50% to 88.30%. This suggests that while CoMAT remains effective overall, certain models might struggle to generalise as effectively in low-resource contexts. In languages like Thai (TH) and Telugu (TE), the model's performance remained relatively stable but did not show the same level of improvement observed in higher-resource languages.

These mixed results indicate that while CoMAT shows promise for improving reasoning in low-resource languages, particularly with models like Gemini, further optimization may be required to fully optimise its performance across models and language contexts with limited training data.

4.3 Answer Order Swapping

Building on the findings of Gupta et al. (2024), which demonstrated that altering answer choices in the MMLU dataset could affect model accuracy, we extended this investigation to both the MMLU-Redux and AQUA datasets using GPT-4o as our baseline. We have also introduced an additional challenge by including a random answer option, as detailed in Appendix D.2.

Figure 7 illustrates the effect of this option swapping. On the AQUA dataset, both CoT and CoMAT models experienced accuracy drops, highlighting their sensitivity to answer structure changes. However, CoMAT was more resilient, with only a 0.17% decrease and a low standard deviation of 0.91%. In contrast, GPT-4o with CoT saw a significant drop of 14.96% and a higher standard deviation of 3.50%, indicating greater inconsistency when options were shuffled. This highlights CoMAT's stability and robustness under altered conditions.

On the MMLU-Redux dataset, GPT-4o baseline showed a slight improvement, with a 1.34% standard deviation. CoT and CoMAT both had minor accuracy decreases, with CoMAT achieving 85.05% (standard deviation of 1.65%), slightly outperforming CoT at 84.43% (1.87% standard deviation). Detailed results can be referred to in Ap-

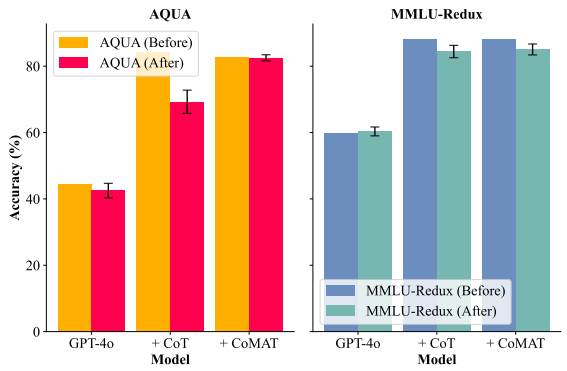


Figure 7: Average accuracy for AQUA and MMLU before and after option changing.

pendix D.1.

These results demonstrate CoMAT's consistency when confronted with complexities of option swapping. Other methods exhibit greater variability and performance sensitivity.

5 Related Work

Logical Reasoning. Logical reasoning tasks require models to handle complex logical structures (Cummins et al., 1991). Traditional methods include rule-based (Robinson, 1965) and neural-based approaches (Amayuelas et al., 2022; Gerasimova et al., 2023) for interpreting symbolic representations. Recent advancements integrate LLMs into symbolic reasoning, introducing frameworks like Logic-LM (Pan et al., 2023), SAT-LM (Ye et al., 2024) and LeanReasoner (Jiang et al., 2024a). These frameworks use LLMs to convert natural language into symbolic syntax processed by external reasoning tools, enhancing performance through self-consistency and non-linear reasoning (Wang et al., 2023; Zhang et al., 2022). However, they often rely on external tools, assuming LLMs cannot parse symbolic expressions as reliably as rule-based reasoners.

Symbolic Chain-of-Thought Prompting. Symbolic CoT prompting (Lyu et al., 2023) combines natural language (NL) and symbolic language (SL) in the reasoning chain. NL decomposes complex queries into subproblems, and SL programs (e.g., Python, LEAN) handle each subproblem. A deterministic solver executes the SL to derive the final answer, ensuring faithfulness while NL aids interpretability. Recent efforts aim to reduce reliance on SL programs by leveraging LLMs (Xu et al., 2024), but these methods focus primarily on logical reasoning rather than complex mathematical tasks and still require verifiers to ensure accuracy.

Mathematical Reasoning. Mathematical reasoning with LLMs has been explored widely (Lewkowycz et al., 2022; Luo et al., 2024; Ahn et al., 2024; Imani et al., 2023; Chen et al., 2024; Meadows and Freitas, 2022; Mirzadeh et al., 2024), with CoT methods yielding significant performance gains (Jiang et al., 2024c; Chu et al., 2023; Ranaldi and Freitas, 2024). Deep problem understanding (Zhong et al., 2024), structured formats (Tam et al., 2024), and building supervision models for reasoning (Lightman et al., 2023; Jiang et al., 2024b) also enhance accuracy. Other studies focus on premise selection and symbolic frameworks for systematic evaluation (Meadows et al., 2023; Ferreira and Freitas, 2020).

6 Conclusion

We propose CoMAT, a simple yet effective framework that decomposes complex mathematical reasoning into two stages — Symbolic Conversion and Reasoning Execution. CoMAT operates entirely within LLMs, eliminating reliance on external solvers and ensuring a transparent and accurate reasoning process. By avoiding external solvers, CoMAT mitigates issues related to code generation failures, providing a more robust solution for a broad range of mathematical tasks. Our analysis highlights four key steps in the CoMAT process and demonstrates its effectiveness across various datasets with different levels of complexity and linguistic diversity, including English, Mandarin, and low-resource languages. CoMAT consistently outperforms traditional CoT, achieving state-of-the-art performance and enhancing consistency when answer options are shuffled, demonstrating robustness and reliability. Despite its simplicity, CoMAT offers a scalable and effective solution for complex mathematical reasoning, providing greater faithfulness and verifiability across a wide range of tasks.

Limitations

While CoMAT demonstrates strong performance, there are several potential limitations. Firstly, our evaluation was limited to the current set of symbolic steps, where some steps have been previously used in other reasoning frameworks. For instance, a step similar to our step 1 is used by Lyu et al.

(2023) and Xu et al. (2024). Although our approach proved effective, further research is needed to assess CoMAT's performance using additional symbolic languages to ensure a more comprehensive evaluation. Secondly, while CoMAT enhances verifiability and faithfulness, it introduces a higher computational overhead compared to CoT due to the structured nature of its formalisations. This process involves generating additional symbolic representations, leading to a larger token count, which increases both computational costs and API usage. Consequently, scaling CoMAT could require more substantial computational resources than traditional CoT methods. Lastly, CoMAT has been focused within the mathematical reasoning. Further research could explore extending symbolic reasoning to other domains, beyond mathematical reasoning, to evaluate its effectiveness in broader tasks.

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*.

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. *arXiv preprint arXiv:2402.00157*.

Alfonso Amayuelas, Shuai Zhang, Xi Susie Rao, and Ce Zhang. 2022. Neural methods for logical reasoning over knowledge graphs. In *International Conference on Learning Representations*.

Oliver Bentham, Nathan Stringham, and Ana Marasović. 2024. Chain-of-thought unfaithfulness as disguised accuracy. *arXiv preprint arXiv:2402.14897*.

Changyu Chen, Xiting Wang, Ting-En Lin, Ang Lv, Yuchuan Wu, Xin Gao, Ji-Rong Wen, Rui Yan, and Yongbin Li. 2024. Masked thought: Simply masking partial reasoning steps can improve mathematical reasoning learning of language models. *arXiv preprint arXiv:2403.02178*.

Zheng Chu, Jingchang Chen, Qianglong Chen, Weijiang Yu, Tao He, Haotian Wang, Weihua Peng, Ming Liu, Bing Qin, and Ting Liu. 2023. A survey of chain of thought reasoning: Advances, frontiers and future. *arXiv preprint arXiv:2309.15402*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Denise D Cummins, Todd Lubart, Olaf Alksnis, and Robert Rist. 1991. Conditional reasoning and causation. *Memory & cognition*, 19:274–282.

Deborah Ferreira and André Freitas. 2020. Premise selection in natural language mathematical texts. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7365–7374.

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, et al. 2024. Are we done with MMLU? *arXiv preprint arXiv:2406.04127*.

Olga Gerasimova, Nikita Severin, and Ilya Makarov. 2023. Comparative analysis of logic reasoning and graph neural networks for ontology-mediated query answering with a covering axiom. *IEEE Access*.

Vipul Gupta, David Pantoja, Candace Ross, Adina Williams, and Megan Ung. 2024. Changing answer order can decrease mmlu accuracy. *arXiv preprint arXiv:2406.19470*.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*.

Joy He-Yueya, Gabriel Poesia, Rose E Wang, and Noah D Goodman. 2023. Solving math word problems by combining language models with symbolic solvers. *arXiv preprint arXiv:2304.09102*.

Shima Imani, Liang Du, and Harsh Shrivastava. 2023. Mathprompter: Mathematical reasoning using large language models. *arXiv preprint arXiv:2303.05398*.

Dongwei Jiang, Marcio Fonseca, and Shay B Cohen. 2024a. Leanreasoner: Boosting complex logical reasoning with lean. *arXiv preprint arXiv:2403.13312*.

Dongwei Jiang, Guoxuan Wang, Yining Lu, Andrew Wang, Jingyu Zhang, Chuyu Liu, Benjamin Van Durme, and Daniel Khashabi. 2024b. Rationalyst: Pre-training process-supervision for improving reasoning. *Preprint*, arXiv:2410.01044.

Zhuoxuan Jiang, Haoyuan Peng, Shanshan Feng, Fan Li, and Dongsheng Li. 2024c. LLMs can find mathematical reasoning mistakes by pedagogical chain-of-thought. *arXiv preprint arXiv:2405.06705*.

Alon Keinan, Ben Sandbank, Claus C Hilgetag, Isaac Meilijson, and Eytan Ruppin. 2004. Fair attribution of functional contribution in artificial and biological networks. *Neural computation*, 16(9):1887–1915.

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.

Tony Lee, Michihiro Yasunaga, Chenlin Meng, Yifan Mai, Joon Sung Park, Agrim Gupta, Yunzhi Zhang, Deepak Narayanan, Hannah Teufel, Marco Bellagente, et al. 2024. Holistic evaluation of text-to-image models. *Advances in Neural Information Processing Systems*, 36.

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.

Jiachun Li, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. 2024. Towards faithful chain-of-thought: Large language models are bridging reasoners. *arXiv preprint arXiv:2405.18915*.

Weixian Waylon Li, Yftah Ziser, Maximin Coavoux, and Shay B Cohen. 2023. BERT is not the count: Learning to match mathematical statements with proofs. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3581–3593.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *arXiv preprint arXiv:1705.04146*.

Zhenyi Lu, Jie Tian, Wei Wei, Xiaoye Qu, Yu Cheng, Dangyang Chen, et al. 2024. Mitigating boundary ambiguity and inherent bias for text classification in the era of large language models. *arXiv preprint arXiv:2406.07001*.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, et al. 2024. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. Faithful chain-of-thought reasoning. *arXiv preprint arXiv:2301.13379*.

Jordan Meadows and Andre Freitas. 2022. A survey in mathematical language processing. *arXiv preprint arXiv:2205.15231*.

Jordan Meadows and André Freitas. 2023. Introduction to Mathematical Language Processing: Informal Proofs, Word Problems, and Supporting Tasks. *Transactions of the Association for Computational Linguistics*, 11:1162–1184.

Jordan Meadows, Marco Valentino, Damien Teney, and Andre Freitas. 2023. A symbolic framework for systematic evaluation of mathematical reasoning with transformers.

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*.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. 2023. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. *arXiv preprint arXiv:2305.12295*.

Leonardo Ranaldi and Andre Freitas. 2024. Aligning large and small language models via chain-of-thought reasoning. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1812–1827.

John Alan Robinson. 1965. A machine-oriented logic based on the resolution principle. *Journal of the ACM (JACM)*, 12(1):23–41.

Lloyd S Shapley. 1953. A value for n-person games. *Contribution to the Theory of Games*, 2.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, et al. 2022. Language models are multilingual chain-of-thought reasoners. *arXiv preprint arXiv:2210.03057*.

Zhi Rui Tam, Cheng-Kuang Wu, Yi-Lin Tsai, Chieh-Yen Lin, Hung-yi Lee, and Yun-Nung Chen. 2024. Let me speak freely? a study on the impact of format restrictions on performance of large language models. *arXiv preprint arXiv:2408.02442*.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. 2024. Language models don't always say what they think: unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems*, 36.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. *arXiv preprint arXiv:2305.04091*.

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

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.

Hao Wen, Yueheng Zhu, Chao Liu, Xiaoxue Ren, Weiwei Du, and Meng Yan. 2024. Fixing code generation errors for large language models. *arXiv preprint arXiv:2409.00676*.

Jundong Xu, Hao Fei, Liangming Pan, Qian Liu, Mong-Li Lee, and Wynne Hsu. 2024. Faithful logical reasoning via symbolic chain-of-thought. *arXiv preprint arXiv:2405.18357*.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Xi Ye, Qiaochu Chen, Isil Dillig, and Greg Durrett. 2024. Satlm: Satisfiability-aided language models using declarative prompting. *Advances in Neural Information Processing Systems*, 36.

Evelyn Yee, Alice Li, Chenyu Tang, Yeon Ho Jung, Ramamohan Paturi, and Leon Bergen. 2024. Dissociation of faithful and unfaithful reasoning in LLMs. *arXiv preprint arXiv:2405.15092*.

Xiaotian Zhang, Chunyang Li, Yi Zong, Zhengyu Ying, Liang He, and Xipeng Qiu. 2023. Evaluating the performance of large language models on gaokao benchmark. *arXiv preprint arXiv:2305.12474*.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022. Automatic chain of thought prompting in large language models. *arXiv preprint arXiv:2210.03493*.

Qihuang Zhong, Kang Wang, Ziyang Xu, Juhua Liu, Liang Ding, Bo Du, and Dacheng Tao. 2024. Achieving> 97% on gsm8k: Deeply understanding the problems makes LLMs perfect reasoners. *arXiv preprint arXiv:2404.14963*.

A Faithful CoT vs CoMAT

Verification of solutions - CoT (Faithful-CoT)

Question:
Xenia and Sergey play the following game. Xenia thinks of a positive integer N not exceeding 5000. Then she fixes 20 distinct positive integers $a_1, a_2, \ldots, a_{20}$ such that, for each $k = 1, 2, \ldots, 20$, the numbers N and a_k are congruent modulo k. By a move, Sergey tells Xenia a set S of positive integers not exceeding 20, and she tells him back the set $\{a_k : k \in S\}$ without spelling out which number corresponds to which index. How many moves does Sergey need to determine for sure the number Xenia thought of?
Answer: 2

Faithful-CoT:

1. What is the maximum number Xenia can think of?
   Xenia thinks of a positive integer N not exceeding 5000.
   - max_number = 5000
   - How many distinct positive integers does Xenia fix?
     She fixes 20 distinct positive integers $a_1, a_2, \ldots, a_{20}$.
   - distinct_integers = 20
   - What is the condition for the numbers N and a_k?
     The numbers N and a_k are congruent modulo k for each $k = 1, 2, \ldots, 20$.
   - Condition: N % k == a_k % k
2. What is a move in the game?
   - Sergey tells Xenia a set S of positive integers not exceeding 20, and she tells him back the set $\{a_k : k \in S\}$ without spelling out which number corresponds to which index.
     Move:Sergey asks set S; Xenia gives $a_k : k \in S$
3. How many moves does Sergey need to determine for sure the number Xenia thought of?
   - This requires finding the least common multiple of the numbers from 1 to 20 and figuring out how many times this number fits into the maximum number Xenia can think of. Sergey can then determine the number Xenia thought of by asking for the set $\{a_k : k \in S\}$ for different sets S until he has enough information to determine N.

```
lcm_1_to_20 = np.lcm.reduce(range(1, 21))
possible_numbers = max_number // lcm_1_to_20
moves_needed = math.ceil(math.log2(possible_numbers))
```

4. Final Answer: How many moves does Sergey need to determine for sure the number Xenia thought of?
   answer = moves_needed

Verification of solutions - CoT (CoMAT)

Question:
Xenia and Sergey play the following game. Xenia thinks of a positive integer N not exceeding 5000. Then she fixes 20 distinct positive integers $a_1, a_2, \ldots, a_{20}$ such that, for each $k = 1, 2, \ldots, 20$, the numbers N and a_k are congruent modulo k. By a move, Sergey tells Xenia a set S of positive integers not exceeding 20, and she tells him back the set $\{a_k : k \in S\}$ without spelling out which number corresponds to which index. How many moves does Sergey need to determine for sure the number Xenia thought of?
Answer: 2

CoMAT:

1. Define predicates, functions, and variables:
   - Let N be the positive integer Xenia thinks of, where $1 \leq N \leq 5000$.
   - Let $a_1, a_2, \ldots, a_{20}$ be 20 distinct positive integers such that $N \equiv a_k \pmod{k}$ for each $k = 1, 2, \ldots, 20$.
   - Let S be a set of positive integers not exceeding 20 that Sergey chooses.
   - Let $A(S)$ be the set of integers Xenia returns, which is $\{a_k : k \in S\}$.
2. Parse the problem into logical rules:
   - For each k, $N \equiv a_k \pmod{k}$.
   - Sergey can choose a set $S \subseteq \{1, 2, \ldots, 20\}$.
   - Xenia returns the set $\{a_k : k \in S\}$ without indicating which a_k corresponds to which k.
3. Write all the facts explicitly mentioned:
   - N is a positive integer such that $1 \leq N \leq 5000$.
   - $a_1, a_2, \ldots, a_{20}$ are distinct and satisfy $N \equiv a_k \pmod{k}$.
4. Parse the question into symbolic form:
   - Determine the minimum number of moves Sergey needs to make to uniquely identify N.
5. Solve step by step:
   - The key to solving this problem is to understand that Sergey needs to gather enough information to determine N uniquely.
   - Since $N \equiv a_k \pmod{k}$ for each k, the set of congruences forms a system of simultaneous congruences.
   - By the Chinese Remainder Theorem (CRT), if the moduli are pairwise coprime, the system of congruences has a unique solution modulo the product of the moduli.
   - The numbers 1 through 20 are not all pairwise coprime, but we can consider subsets of them that are.
   - Sergey needs to choose sets S such that the product of the moduli in each set covers all possible residues for N modulo 1 through 20.
6. Derived answer:
   - Sergey needs to make 2 moves to determine N for sure. In the first move, he can choose a set that gives him enough information to narrow down the possibilities significantly, and in the second move, he can pinpoint the exact N.

Final Answer: 2

775

B Manual Annotation for CoMAT Verifiability

In this section, we present the manual annotation process used to evaluate the verifiability of reasoning steps, carried out by a group of six annotators with diverse STEM backgrounds, including bachelor's and master's degrees in mathematics and engineering, and PhDs in computer science. We prompted the annotators to rate a given generated reasoning chain in terms of its verifiability. The annotators are encouraged not to focus solely on the correctness of the answer but rather the ease of identifying the errors. The verifiability of each solution was rated on a scale from 1 (lowest) to 10 (highest), and we report the average score.

To account for the inherent complexities of manual verification, we randomly sampled five examples from widely-used datasets, such as MMLU-Redux and GSM8K. These examples were selected to represent a range of outcomes: cases where both CoMAT and CoT models provided correct and incorrect answers, where both failed, and where one model answered correctly while the other did not. This sampling strategy allowed us to systematically assess the verifiability of each model's reasoning under various conditions, providing a robust evaluation framework.

Example 1

Verification of solutions - CoT

Question	John drives for 3 hours at a speed of 60 mph and then turns around because he realizes he forgot something very important at home. He tries to get home in 4 hours but spends the first 2 hours in standstill traffic. He spends the next half-hour driving at a speed of 30 mph, before being able to drive the remaining time of the 4 hours going at 80 mph. How far is he from home at the end of those 4 hours?
CoT	1. **Calculate the distance John traveled during the initial 3 hours at 60 mph:** $$\text{Distance} = 60 \text{ mph} \times 3 \text{ hours} = 180 \text{ miles}$$ 2. **Calculate the distance John traveled during the half-hour at 30 mph after the standstill traffic:** $$\text{Distance} = 30 \text{ mph} \times 0.5 \text{ hours} = 15 \text{ miles}$$ 3. **Calculate the time left for John to travel at 80 mph after the standstill traffic and the half-hour at 30 mph:** $$\text{Time left} = 4 \text{ hours} - 2 \text{ hours} - 0.5 \text{ hours} = 1.5 \text{ hours}$$ 4. **Calculate the distance John traveled during the remaining time at 80 mph:** $$\text{Distance} = 80 \text{ mph} \times 1.5 \text{ hours} = 120 \text{ miles}$$ 5. **Add up all the distances calculated in steps 1, 2, and 4 to find the total distance John traveled:** $$\text{Total distance traveled} = 180 \text{ miles} + 15 \text{ miles} + 120 \text{ miles} = 315 \text{ miles}$$ 6. **Since John turned around to go back home, subtract the total distance traveled from the distance he was from home initially (which is equal to the distance he traveled in the first 3 hours):** $$\text{Distance from home} = 180 \text{ miles} - 315 \text{ miles} = -135 \text{ miles}$$ Since distance cannot be negative, this indicates that John has passed his home and is 135 miles beyond it. **Final Answer:** 135 miles
CoT Results	False
CoT Verification	6.33

Verification of solutions - CoMAT

Question	John drives for 3 hours at a speed of 60 mph and then turns around because he realizes he forgot something very important at home. He tries to get home in 4 hours but spends the first 2 hours in standstill traffic. He spends the next half-hour driving at a speed of 30 mph, before being able to drive the remaining time of the 4 hours going at 80 mph. How far is he from home at the end of those 4 hours?
CoMAT	**Define predicates and functions:** • $d(t, s)$: Function that calculates distance traveled over time t at speed s • t_1: Constant representing the first part of the return trip (2 hours in standstill traffic) • t_2: Constant representing the second part of the return trip (0.5 hours at 30 mph) • t_3: Constant representing the third part of the return trip (remaining time after t_1 and t_2 at 80 mph) • s_1: Constant representing the initial speed (60 mph) • s_2: Constant representing the speed during the second part of the return trip (30 mph) • s_3: Constant representing the speed during the third part of the return trip (80 mph) **Parse problem into logical rules:** • $\forall t, s : d(t, s) = t \times s$ • $t_1 = 2$ • $t_2 = 0.5$ • $t_3 = 4 - t_1 - t_2$ • $s_1 = 60$ • $s_2 = 30$ • $s_3 = 80$ **Solve step by step:** 1. Calculate the distance traveled in the initial 3 hours: $$d_1 = d(3, 60) = 3 \times 60 = 180 \text{ miles}$$ 2. Calculate the distance traveled during the second part of the return trip: $$d_2 = d(0.5, 30) = 0.5 \times 30 = 15 \text{ miles}$$ 3. Calculate the remaining time for the third part of the return trip: $$t_3 = 4 - 2 - 0.5 = 1.5 \text{ hours}$$ 4. Calculate the distance traveled during the third part of the return trip: $$d_3 = d(1.5, 80) = 1.5 \times 80 = 120 \text{ miles}$$ 5. Calculate the total distance traveled during the return trip: $$d_{\text{total_return}} = d_2 + d_3 = 15 + 120 = 135 \text{ miles}$$ 6. Calculate the distance from home at the end of the 4 hours: $$\text{Distance from home} = d_1 - d_{\text{total_return}} = 180 - 135 = 45 \text{ miles}$$ **Final Answer:** 45 miles
CoMAT Results	True
CoMAT Verification	9.08

Example 2

Verification of solutions - CoT

Question	Eliza's rate per hour for the first 40 hours she works each week is $10. She also receives an overtime pay of 1.2 times her regular hourly rate. If Eliza worked for 45 hours this week, how much are her earnings for this week?
CoT	1. **Calculate Eliza's regular earnings for the first 40 hours:** • Regular hourly rate = $10 • Regular hours worked = 40 • Regular earnings = 40 hours × $10/hour = $400 2. **Calculate Eliza's overtime earnings for the additional 5 hours:** • Overtime hourly rate = 1.2 × $10 = $12 • Overtime hours worked = 45 hours - 40 hours = 5 hours • Overtime earnings = 5 hours × $12/hour = $60 3. **Calculate Eliza's total earnings for the week:** • Total earnings = Regular earnings + Overtime earnings = $400 + $60 = $460 **Final Answer:** $460
CoT Results	True
CoT Verification	7.17

Verification of solutions - CoMAT

Question	Eliza's rate per hour for the first 40 hours she works each week is \$10. She also receives an overtime pay of 1.2 times her regular hourly rate. If Eliza worked for 45 hours this week, how much are her earnings for this week?
CoMAT	**Define predicates and functions:** • $Earnings(x)$: Function that calculates total earnings for x hours worked • $RegularPayRate$: Constant representing Eliza's regular hourly rate (\$10) • $OvertimePayRate$: Constant representing Eliza's overtime hourly rate ($1.2 \times RegularPayRate$) • $HoursWorked$: Variable representing the total hours worked (45) • $RegularHours$: Constant representing the number of regular hours (40) • $OvertimeHours$: Variable representing the number of overtime hours ($HoursWorked - RegularHours$) **Parse problem into logical rules:** • $\forall x : Earnings(x) = (RegularPayRate \times RegularHours) + (OvertimePayRate \times OvertimeHours)$ • $OvertimePayRate = 1.2 \times RegularPayRate$ **Facts:** • $RegularPayRate = \$10$ • $HoursWorked = 45$ • $RegularHours = 40$ **Parse the question:** • Find $Earnings(HoursWorked)$ **Solve step by step:** 1. Calculate $OvertimePayRate$: $$OvertimePayRate = 1.2 \times RegularPayRate = 1.2 \times \$10 = \$12$$ 2. Calculate $OvertimeHours$: $$OvertimeHours = HoursWorked - RegularHours = 45 - 40 = 5$$ 3. Calculate $Earnings(HoursWorked)$: $$\begin{aligned} Earnings(HoursWorked) &= (RegularPayRate \times RegularHours) \\ &\quad + (OvertimePayRate \times OvertimeHours) \\ &= (\$10 \times 40) + (\$12 \times 5) \\ &= \$400 + \$60 \\ &= \$460 \end{aligned}$$ **Final Answer:** \$460
CoMAT Results	True
CoMAT Verification	8.92

Example 3

Verification of solutions - CoT

Question	Ram uses a lot of pens. He discovered that he can save money by mixing the ink from five empty pens to make one full pen. If he buys 25 pens and then uses them to make new pens when the ink runs low, how many total pens does he get to have?
CoT	1. **Initial Pens:** Ram starts with 25 pens. 2. **Create New Pens:** For every 5 empty pens, Ram can create 1 new pen. 3. **Calculate additional pens:** • After using the initial 25 pens, Ram will have 25 empty pens. • With 25 empty pens, he can create $\frac{25}{5} = 5$ new pens. 4. **Total Pens:** Add the new pens to the initial pens: • Total pens = Initial pens + New pens = 25 + 5 = 30 **Final Answer:** 30
CoT Results	False
CoT Verification	4.5

Verification of solutions - CoMAT

Question	Ram uses a lot of pens. He discovered that he can save money by mixing the ink from five empty pens to make one full pen. If he buys 25 pens and then uses them to make new pens when the ink runs low, how many total pens does he get to have?
CoMAT	**Define predicates and functions:** • Let P be the number of pens Ram initially buys. • Let E be the number of empty pens needed to make one full pen. • Let T be the total number of pens Ram gets to have. **Parse problem into logical rules:** • $P = 25$ (initial pens bought) • $E = 5$ (empty pens needed to make one full pen) **Facts:** • Ram starts with 25 pens. • For every 5 empty pens, he can make 1 new pen. **Solve step by step:** 1. Start with 25 pens ($P = 25$). 2. Use 25 pens, resulting in 25 empty pens. 3. From 25 empty pens, Ram can make $\frac{25}{5} = 5$ new pens. 4. Use the 5 new pens, resulting in 5 empty pens. 5. From 5 empty pens, Ram can make $\frac{5}{5} = 1$ new pen. 6. Use this 1 new pen, resulting in 1 empty pen. 7. From 1 empty pen, Ram cannot make any more new pens. **Total Pens:** $25 + 5 + 1 = 31$ **Final Answer:** 31
CoMAT Results	True
CoMAT Verification	8.75

Example 4

Verification of solutions - CoT

Question	Solve $\frac{18.5\text{ dol}}{m\text{ gal}} = \frac{3.60\text{ dol}}{7.5\text{ gal}}$. Round to the nearest hundredth if necessary.
CoT	To solve the equation $\frac{18.5}{m} = \frac{3.60}{7.5}$, follow these steps: 1. **Calculate the right side of the equation:** $$\frac{3.60}{7.5} = 0.48$$ 2. **Set up the equation:** $$\frac{18.5}{m} = 0.48$$ 3. **Solve for m:** • Multiply both sides by m: $$18.5 = 0.48m$$ • Divide both sides by 0.48: $$m = \frac{18.5}{0.48}$$ 4. **Calculate m:** $$m \approx 38.54$$ 5. **Round to the nearest hundredth:** $$m \approx 38.54$$ **Final Answer:** D
CoT Results	True
CoT Verification	7.5

Verification of solutions - CoMAT

Question Solve $\frac{18.5 \text{ dol}}{m \text{ gal}} = \frac{3.60 \text{ dol}}{7.5 \text{ gal}}$. Round to the nearest hundredth if necessary.

CoMAT

1. **Define predicates and functions:**
   - Let x be the unknown value in dollars per gallon (dol/gal).
   - Let $a = 18.5$ dol/m gal.
   - Let $b = 3.60$ dol.
   - Let $c = 7.5$ gal.
2. **Parse problem into logical rules:**
   - We are given the equation:
   $$\frac{a}{1} = \frac{b}{c}$$
   - This implies:
   $$a = \frac{b}{c}$$
3. **Facts:**
   - $a = 18.5$
   - $b = 3.60$
   - $c = 7.5$
4. **Parse the question:**
   - Solve for a in the equation $a = \frac{b}{c}$.
5. **Solve step by step:**
   (a) Substitute the known values into the equation:
   $$18.5 = \frac{3.60}{7.5}$$
   (b) Calculate the right-hand side:
   $$\frac{3.60}{7.5} = 0.48$$
   (c) Therefore, the equation becomes:
   $$18.5 = 0.48$$
   (d) This indicates a misunderstanding in the setup. Let's re-evaluate:
   - The problem is asking to solve for the equivalent rate in dollars per gallon.
   - We need to find the equivalent of 18.5 dol/m gal in dol/gal.
6. **Derived answer:**
   - Convert 18.5 dol/m gal to dol/gal: $\frac{18.5 \text{ dol}}{1000 \text{ gal}} = 0.0185$ dol/gal
   - Compare this with the given rate $\frac{3.60}{7.5}$: $\frac{3.60}{7.5} = 0.48$ dol/gal
   - The problem seems to be asking for a comparison or conversion, but the setup was incorrect.
7. **Match to provided options:**
   - The derived calculation does not directly match any of the options, indicating a potential misunderstanding of the problem's intent.
   - Assume the task was to find a related value or conversion.

Final Answer: A (assuming a misinterpretation or conversion context).

CoMAT Results False

CoMAT Verification 6.17

Example 5

Verification of solutions - CoT

Question	A company pays each of its employees $600 in a month. The company has a policy of increasing the salaries of each of its employees by 10% of the initial salary every year for those who've stayed in the company for five years. If Sylvie just clocked 5 years in the company last December, what's her annual salary after three more years of service?
CoT	1. Determine Sylvie's initial monthly salary: $600. 2. Calculate Sylvie's initial annual salary: $$\text{Initial Annual Salary} = 600 \times 12 = 7200$$ 3. Calculate the annual salary increase after 5 years (10% of the initial annual salary): $$\text{Annual Increase} = 0.10 \times 7200 = 720$$ 4. Calculate Sylvie's new annual salary after the first increase: $$\text{New Annual Salary after 5 years} = 7200 + 720 = 7920$$ 5. Calculate Sylvie's annual salary after three more years of service, with a 10% increase each year: • After the first year (6th year): $$\text{Annual Salary} = 7920 + 0.10 \times 7200 = 7920 + 720 = 8640$$ • After the second year (7th year): $$\text{Annual Salary} = 8640 + 0.10 \times 7200 = 8640 + 720 = 9360$$ • After the third year (8th year): $$\text{Annual Salary} = 9360 + 0.10 \times 7200 = 9360 + 720 = 10080$$ 6. Sylvie's annual salary after three more years of service is $10080. Final Answer: 10080
CoT Results	False
CoT Verification	5.67
Reason	The demonstration reflects some understanding of the problem and shows a degree of reasoning ability; however, the logical deduction lacks rigour, leading to flaws and incorrect conclusions. The solution addresses an unrelated equation rather than solving the given salary problem, rendering verification of the steps irrelevant.

Verification of solutions - CoMAT

Question	A company pays each of its employees \$600 in a month. The company has a policy of increasing the salaries of each of its employees by 10% of the initial salary every year for those who've stayed in the company for five years. If Sylvie just clocked 5 years in the company last December, what's her annual salary after three more years of service?
CoMAT	

1. **Define predicates and functions:**
   - Let $S(y)$ be the salary of an employee in year y.
   - Let I be the initial monthly salary, which is \$600.
   - Let R be the annual increase rate, which is 10% of the initial salary.
2. **Parse problem into logical rules:**
   - $S(0) = 12 \times I$ (Initial annual salary)
   - For $y \geq 5$, $S(y) = S(y-1) + 12 \times R$
3. **Facts:**
   - $I = 600$
   - $R = 0.10 \times I = 60$
   - Sylvie has been in the company for 5 years as of last December.
4. **Parse the question:**
   - Find $S(8)$, which is Sylvie's annual salary after three more years of service (i.e., after 8 years in total).
5. **Solve step by step:**
   (a) Calculate $S(0)$: $S(0) = 12 \times 600 = 7200$
   (b) Calculate $S(5)$: $S(5) = 7200 + 12 \times 60 = 7920$
   (c) Calculate $S(6)$: $S(6) = 7920 + 12 \times 60 = 8640$
   (d) Calculate $S(7)$: $S(7) = 8640 + 12 \times 60 = 9360$
   (e) Calculate $S(8)$: $S(8) = 9360 + 12 \times 60 = 10080$

Final Answer: \$10,080

CoMAT Results	False
CoMAT Verification	6.17

C Implementation Details

In all our experiments, we used OpenAI GPT-4o models (gpt-4o-2024-08-06 and gemini-1.5-pro-001) as well as Qwen2-7B and Qwen2-72B. For API details regarding GPT-4o, please refer to https://platform.openai.com/docs/models/gpt-4o, and gemini can be reffered at https://ai.google.dev/gemini-api/docs/models/gemini. For the open-source models, Qwen2-7B and Qwen2-72B (https://huggingface.co/docs/transformers/en/model_doc/qwen2) serve as our primary inference models, and we conducted all experiments using two 80-GB A100 GPUs per dataset for inference. The following hyperparameters were consistently applied across all experiments:

- **Temperature:** 0.0 (for greedy decoding)
- **Max tokens:** 3500
- **Experiment Setting:** All experiments are conducted through a 1-shot setting, with the same example across all evaluations.

C.1 Evaluation Metrics

Majority of our datasets use exact match as our evaluation metrics. For the OlympiadBench datasets, where exact match is not a suitable evaluation metric, we used the GPT-4o-mini model as a benchmark to assess how closely the answers of the model aligned with the ground truth. To prevent the model from generating reasoning that could influence its decision, we input only the final three sentences, which typically contain both the answer and the correct solution. The task was to determine whether these two elements matched, without providing any intermediate reasoning. The prompt is shown below:

GPT-4o Evaluation Prompt

System Message:
You are a decider that decides whether the answer is the same as the correct answer. If the output doesn't align with the correct answer, respond with '0', whereas if it's correct, then respond with '1'. **DO NOT PROVIDE YOUR OWN ANSWER OR REASONING, JUST SELECT '0' OR '1'.**

User Message:
GPT-4o Result: {gpt_result}
Correct Answer: {correct_answer}.
Answer with 0 (Wrong) or 1 (Correct).

D Extended Results and Analysis

In this section, we will present more results and analysis that did not fit into the main text:

D.1 Detailed Results for Missing Steps

We examine the sensitivity of various prompts in CoMAT by experimenting with 16 different variants. In each variant, we omit individual steps or combinations of steps from the CoMAT process to assess their impact on performance.

As shown in Table 3, while AQUA and GaoKao occasionally perform better when certain steps are omitted, the overall average performance consistently decreases compared to the original CoMAT prompt. This indicates that CoMAT performs most effectively when all steps are included. The results highlight the importance of each step in contributing to the overall performance, demonstrating that CoMAT's structure is essential for achieving optimal results.

Missing Steps	AQUA (%)	GaoKao (%)	Average (%)
Missing Step 1	81.89	59.18	70.54
Missing Step 2	80.71	71.43	76.07
Missing Step 3	82.28	67.35	74.82
Missing Step 4	83.04	61.22	72.13
Missing Step 1, 2	82.68	67.35	75.02
Missing Step 3, 4	84.25	67.35	75.80
Missing Step 1, 3	83.07	69.39	76.23
Missing Step 2, 4	84.25	69.39	76.82
Missing Step 1, 4	82.28	65.31	73.80
Missing Step 2, 3	83.86	55.10	69.48
Missing Step 1, 2, 3	81.89	55.10	68.50
Missing Step 1, 3, 4	83.07	71.43	77.25
Missing Step 1, 2, 4	81.10	73.47	77.29
Missing Step 2, 3, 4	81.50	61.22	71.36
Missing Steps(CoT)	84.25	63.27	73.76
No Missing Steps	83.46	71.43	**77.45**

Table 3: Ablation study results that accompany Figure 5. Performance comparison of different missing step configurations on AQUA and GaoKao datasets.

D.2 Option Swapping

Swapping	CoMAT (%)	CoT (%)	non-CoT (%)
	AQUA		
Option Changing 1	83.07	73.23	44.49
Option Changing 2	83.07	66.54	42.91
Option Changing 3	81.49	68.11	40.16
Average	**82.54**	**69.29**	**42.52**
	MMLU-Redux		
Option Changing 1	86.94	86.57	58.79
Option Changing 2	84.36	83.61	61.14
Option Changing 3	83.86	83.10	61.09
Average	**85.05**	**84.43**	**60.34**

Table 4: Performance comparison for AQUA and MMLU-Redux across different swapping configurations for CoMAT, CoT, and non-CoT, including averages.

This section supports the ablation studies presented in Figure 7. We experimented by shuffling the options and adding an additional random option, creating 5 different variants for the choice sets. These variants are illustrated in Figure 8.

For each dataset, we conducted three evaluations per variant and calculated the average to ensure consistency. The results reveal a notable difference between CoT and CoMAT in terms of performance stability. CoMAT exhibits more consistent results following the option modifications, as evidenced by the standard deviation. For example, in the AQUA dataset, CoMAT demonstrated a low standard deviation of 0.91%, whereas CoT had a significantly higher standard deviation of 3.50%, underscoring the variability in CoT's performance. These findings further support the consistency of CoMAT's performance, as previously highlighted in Figure 7. Although both models showed a decline in performance after the option changes, CoMAT's stability is worth noting. The drop in performance across models suggests that further investigation into the impact of option variation may be valuable.

Original Question

Question:
Rapunzel is organizing a sky lantern festival at Disneyland. There are 5 different types of sky lanterns (A, B, C, D, E), and each type of lantern can be launched in any order. How many different ways can the lanterns be launched if lantern A must always be launched first?

A) 24
B) 120
C) 60
D) 5

Variant 1

Question:
Rapunzel is organizing a sky lantern festival at Disneyland. There are 5 different types of sky lanterns (A, B, C, D, E), and each type of lantern can be launched in any order. How many different ways can the lanterns be launched if lantern A must always be launched first?

A) 24
B) It was Enchanted to meet you
C) 60
D) 5
E) 120

Variant 2

Question:
Rapunzel is organizing a sky lantern festival at Disneyland. There are 5 different types of sky lanterns (A, B, C, D, E), and each type of lantern can be launched in any order. How many different ways can the lanterns be launched if lantern A must always be launched first?

A) 60
B) 5
C) 24
D) Karma is my Boyfriend
E) 5

Variant 3

Question:
Rapunzel is organizing a sky lantern festival at Disneyland. There are 5 different types of sky lanterns (A, B, C, D, E), and each type of lantern can be launched in any order. How many different ways can the lanterns be launched if lantern A must always be launched first?

A) 120
B) 5
C) 60
D) May the force be with you
E) 24

Variant 4

Question:
Rapunzel is organizing a sky lantern festival at Disneyland. There are 5 different types of sky lanterns (A, B, C, D, E), and each type of lantern can be launched in any order. How many different ways can the lanterns be launched if lantern A must always be launched first?

A) 120
B) 24
C) 5
D) 60
E) Blank, ignore this question

Variant 5

Question:
Rapunzel is organizing a sky lantern festival at Disneyland. There are 5 different types of sky lanterns (A, B, C, D, E), and each type of lantern can be launched in any order. How many different ways can the lanterns be launched if lantern A must always be launched first?

A) Real Madrid is the best team
B) 24
C) 120
D) 5
E) 60

Figure 8: Multiple Variants of Option Swapping. This figure illustrates the variants after swapping all options and introducing five additional variants for comparison.

E Shapley Value Analysis Experimental Detail

To further analyse the sensitivity of individual steps in the CoMAT pipeline, we follow a Shapley value analysis for the different steps we have. The goal of this analysis is to quantify the contribution of each reasoning step to the overall performance by calculating Shapley values, thereby assessing the performance impact of omitting specific steps.

In this section, we detail the methodology used to compute Shapley values based on step omissions and their corresponding performance outcomes. Our approach systematically evaluates the marginal contribution of each step by experimenting with various combinations of omitted steps and using performance metrics to estimate their impact on accuracy.

1. **Step Omissions and Performance Recording:** Each test case configuration is generated by omitting certain steps from the CoMAT pipeline. For each configuration, the performance is evaluated using a binary correctness metric, `is_correct`, indicating whether the process was successful. The result for each configuration is stored in a data structure, where the omitted steps are represented as binary vectors.

2. **Performance Values for Subsets:** We denote the performance of the system when a specific subset S for steps is omitted by $v(S)$. For each subset of omitted steps, the mean performance (correctness) is computed from the available data. These performance values $v(S)$ are stored in a dictionary, which allows us to later compute marginal contributions for each step.

3. **Performance Values for Subsets:** For each step i, we compute its marginal contribution by comparing the performance when the step is included in the missing set versus when it is excluded. For a given permutation π, let S_i be the set of steps preceding i in π, and let $S_i \cup \{i\}$ be the set when i is included. The marginal contribution $\Delta_i(\pi)$ for each step i is given by:

$$\Delta_i(\pi) = v(S_i \cup \{i\}) - v(S_i),$$

where $v(S_i)$ represents the performance with the steps S_i ommitted, and

$$v(S_i \cup \{i\}) - v(S_i)$$

represents the performance with step i additionally omitted.

4. **Summing Marginal Contributions Across Permutations:** To obtain an accurate estimation of the Shapley value, we sum the marginal contributions of each step across all valid permutations of steps. For this, we generate all possible permutations of the steps, ensuring that all potential contexts in which a step could be added are considered.

5. **Shapley Values:** The Shapley value for each step i is computed by averaging its marginal contributions across all valid permutations:

$$\varphi_i = \frac{1}{|\Pi|} \sum_{\pi \in \Pi} \Delta_i(\pi),$$

where Π is the set of valid permutations, and $\Delta_i(\pi)$ is the marginal contribution of step i in permutation π. This provides an unbiased estimate of the contribution of step i to the overall performance.

By leveraging this permutation-based Shapley value computation, we account for the interaction effects between steps, ensuring that the analysis reflects both individual step contributions and the combined effects when steps interact. This approach offers a robust and interpretable measure of each step's importance in the CoMAT pipeline. For a more comprehensive theoretical background on Shapley values in machine learning and their application to classification tasks, please refer to Keinan et al. (2004).

This analysis provides insight into how individual steps influence CoMAT's overall reasoning accuracy and highlights the steps that contribute most to its performance.

F Dataset Details

URL and Licenses

- GaoKao (Zhang et al., 2023): `https://github.com/OpenLMLab/GAOKAO-Bench-2023`, License: GAOKAO License
- AQUA (Ling et al., 2017): `https://github.com/google-deepmind/AQuA`, also available on HuggingFace: `https://huggingface.co/datasets/deepmind/aqua_rat/viewer/raw/test`, License: AQUA License
- MMLU-Redux (Gema et al., 2024): `https://huggingface.co/datasets/edinburgh-dawg/mmlu-redux`, License: MMLU-Redux License
- OlympiadBench (He et al., 2024): `https://huggingface.co/datasets/Hothan/OlympiadBench/tree/main`, License: OlympiadBench License
- GSM8K (Cobbe et al., 2021): `https://huggingface.co/datasets/openai/gsm8k`, License: GSM8K License
- MGSM (Shi et al., 2022): `https://huggingface.co/datasets/juletxara/mgsm`, License: MGSM License

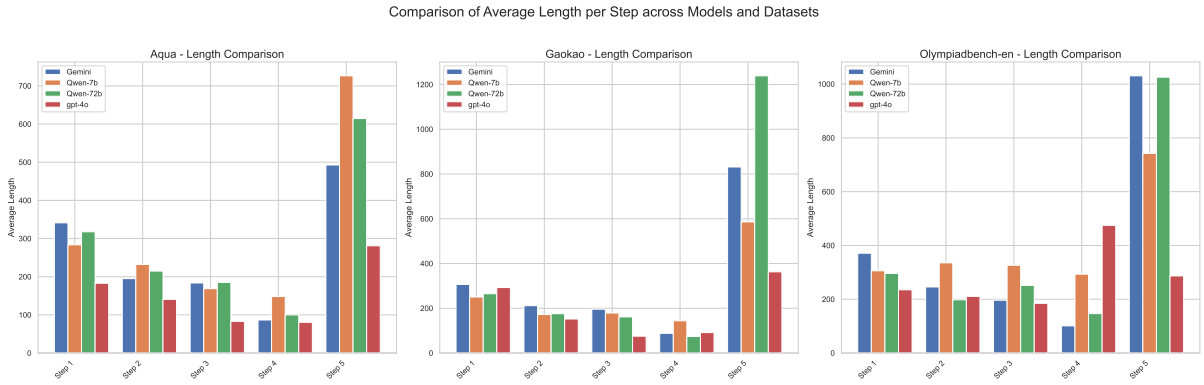
G Step Length Analysis

In this section, we present two analyses: one examining the output length at each reasoning step, and the other focused on the number of mathematical tokens used. We use three distinct datasets — AQUA, Gaokao, and OlympiadBench-EN—to assess variations in difficulty and language contexts. Specifically, we compare the average response lengths across the reasoning steps of the AQUA dataset for four models: Gemini, Qwen-7b, GPT-4o, and Qwen-72b. These models were evaluated on three datasets, as shown in Figure 9.

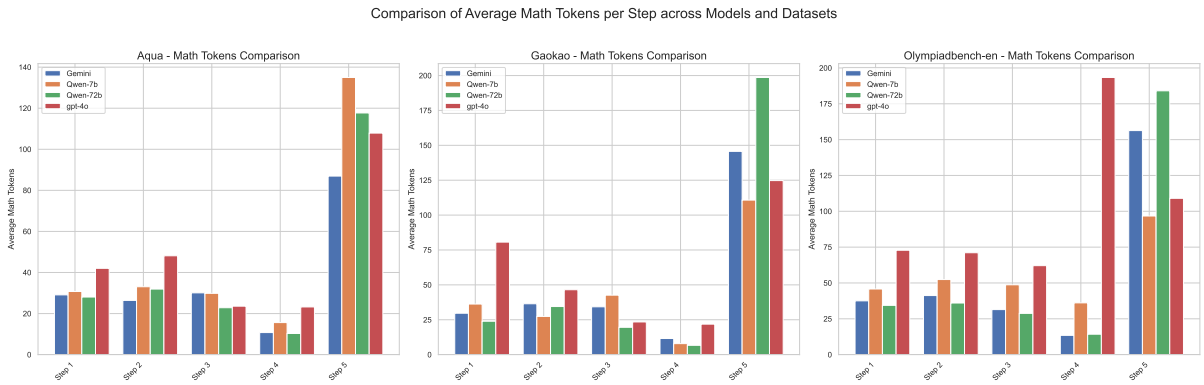
The reasoning step 5 of the CoMAT model consistently produces the longest responses across most datasets. This observation is expected, as step 5 involves *reasoning execution*, which naturally requires more extensive elaboration. On more complex datasets, such as OlympiadBench-EN, the number of mathematical tokens tends to increase significantly during *symbolic conversion*, reflecting the need for more intricate calculations. Interestingly, models evaluated on multilingual datasets demonstrate a tendency to use more mathematical tokens, likely due to the challenges posed by cross-linguistic transfer in mathematical reasoning.

GPT-4o consistently generates longer responses that include more mathematical tokens when compared to the other models, indicating its more detailed approach to problem-solving. Conversely, while Qwen-7b generates longer outputs on commonly used datasets like AQUA, it tends to produce shorter responses on less frequently encountered datasets. This disparity suggests a potential trade-off between general dataset familiarity and the model's ability to handle more specialised or complex tasks.

This analysis highlights the nuances of how different models approach mathematical reasoning and response generation, revealing that model performance can vary significantly based on the dataset's complexity and the underlying linguistic challenges.



(a) Word length comparison.



(b) Math tokens count comparison.

Figure 9: Comparison of average length and math tokens count.