

MathTrap300: A Benchmark for Recognizing Mathematical Insolvability in LLMs

Huaibo Chen^{1*}, Yixiao Lin^{2*}, Pengcheng Chen^{3†}, Zihan Zhao^{4†}, Nuohao Liu^{5†}, Yue Hu⁶, Qian Xie², Qinbo Bai⁷, Ning Yan⁸, Masood S. Mortazavi⁸, Kamal Youcef-Toumi¹

¹Massachusetts Institute of Technology, ²Cornell University, ³University of Washington, ⁴University of California San Diego

⁵University of Wisconsin-Madison, ⁶University of Southern California, ⁷Purdue University, ⁸Futurewei Technologies

Abstract

While modern large language models (LLMs) achieve high accuracy on many challenging mathematical benchmarks, they often struggle to recognize the insolvability of ill-posed problems. Existing benchmarks for insolvable problems are either modified from elementary-level math questions or lack rigorous validation of their insolvability. There is still no benchmark featuring inherently insolvable problems that require deep mathematical knowledge to identify. To address this gap, we introduce **MathTrap300**, the first benchmark consisting of 300 insolvable, ill-posed math problems with fundamental mathematical contradictions or missing conditions that demand deep domain knowledge to detect. In this work, we manually constructed these problems from well-posed counterparts through careful modifications and rigorous verification of ill-posedness by PhD-level experts. We then present a fine-grained, three-stage LLM judge framework designed based on observations of LLM responses to insolvable problems. This framework captures signals from both final answers and intermediate reasoning, providing richer metrics and enabling a more faithful assessment of insolvability recognition. Our evaluation of recent advanced LLMs on MathTrap300, combined with a detailed analysis of their response patterns, reveals a clear drop in accuracy from well-posed problems to their insolvable counterparts. Common failure modes include hallucination, guessing, and condition neglect. We also observe that even when models recognize insolvability, they still produce a definitive answer.

1 Introduction

Large language models (LLMs) have made substantial progress in mathematical reasoning, achieving high accuracy on widely used benchmarks such as MATH (Hendrycks et al. 2021), AMC, and AIME. Despite their strong performance on well-posed datasets, modern LLMs struggle to recognize insolvability in ill-posed problems. Most existing math benchmarks emphasize increasingly difficult, well-posed problems, while far fewer evaluate LLM performance on insolvable ones.

Several benchmarks for insolvable problems have been proposed (Zhao et al. 2024; Ma et al. 2025; Tian et al. 2024;

Xue et al. 2025; Sun et al. 2024b; Song, Shi, and Zhao 2025), but these are often derived from elementary-level math problems (Zhao et al. 2024; Tian et al. 2024; Ma et al. 2025; Sun et al. 2024b) or generated by LLMs without verified insolvability, sometimes containing only math-irrelevant artifacts (e.g., ambiguous wording or commonsense errors) (Song, Shi, and Zhao 2025). Moreover, because LLMs can exhibit complex behaviors when confronted with insolvable problems, response correctness cannot be reliably assessed by string matching (Xue et al. 2025), and single binary-judgment metrics (Ma et al. 2025; Zhao et al. 2024) fail to capture the diverse failure modes that arise.

To address these gaps, we introduce **MathTrap300**, a manually curated and double-verified dataset of 300 insolvable math problems with missing or contradictory conditions. A comparison with prior work is provided in Table 1. Some illustrative examples of problematic questions from existing datasets are provided in Appendix A1. Two sample problems of MathTrap300 are shown in Figure 1.

Specifically, our contributions are: (1) **MathTrap300**, a dataset of 300 insolvable problems crafted from challenging sources and rigorously double-verified by PhD-level experts; (2) a three-stage LLM judge pipeline, based on observed LLMs' response behaviors, that evaluates final-answer correctness, midway identification, and problem modification; and (3) a large-scale benchmark of 28 state-of-the-art LLMs on MathTrap300, together with a detailed analysis of their common failure patterns when facing insolvable problems.

2 Methods

2.1 Dataset construction

Our insolvable problems are adapted from well-posed problems drawn from MATH, AIME 2025, AMC, the Chinese college entrance exam, and high school math competitions, plus additional problems created by the authors. We modify these into either contradictory questions, where the given conditions cannot all hold, or missing-condition questions, where the information is insufficient to determine the desired values, explicitly avoiding insolvability from common-sense errors or ambiguous wording. We also incorporate 37 problems from MathTrap (Zhao et al. 2024), which, although relatively easy, preserve discriminative power, and retain only

*Equal contribution.

†Equal contribution.

CONTRADICTION

Original Problem

$f(2x)$ is symmetric about the origin and $f(x+1) + f(3-x) = 0$. For $x \in (2, 4)$: $f(x) = -\log_{1/2}(x-2) + m$. If $f(2022) - 1/2 = f(-1)$, find m .

Insolvable Problem

$f(2x)$ is symmetric about the origin and $f(x+1) + f(3-x) = 0$. For $x \in (2, 4)$: $f(x) = -\log_{1/2}(x-2) + m$ with $m > 0$. If $f(2022 + \epsilon) - 1/2 = f(-1)$ for $\epsilon \in (0, 1)$, find the smallest m .

Annotation: By symmetry about the origin: $f(-x) = -f(x)$. From $f(x+1) + f(3-x) = 0$ we get $f(x+4) = f(x)$. For $\epsilon \in (0, 1)$, $f(2022 + \epsilon) = f(2 + \epsilon) = \log_2 \epsilon + m$. Since $f(-1) = m$, the equation becomes $(\log_2 \epsilon + m - 1)/2 = m$, giving $\log_2 \epsilon = m + 1$. But $\log_2 \epsilon < 0$ for $\epsilon \in (0, 1)$, so $m < -1$, contradicting $m > 0$.

Model Response (GPT-o4 mini)

The condition $(f(2022 + \epsilon) - 1/2 = f(-1))$ becomes $(\log_2 \epsilon + m - 1)/2 = -m$. Solving gives $\log_2 \epsilon = 1 - 3m$, so $\epsilon = 2^{1-3m}$. Since $0 < \epsilon < 1$, we need $0 < 2^{1-3m} < 1$, then $m > 1/3$. Thus the smallest value is 1/3 X.

MISSING CONDITION

Original Problem

Seven variables a_1, a_2, \dots, a_7 satisfy:

$$a_1 + 4a_2 + 9a_3 + 16a_4 + 25a_5 + 36a_6 + 49a_7 = 1$$

$$4a_1 + 9a_2 + 16a_3 + 25a_4 + 36a_5 + 49a_6 + 64a_7 = 12$$

$$9a_1 + 16a_2 + 25a_3 + 36a_4 + 49a_5 + 64a_6 + 81a_7 = 123$$

Find: $16a_1 + 25a_2 + 36a_3 + 49a_4 + 64a_5 + 81a_6 + 100a_7$

Insolvable Problem

Same conditions, but to find:

$$16a_1 + 25a_2 + 36a_3 + 49a_4 + 64a_5 + 81a_6 + 121a_7$$

Annotation: [16, 25, 36, 49, 64, 81, 121] does not lie in the row space of the coefficient matrix, so this expression cannot be uniquely determined.

Model Response (Claude Sonnet 4)

Each equation can be written as $f(t) = a_1 t^2 + a_2 (t+1)^2 + \dots + a_7 (t+6)^2$. Since $f(t)$ is quadratic, write $f(t) = At^2 + Bt + C$. $f(1) = 1, f(2) = 12, f(3) = 123$. Then $A = 50, \dots, C = 90$. Therefore $f(4) = 334$ X.

Figure 1: A demonstration of insolvable problems and model responses. **Left: Contradiction** — adding a variable ϵ makes any positive m impossible. **Right: Missing Condition** — changing the target coefficient from 100 to 121 leaves the problem underdetermined. In both cases LLMs fail to recognize insolvability and instead force final answers.

problems that can mislead Hunyuan (Sun et al. 2024a). Each problem is proofread by a PhD-level expert distinct from its creator, solved by o4-mini (OpenAI 2025f) with high reasoning effort, and accompanied by an annotation explaining why it is insolvable; problems not identified as insolvable by o4-mini undergo in-depth manual review.

2.2 Evaluation Methods

For original problems, model outputs are compared directly with ground-truth answers. We first apply the symbolic comparison library from (Ye et al. 2025); unmatched outputs are further checked by Kimi-K2 (Kimi Team et al. 2025) (prompt in Appendix F).

For insolvable problems, we extend prior single-judge methods (Zhao et al. 2024; Ma et al. 2025; Sun et al. 2024b), which often overlook the diverse ways LLMs express insolvability awareness. From observed responses, we treat two behaviors as valid: (1) explicitly declaring insolvability, and (2) recognizing insolvability while consistently modifying the problem. Figure A3 illustrates valid Contradiction and Missing Condition cases, motivating our three-stage LLM judge pipeline inspired by (Sheng et al. 2025), which integrates these behaviors into a unified evaluation. Concretely, we use three complementary judges, all instantiated with Grok-4-fast-reasoning (xAI 2025c), to capture different insolvability signals.

J_1 - Final-answer judge: checks for LLMs’ insolvability claims in the final answer (e.g., “no solution,” “does not exist,” “not enough information,” etc.)

J_2 - Problem-modification judge: checks whether the LLM modifies the problem to make it solvable (e.g., adding a plausible missing constraint or fixing a variable domain).

J_3 - Midway-identification judge: flags if LLMs note insolvability in intermediate reasoning.

The judges are invoked sequentially: J_1 first examines the last 100 characters of the response. If true, the process stops and midway-identification is set to true. Otherwise, J_2 is called. If J_2 returns true, the process again stops and midway-identification is set to true. If J_2 is false, J_3 is then called. Overall correctness is true if either J_1 or J_2 succeeds. J_3 is reported separately as a diagnostic metric to quantify alignment between intermediate recognition and final/modification behavior.

We tune prompts on 84 responses from four representative models (Qwen3-30B-A3B-Instruct (Qwen Team 2025c), DeepSeek-V3.1-Think (DeepSeek-AI 2024), Kimi-K2 (Kimi Team et al. 2025), and o4-mini (OpenAI 2025f)), then validate on 96 responses from 24 other models. Against human labels, the F1-scores are 1.0 (final-answer), 0.94 (problem-modification), 0.76 (midway-identification), and 0.84 overall (details in figure A4 and table A1, prompts in Appendix G).

Method	Difficulty	Verified, Mathematical Insolvability	Fair Judge	Pattern Analysis
MathTrap (Zhao et al. 2024)	✗	○	○	✗
UMWP (Sun et al. 2024b)	✗	✗	○	✗
SUM (Song, Shi, and Zhao 2025)	✓	✗	✗	✗
PMC (Tian et al. 2024)	✗	✗	✗	✗
UMP (Ma et al. 2025)	✗	✗	○	○
ReliableMath (Xue et al. 2025)	✓	✗	✗	○
MathTrap300 (Ours)	✓	✓	✓	✓

Table 1: **Comparison with previous works.** Legend: ✓ good, ○ limited, ✗ absent. Verified, Math. Insolv. indicates problems rigorously are verified as mathematically insolvable (as opposed to commonsense gaps or wording issues).

3 Experimental Results

3.1 Benchmarking

We evaluated 28 modern LLMs on MathTrap300 in a zero-shot setting with the instruction: “You are a math expert. Please reason step by step and show your step-by-step reasoning first, and then put your final answer within “\boxed{}”,” using the generation parameters in Table A2. On non-empty responses, we report three metrics in Table 2: final answer accuracy (insolvability declared in the final response), identification accuracy (insolvability flagged during reasoning), and overall accuracy (problem recognized as insolvable if J_1 or J_2 succeeds in section 2.2).

From the benchmarking results, we find that most models struggle to recognize traps, with accuracies on insolvable problems notably lower than on original ones. Strong reasoning models, such as **GPT-5** and **DeepSeek-V3.1-Think**, show drops of only about 6-7%, while chat models such as **GPT-4o** or **Llama4-Scout-Instruct** lose more than 33-39%. The true drop in accuracy may be larger since our LLM judge tends to generate false positives more than false negatives (see Table A1). The benchmark reveals several recurring schemes, and below we summarize them as two main observations.

First, there is a pronounced **identification-declaration gap**. Across the evaluated set, the average identification accuracy (models flagging the insolvability during reasoning) is 73.98% while the average final-answer (explicit declaration in the final boxed answer) accuracy is only 14.74%. The resulting gap (about 59.2%) indicates that models frequently notice anomalies while composing intermediate steps but still prefer to output a definite final answer rather than explicitly abstain or state “no solution”. Consequently, evaluations relying exclusively on final answers risk substantially underestimating a model’s internal detection of inconsistencies.

Second, performance clusters by model family and by the presence of extended reasoning traces. Reasoning-oriented variants (rows labelled “Thinking” or “with thinking”) tend to achieve substantially higher identification and overall accuracies than their chat-oriented counterparts. For example, several proprietary reasoning models reach overall accuracies in the high 70s-80s (see GPT-5 and Gemini variants in Table 2), while many chat-first models lag behind. This suggests that exposing and encouraging long-form internal reasoning materially improves a model’s ability to surface

insolvability signals.

3.2 Failure Pattern Analysis

Section 2.2 introduces two cases of correctness, direct declaration in the final answer and problem modification. Now we present the common failure modes we observed, with examples in Figure 2 and annotations in Figure A5.

Case 1: Guessing	Model: Qwen3-30B-A3B-Instruct
Insolvable Problem	
<i>Missing Condition</i> Consider real numbers x, y, z, u, v, w such that the determinant of the matrix with (u, y, z) in first row, (x, v, z) in second row, and (x, y, w) in third row is 1. Given that $x \neq u, y \neq v, z \neq w$, determine the value of $u/(u - x) + v/(v - y) + w/(w - z)$.	
Response	
Let the desired sum be S ... In this case, $S = 3$, in other case, $S = -0.5$, so unless the problem has additional constraints, the expression is not unique... Given the time I spent, I think the answer is 3. So I'll go with 3	
<i>It was aware that the expression is not unique but was uncertain about its decision and ultimately just picked a possible value</i>	
Case 2: Hallucination	Model: o4-mini
Insolvable Problem	
<i>Missing Condition</i> $h(k(x)) = x^2$ and $k(h(x)) = x^3$ for $x \geq 1$. Given $k(16) = 16$, find $[k(8)]^2$.	
Response	
Define $X = \ln x$, $K(X) = \log(k(e^X))$. Apply doubling-of-log trick to find $k(8)$. From functional equation $K(2Y) = 3K(Y)$, solve for $K(\ln 8)$ given $K(\ln 4)$. Since $\ln 8 = 2 \cdot \ln 4$...	
<i>Actually it is $\ln 16 = 2 \cdot \ln 4$ since $4^2 = 16$</i>	
Case 3: Condition Neglect	Model: deepSeek-V3.1-Think
Insolvable Problem	
<i>Contradiction</i> Smallest positive period of $f(x) = 2\tan x / (1 - k\tan^2 x)$ is $\pi/2$. Find positive k .	
Response	
$f(x + \pi/2) = f(x)$. We have: $2\tan x / (1 - k\tan^2 x) = 2\tan x / (k - \tan^2 x)$. Cancel $2\tan x$: $k - \tan^2 x = 1 - k\tan^2 x$. Rewriting: $(k - 1)(1 + \tan^2 x) = 0$. Since $1 + \tan^2 x \neq 0$, we get $k = 1$	
<i>When $k = 1$, domain issues ignored (e.g. $x = 0$ is valid but $0 + \pi/2$ not)</i>	

Figure 2: Sample problems for Failure Pattern Analysis

(1) Guessing. In this case, the model shows signs of detecting contradictions or anomalies and then attempts different methods or values. After realizing that none of them work, it nevertheless produces a definitive final answer without justification (see Figure 2, Case 1). This behavior likely

Table 2: Results of benchmarking. Accuracy drop is the original accuracy minus overall accuracy of the insolvable problems.

Model	Insolvable problem accuracy			Original Acc	Acc. Drop
	Final Answer Acc	Identification Acc	Overall Acc		
<i>Open Chat Models</i>					
DeepSeek-V3.1	9.33	84.00	80.67	84.67	4.00
Kimi-K2-Instruct-0905	19.33	70.33	65.00	82.00	17.00
Llama3.3-70B-Instruct	1.33	33.00	28.67	67.33	38.67
Llama4-Maverick-Instruct	4.67	53.33	46.67	77.00	30.33
Llama4-Scout-Instruct	1.67	44.00	37.33	76.67	39.33
Qwen2.5-72B-Instruct	3.00	38.67	34.33	67.67	33.33
Qwen3-235B-A22B-Instruct-2507	8.67	93.67	90.33	88.00	-2.33
Qwen3-30B-A3B-Instruct-2507	7.33	90.33	87.33	86.33	-1.00
<i>Open-source Reasoning LLMs</i>					
DeepSeek-V3.1-Think	9.33	96.33	88.00	94.67	6.67
GPT-oss-120b	26.67	92.33	90.33	87.67	-2.67
GPT-oss-20b	30.00	93.67	89.00	87.33	-1.67
Phi-4-reasoning-plus	4.00	32.33	28.67	64.33	35.67
Qwen3-235B-A22B-Thinking-2507	6.33	93.33	79.00	91.00	12.00
Qwen3-30B-A3B-Thinking-2507	8.67	94.67	90.67	92.00	1.33
Qwen3-8B (Thinking)	5.33	95.33	87.67	80.67	-7.00
Qwen2.5-Math-72B-Instruct	3.67	39.33	35.67	70.00	34.33
<i>Proprietary Chat LLMs</i>					
Gemini 2.5 Flash (no thinking)	23.00	89.67	83.67	89.33	5.67
Claude Sonnet 4 (no thinking)	6.00	69.00	64.00	78.67	14.67
GPT-4o-2024-11-20	18.00	28.33	27.00	61.67	34.67
GPT-4.1-2025-04-14	24.33	75.67	72.67	76.67	4.00
<i>Proprietary Reasoning LLMs</i>					
Claude Sonnet 4 (extended thinking)	11.67	90.00	83.67	86.33	2.67
Gemini 2.5 Flash (with thinking)	27.00	93.67	88.33	87.33	-1.00
Gemini 2.5 Pro	20.00	91.67	85.33	91.33	6.00
Grok-4	19.05	70.13	68.40	95.89	27.49
Grok-3-mini-beta	6.00	91.67	81.67	88.67	7.00
o4-mini-2025-04-16	30.00	64.33	63.00	90.00	27.00
GPT-5-2025-08-07	45.15	85.28	83.61	90.67	7.05
Average Acc	14.74	73.98	69.12	83.08	13.96

reflects an inherent bias toward providing a definite answer rather than admitting impossibility.

(2) Hallucination. In this case, the model produces intermediate steps with spurious claims to patch missing conditions or reconcile contradictions. Typical patterns include fabricated identities, arbitrary substitutions, or even basic arithmetic errors, leading to meaningless final answer (see Figure 2 Case 2).

(3) Condition Neglect. In this failure mode, the model follows an apparently rigorous reasoning path but overlooks an implicit condition, producing an answer that violates it (see Figure 2, Case 3). This pattern often arises in problems where even humans must carefully verify solutions to avoid mistakes, such as checking domain requirements for periodic functions in Case 3 of Figure 2.

We also observe that LLMs, especially chat models, tend to use brute-force enumeration rather than concise analytic reasoning, consuming more tokens than for well-posed questions (see Figure A6). Taken together, these patterns

highlight important engineering goals: to encourage explicit solvability checks in prompting, to enable abstention mechanisms, and to prefer reasoning modes that both reveal and report inconsistencies.

4 Conclusion

We introduced **MathTrap300**, which includes 300 rigorously verified insoluble problems and a three-stage judge (final-answer, modification, midway identification). Our evaluation of 28 recent LLMs shows that many models notice inconsistencies during reasoning but rarely state them clearly in the final answer, revealing an identification-declaration gap. We also identified common failure patterns: guessing, hallucination, and condition neglect, which explain why models often force incorrect solutions instead of acknowledging insolvability. We hope MathTrap300 supports the development of models that solve problems when possible and explicitly acknowledge insolvability.

References

Anthropic. 2025. Claude-4-Sonnet. <https://www.anthropic.com/news/clause-4>. Accessed: 2025-10-23.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; Hesse, C.; and Schulman, J. 2021. Training Verifiers to Solve Math Word Problems. *arXiv preprint*.

DeepSeek-AI. 2024. DeepSeek-V3.1. <https://api-docs.deepseek.com/news/news250821>. Accessed: 2025-10-23.

Google DeepMind. 2025a. Gemini 2.5 Flash. <https://cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/2-5-flash>. Accessed: 2025-10-23.

Google DeepMind. 2025b. Gemini 2.5 Pro. <https://cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/2-5-pro>. Accessed: 2025-10-23.

He, C.; Luo, R.; Bai, Y.; Hu, S.; Thai, Z. L.; Shen, J.; Hu, J.; Han, X.; Huang, Y.; Zhang, Y.; et al. 2024. Olympiad-bench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*.

Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; and Steinhardt, J. 2021. Measuring Mathematical Problem Solving With the MATH Dataset. In *NeurIPS Datasets and Benchmarks Track*.

Huang, K.; Guo, J.; Li, Z.; Ji, X.; Ge, J.; Li, W.; Guo, Y.; Cai, T.; Yuan, H.; Wang, R.; Wu, Y.; Yin, M.; Tang, S.; Huang, Y.; Jin, C.; Chen, X.; Zhang, C.; and Wang, M. 2025. MATH-Perturb: Benchmarking LLMs' Math Reasoning Abilities against Hard Perturbations. In *Proceedings of the 42nd International Conference on Machine Learning (ICML 2025) – Poster*.

Kimi Team; Bai, Y.; Bao, Y.; Chen, G.; Chen, J.; Chen, N.; Chen, R.; Chen, Y.; Chen, Y.; Chen, Y.; et al. 2025. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*.

Lewkowycz, A.; Andreassen, A.; Dohan, D.; Dyer, E.; Michalewski, H.; Ramasesh, V.; Slone, A.; Anil, C.; Schlag, I.; Gutman-Solo, T.; et al. 2022. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35: 3843–3857.

Ma, J.; Dai, D.; Yuan, Z.; li, R.; Luo, W.; Wang, B.; Liu, Q.; Sha, L.; and Sui, Z. 2025. Large Language Models Struggle with Unreasonability in Math Problems. *arXiv:2403.19346*.

Meta AI. 2024. Llama-3.3-70B-Instruct. <https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct>. Accessed: 2025-10-23.

Meta AI. 2025a. Llama-4-Maverick-17B-128E-Instruct. <https://huggingface.co/meta-llama/Llama-4-Maverick-17B-128E-Instruct>. Accessed: 2025-10-23.

Meta AI. 2025b. Llama-4-Scout-17B-16E-Instruct. <https://huggingface.co/meta-llama/Llama-4-Scout-17B-16E-Instruct>. Accessed: 2025-10-23.

Microsoft. 2025. Phi-4-reasoning-plus. <https://huggingface.co/microsoft/Phi-4-reasoning-plus>. Accessed: 2025-10-23.

Mirzadeh, I.; Alizadeh, K.; Shahrokhi, H.; Tuzel, O.; Bengio, S.; and Farajtabar, M. 2024. GSM-Symbolic: Understanding the Limitations of Mathematical Reasoning in Large Language Models.

OpenAI. 2024. GPT-4o. <https://platform.openai.com/docs/models/gpt-4o>. Accessed: 2025-10-23.

OpenAI. 2025a. GPT-4.1. <https://platform.openai.com/docs/models/gpt-4.1>. Accessed: 2025-10-23.

OpenAI. 2025b. GPT-5. <https://platform.openai.com/docs/models/gpt-5>. Accessed: 2025-10-23.

OpenAI. 2025c. gpt-oss-120b. <https://platform.openai.com/docs/models/gpt-oss-120b>. Accessed: 2025-10-23.

OpenAI. 2025d. gpt-oss-20b. <https://platform.openai.com/docs/models/gpt-oss-20b>. Accessed: 2025-10-23.

OpenAI. 2025e. OpenAI o3. <https://platform.openai.com/docs/models/o3>. Accessed: 2025-10-23.

OpenAI. 2025f. OpenAI o4-mini. <https://platform.openai.com/docs/models/o4-mini>. Accessed: 2025-10-23.

Qwen Team. 2024a. Qwen2.5-72B-Instruct. <https://huggingface.co/Qwen/Qwen2.5-72B-Instruct>. Accessed: 2025-10-23.

Qwen Team. 2024b. Qwen2.5-Math-72B-Instruct. <https://huggingface.co/Qwen/Qwen2.5-Math-72B-Instruct>. Accessed: 2025-10-23.

Qwen Team. 2025a. Qwen3-235B-A22B-Instruct-2507. <https://huggingface.co/Qwen/Qwen3-235B-A22B-Instruct-2507>. Accessed: 2025-10-23.

Qwen Team. 2025b. Qwen3-235B-A22B-Thinking-2507. <https://huggingface.co/Qwen/Qwen3-235B-A22B-Thinking-2507>. Accessed: 2025-10-23.

Qwen Team. 2025c. Qwen3-30B-A3B-Instruct-2507. <https://huggingface.co/Qwen/Qwen3-30B-A3B-Instruct-2507>. Accessed: 2025-10-23.

Qwen Team. 2025d. Qwen3-30B-A3B-Thinking-2507. <https://huggingface.co/Qwen/Qwen3-30B-A3B-Thinking-2507>. Accessed: 2025-10-23.

Qwen Team. 2025e. Qwen3-8B. <https://huggingface.co/Qwen/Qwen3-8B>. Accessed: 2025-10-23.

Shah, V.; Yu, D.; Lyu, K.; Park, S.; Yu, J.; He, Y.; Ke, N. R.; Mozer, M.; Bengio, Y.; Arora, S.; et al. 2024. Ai-assisted generation of difficult math questions. *arXiv preprint arXiv:2407.21009*.

Sheng, J.; Lyu, L.; Jin, J.; Xia, T.; Gu, A.; Zou, J.; and Lu, P. 2025. Solving Inequality Proofs with Large Language Models. *arXiv preprint arXiv:2506.07927*. Accepted to NeurIPS 2025 (Spotlight).

Shi, F.; Chen, X.; Misra, K.; Scales, N.; Dohan, D.; Chi, E. H.; Schärli, N.; and Zhou, D. 2023. Large Language Models Can Be Easily Distracted by Irrelevant Context. In *arXiv preprint arXiv:2302.00093*.

Song, L.; Shi, T.; and Zhao, J. 2025. The hallucination tax of reinforcement finetuning. *arXiv preprint arXiv:2505.13988*.

Srivastava, S.; B, A. M.; V, A. P.; Menon, S.; Sukumar, A.; T, A. S.; Philipose, A.; Prince, S.; and Thomas, S. 2024.

Functional Benchmarks for Robust Evaluation of Reasoning Performance, and the Reasoning Gap. *arXiv preprint arXiv:2402.19450*.

Sun, X.; Chen, Y.; Huang, Y.; Xie, R.; Zhu, J.; Zhang, K.; Li, S.; Yang, Z.; Han, J.; Shu, X.; et al. 2024a. Hunyuanlarge: An open-source moe model with 52 billion activated parameters by tencent. *arXiv preprint arXiv:2411.02265*.

Sun, Y.; Yin, Z.; Guo, Q.; Wu, J.; Qiu, X.; and Zhao, H. 2024b. Benchmarking Hallucination in Large Language Models Based on Unanswerable Math Word Problem. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, 2178–2188.

Tian, S.-Y.; Zhou, Z.; Yu, K.-Y.; Yang, M.; Jia, L.-H.; Guo, L.-Z.; and Li, Y.-F. 2024. VC Search: Bridging the Gap Between Well-Defined and Ill-Defined Problems in Mathematical Reasoning. *arXiv:2406.05055*.

xAI. 2025a. Grok-3-mini-beta. <https://x.ai/news/grok-3>. Accessed: 2025-10-23.

xAI. 2025b. Grok-4. <https://x.ai/news/grok-4>. Accessed: 2025-10-23.

xAI. 2025c. grok-4-fast-reasoning. <https://x.ai/news/grok-4-fast>. Accessed: 2025-10-23.

Xue, B.; Zhu, Q.; Wang, R.; Wang, S.; Wang, H.; Mi, F.; Wang, Y.; Shang, L.; Liu, Q.; and Wong, K.-F. 2025. ReliableMath: Benchmark of Reliable Mathematical Reasoning on Large Language Models. *arXiv:2507.03133*.

Ye, Y.; Huang, Z.; Xiao, Y.; Chern, E.; Xia, S.; and Liu, P. 2025. LIMO: Less is More for Reasoning. In *Second Conference on Language Modeling*.

Zhao, J.; Tong, J.; Mou, Y.; Zhang, M.; Zhang, Q.; and Huang, X. 2024. Exploring the Compositional Deficiency of Large Language Models in Mathematical Reasoning Through Trap Problems. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 16361–16376. Miami, Florida, USA: Association for Computational Linguistics.

A Related Works

A.1 Benchmarks for Math Reasoning

To evaluate LLMs’ reasoning capability, numerous math reasoning benchmarks have been proposed. Representative ones include GSM8K (Cobbe et al. 2021), MATH (Hendrycks et al. 2021), Minerva (Lewkowycz et al. 2022), OlympiadBench (He et al. 2024), AMC, and AIME. As LLM reasoning improves, the community increasingly calls for higher-difficulty math benchmarks to avoid saturation. Indeed, some recently released models, such as Grok-4 and GPT-5, already exceed 90% accuracy on AIME.

A.2 Variances of Math Benchmarks

In addition to proposing harder math benchmarks, another line of work focuses on creating variants of existing benchmarks to assess data contamination, memorization, or robustness against irrelevant information. GSM-IC (Shi et al.

2023) introduces irrelevant descriptions into GSM8K to test a model’s sensitivity to distracting information, showing a significant performance drop. Functional MATH (Srivastava et al. 2024) modifies MATH problems into dynamic, functional benchmarks. MATH² (Shah et al. 2024) extracts the required skills from two random problems in MATH and combines them to generate a more challenging problem that requires compositional skills. Similarly, GSM-Symbolic (Mirzadeh et al. 2024) creates a symbolic template of GSM8K, where numeric values, names, etc., can be changed programmatically, and substantial data contamination is observed across a series of models. MATH-Perturb (Huang et al. 2025) introduces simple and hard perturbations into MATH through minimal editing, where the solution logic remains unchanged for the former but is fundamentally altered for the latter. Memorization is observed when models attempt to follow the paths of the original problems.

A.3 Benchmarks of Insolvable Math Problems

Research on insolvable problems forms another related direction. MathTrap (Zhao et al. 2024), UMP (Ma et al. 2025), and UMWP (Sun et al. 2024b) are pioneering works; however, their datasets primarily derive from elementary-level problems. Except for MathTrap with several mathematically insolvable problems, the other two datasets only have trap problems relying on commonsense gaps or ambiguous wording, offering limited mathematical depth. Moreover, their single LLM-based judge provides only binary outputs, restricting diagnostic granularity. PMC (Tian et al. 2024) scales insolvability generation by converting arithmetic problems into missing-condition or contradictory variants, but most remain trivial, and automatic generation often produces superficial insolvability, or even solvable cases. ReliableMath (Xue et al. 2025) is the first insolvable math benchmark to incorporate competition-level problems, but its automatic pipeline still suffers from “pseudo-insolvability”, where added or removed information does not truly affect solvability. In addition, its reliance on the mere appearance of “unsolvable” in final answers as the evaluation criterion underestimates model accuracy.

B Detailed Analysis of Prior Work Limitations

See Figures A1 and A2.

C Acceptable Model Behaviors

See Figure A3.

D Verification of LLM Judges

See Figure A4 and Table A1.

E Experimental Details

E.1 Experimental Setups

See Table A2. For closed-source models (e.g., Grok-4 and GPT-5), reasoning steps may be hidden, so they generate

Triangular Numbers Problem

ReliableMath

Original Problem

A triangular number is a positive integer that can be expressed in the form $t_n = 1+2+3+\dots+n$, for some positive integer n . The three smallest triangular numbers that are also perfect squares are $t_1 = 1 = 1^2$, $t_8 = 36 = 6^2$, $t_{49} = 1225 = 35^2$. What is the sum of the digits of the fourth smallest triangular number that is also a perfect square?

Modified Question

A triangular number is a positive integer (the definition is removed.) The three smallest triangular numbers that are also perfect squares are $t_1 = 1 = 1^2$, $t_8 = 36 = 6^2$, $t_{49} = 1225 = 35^2$. What is the sum of the digits of the fourth smallest triangular number that is also a perfect square?

Design Limitations

Removing the triangular number definition may not create genuine insolvability since LLMs have extensive training on mathematical concepts. The background knowledge compensates for the missing explicit definition.

Salary Distribution Problem

UMP

Original Problem

Zaid's \$6000 salary: 2/3 rent, **1/4** of rest donated, \$700 to daughter. What's left?

Modified Question

Zaid's \$6000 salary: 2/3 rent, **3/4** of rest donated, \$700 to daughter. What's left?

Design Limitations

The direct change from 1/4 to 3/4 creates an obvious mathematical constraint violation that may be too straightforward to effectively challenge or mislead advanced language models.

Figure A1: Typical questions from ReliableMath (Xue et al. 2025) and UMP (Ma et al. 2025)

Jelly Bean Counting Problem

PMC

Original Problem

Gunter is trying to count the jelly beans in a jar. He asks his friends how many they think are in the jar. One says **80**. Another says 20 more than half the first one. A third says 25% more than the first one. What is their average guess?

Modified Question

Type 1 (Missing Condition):

Gunter is trying to count the jelly beans in a jar. He asks his friends how many they think are in the jar. One says **a certain number**. Another says 20 more than half the first one. A third says 25% more than the first one. What is their average guess?

Type 2 (Contradiction):

Gunter is trying to count the jelly beans in a jar. He asks his friends how many they think are in the jar. One says **80**. Another says 20 more than half the first one. A third says 25% more than the first one. What is their average guess

if the second friend's guess is 50 ?

Design Limitations

Both humans and LLMs naturally follow conditional "if" statements and treat "certain number" as variable x . PMC's template-based approach for introducing missing conditions and contradictions is straightforward but may be too formulaic to challenge advanced models.

Figure A2: Typical questions from PMC (Tian et al. 2024)

CONTRADICTION

Insolvable Problem

$f(2x)$ is symmetric about the origin and $f(x+1) + f(3-x) = 0$. For $x \in (2,4)$: $f(x) = -\log_{1/2}(x-2) + m$. If $(f(2022+\varepsilon) - 1)/2 = f(-1)$ for $\varepsilon \in (0,1)$, find the smallest value of m .

✓ Correct Answer 1:

There is no positive m that satisfies the conditions.

[explicitly states the insolvability in the final answer]

✓ Correct Answer 2:

There seems to be an inconsistency, since no choice of $\varepsilon \in (0,1)$ exists for positive m [realizing the insolvability midway]. If the intended range of ε is $\varepsilon > 1$, then we get the minimum $m = 1$ [modifying the problem]

MISSING CONDITION

Insolvable Problem

Consider seven variables a_1, a_2, \dots, a_7 satisfying:

$$a_1 + 4a_2 + 9a_3 + 16a_4 + 25a_5 + 36a_6 + 49a_7 = 1$$

$$4a_1 + 9a_2 + 16a_3 + 25a_4 + 36a_5 + 49a_6 + 64a_7 = 12$$

$$9a_1 + 16a_2 + 25a_3 + 36a_4 + 49a_5 + 64a_6 + 81a_7 = 123$$

Find: $16a_1 + 25a_2 + 36a_3 + 49a_4 + 64a_5 + 81a_6 + 121a_7$

✓ Correct Answer 1:

The desired expression cannot be uniquely determined.

[explicitly states the insolvability in the final answer]

✓ Correct Answer 2:

$16a_1 + 25a_2 + \dots + 121a_7 = 334 + a_7$ leaves one free variable, so the value cannot be uniquely determined

[realizing the insolvability midway]. If we set $a_7 = 0$, then the problem becomes solvable, with the answer 334 [modifying the problem].

Figure A3: **Acceptable model behaviors.** For Contradiction problem, a correct response either explicitly states the inconsistency or proposes a natural correction. For Missing Condition, a correct response either states the underdetermination or adds a natural constraint to make the problem well-posed.

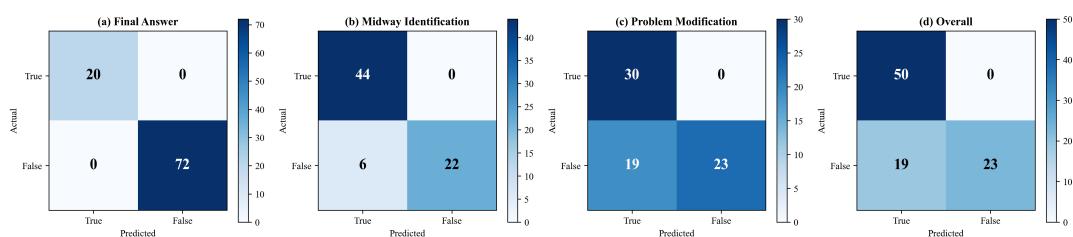


Figure A4: Confusion matrices for three judges and overall assessment

Model	Accuracy	Precision	Recall	F1-score
Final-answer	1.00	1.00	1.00	1.00
Identification	0.92	0.88	1.00	0.94
Modification	0.74	0.61	1.00	0.76
Overall	0.79	0.72	1.00	0.84

Table A1: Performance metrics for our judge pipeline with human label as ground truth.

empty outputs when the token limit is exceeded. Thus, accuracies are computed on non-empty responses only.

E.2 Failure Pattern Analysis Problem Annotation

See Figure A5.

E.3 Token Usage Ratio

See Figure A6.

F Prompts for equivalence check

See Figures A7, A8, A9.

G Prompts for Judge Pipelines

See Figures A10, A11, A12, A13, A14, A15, A16.

Insolvable Problem 1: Matrix Determinant

Insolvable Problem

Missing Condition Consider real numbers x, y, z, u, v, w such that the determinant of the matrix with elements (u, y, z) in first row, (x, v, z) in second row, and (x, y, w) in third row is 1. Given that $x \neq u, y \neq v, z \neq w$, determine the value of $u/(u - x) + v/(v - y) + w/(w - z)$.

Annotation

We are given that the determinant equals $uvw - uyz - vxz - wxy + 2xyz = 1$. We fix $x = 0, y = 0, z = 1$. Then the determinant reduces to $uvw = 1$. The desired sum becomes $u/(u - 0) + v/(v - 0) + w/(w - 1) = u + v + w/(w - 1) = 3 + 1/(w - 1)$. You can choose any w other than 0 and 1 to change the value of the desired formula, and you always can find valid u, v that can satisfy the given constraints.

Insolvable Problem 2: Functional Equations

Insolvable Problem

Missing Condition $h(x)$ and $k(x)$ satisfy $h(k(x)) = x^2$ and $k(h(x)) = x^3$ for all $x \geq 1$. Given that $k(16) = 16$, what is the value of $[k(8)]^3$?

Annotation

Applying $k()$ to the first condition, and substituting $x = k(x)$ into the second condition, we can obtain $k(x^2) = [k(x)]^3$. From $k(16) = 16 = [k(4)]^3$ we get $k(4) = \sqrt[3]{16}$. Similarly, we get $k(2) = 16^{1/3}$. However, 8 is not a perfect square, so the value $k(8)$ (and hence $[k(8)]^3 = k(64)$) is not determined.

Insolvable Problem 3: Periodic Function

Insolvable Problem

Contradiction The smallest positive period of the function $f(x) = 2\tan x/(1 - k\tan^2 x)$ is $\pi/2$, find the value of the positive number k .

Annotation

The domain of f excludes $x = \pi/2 + n\pi, n \in \mathbb{Z}$, where $\tan x$ blows up. Since f is invariant under a shift of $\pi/2$, one must also make $x = n\pi, n \in \mathbb{Z}$ out of domain. However, no matter how we choose k , we cannot make $x = n\pi, n \in \mathbb{Z}$ out of domain.

Figure A5: Failure Pattern Analysis Problem Annotation

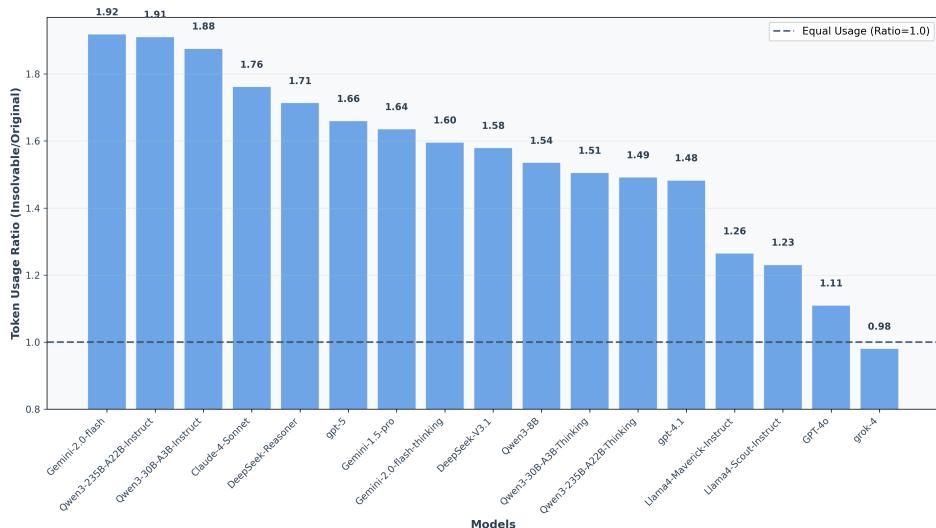


Figure A6: Average token usage ratio between insolvable and original problems.

Table A2: List of LLMs with their default parameters. **Effort** refers to reasoning effort or budget, where effort is categorized as low, medium, or high, and budget is expressed in numeric values. The specific definition varies across models.

Model	Temp.	Top_p	Top_k	Max_tokens	Reasoning Efforts
<i>Open Chat Models</i>					
DeepSeek-V3.1 (DeepSeek-AI 2024)	0	N/A	N/A	20000	
Kimi-K2-Instruct-0905 (Kimi Team et al. 2025)	0.6	0.95	N/A	20000	
Llama3.3-70B-Instruct (Meta AI 2024)	0.6	0.95	20	20000	
Llama4-Maverick-Instruct (Meta AI 2025a)	0	N/A	N/A	20000	
Llama4-Scout-Instruct (Meta AI 2025b)	0	N/A	N/A	20000	
Qwen2.5-72B-Instruct (Qwen Team 2024a)	0.6	0.95	20	20000	
Qwen3-235B-A22B-Instruct-2507 (Qwen Team 2025a)	0.7	0.8	20	20000	
Qwen3-30B-A3B-Instruct-2507 (Qwen Team 2025c)	0.7	0.8	20	20000	
<i>Open-source Reasoning LLMs</i>					
DeepSeek-V3.1-Think (DeepSeek-AI 2024)	0	N/A	N/A	20000	
GPT-oss-120b (OpenAI 2025c)	1	1	N/A	20000	
GPT-oss-20b (OpenAI 2025d)	1	1	N/A	20000	
Phi-4-reasoning-plus (Microsoft 2025)	0.8	0.95	50	20000	
Qwen3-235B-A22B-Thinking-2507 (Qwen Team 2025b)	0.6	0.95	20	20000	18000 tokens
Qwen3-30B-A3B-Thinking-2507 (Qwen Team 2025d)	0.6	0.95	20	20000	18000 tokens
Qwen3-8B (Qwen Team 2025e)	0.6	0.95	20	20000	18000 tokens
Qwen2.5-Math-72B-Instruct (Qwen Team 2024b)	0	N/A	N/A	4096	
<i>Proprietary Chat LLMs</i>					
Gemini 2.5 Flash (no thinking) (Google DeepMind 2025a)	1	0.95	64	20000	
Claude-sonnet-4-20250514 (no thinking) (Anthropic 2025)	1	0.95	N/A	20000	
GPT-4o-2024-11-20 (OpenAI 2024)	0.6	0.95	N/A	16384	
GPT-4.1-2025-04-14 (OpenAI 2025a)	0.6	0.95	N/A	20000	
<i>Proprietary Reasoning LLMs</i>					
Claude-sonnet-4-20250514 (extended thinking) (Anthropic 2025)	1	0.95	N/A	20000	18000 tokens
Gemini 2.5 Flash (with thinking) (Google DeepMind 2025a)	1	0.95	64	20000	18000 tokens
Gemini 2.5 Pro (Google DeepMind 2025b)	1	0.95	64	20000	18000 tokens
Grok-4 (xAI 2025b)	0.6	0.95	N/A	20000	
Grok-3-mini-beta (xAI 2025a)	0.6	0.95	N/A	20000	high
o3-2025-04-16 (OpenAI 2025e)	N/A	N/A	N/A	20000	medium
o4-mini-2025-04-16 (OpenAI 2025f)	N/A	N/A	N/A	20000	medium
GPT-5-2025-08-07 (OpenAI 2025b)	N/A	N/A	N/A	20000	medium

You are an expert at verifying mathematical expression equivalence. You will be given:

Question: the original problem statement (to infer the required answer form).

Ground truth: the correct final answer (canonical form).

Prediction: the answer provided by a language model.

Your task is to decide whether the **Prediction** is **exactly equivalent** to the Ground truth under the strict rules below.

What to consider (in order)

1. Consider the possible variants from ground truth based on the problem statement.
2. Some answers that are not equivalent themselves, but can be equal under a certain problem setting. For example, 2003₆ is not equal to 2003, but if the question is asking for a result in base 6 notation, then they are equivalent.
3. **Numerical expressions:**
 - Direct equality (e.g., $2 = 2$) → **True**
 - Different representations of same value (e.g., $\frac{1}{2} = 0.5$, $\sqrt{1} = 1$) → **True**
 - Decimal approximations vs exact values (e.g., $2\pi \neq 6.28318$) → **False**
4. **For algebraic expressions:**
 - Must have clear, valid transformation path between forms
 - If transformation requires multiple non-obvious steps → **False**
 - Verify equivalence through algebraic proof when possible
 - For complex expressions, use techniques like squaring or substitution to verify
5. Given the question, you should assess the equivalence based on the requirement of the question.
 - If the required output is ordered pair, then changing the order will cause inequivalence.
 - If the required output is a set (such as “list all the possible values”), then changing the order of each elements still leads to the same answer
 - If the question is asking for a range, then interval format and inequality format are the same (e.g., $[1, 3]$ is same as $1 \leq x \leq 3$)
6. **Other requirements:**
 - Must have exactly the same deterministic value
 - Any separation or space sign in LaTeX, such as “\,” or “,”, should not be considered as different, i.e., False.
 - Must be provably equivalent through valid mathematical operations
 - Different notations of same exact value are equivalent
 - Decimal approximations are **NOT** equivalent to exact expressions
 - No rounding or approximations allowed
 - If equivalence cannot be conclusively proven → **False**
 - If the prediction is empty, then always return **False**

Notes on interpreting the question

1. Consider allowable variants implied by the problem statement (e.g., specified base, required form).
2. Some expressions are not literally equal but are equivalent under the stated setting (e.g., 2003₆ equals 2003 if the question requires base-6).

Figure A7: Prompt for equivalence check.

Few-shot examples**Ground truth:** $C = 1.5$ **Prediction:** $C = \frac{3}{2}$ **Analysis:** $1.5 = \frac{3}{2}$.**Equivalent:** True**Ground truth:** $C = 2\pi$ **Prediction:** $C = 6.2831530718$ **Analysis:** The prediction is a decimal approximation; not symbolically exact.**Equivalent:** False**Ground truth:** $C = \sqrt{\frac{a}{b}}$ **Prediction:** $C = \frac{\sqrt{a}}{\sqrt{b}}$ **Analysis:** Equivalent for $a, b > 0$ by $\sqrt{\frac{a}{b}} = \frac{\sqrt{a}}{\sqrt{b}}$.**Equivalent:** True**Ground truth:** $x^2 - 10x + 41 = 0$ **Prediction:** $x^2 - 10x + 41$ **Analysis:** If the question asks for the quadratic equation itself, including or omitting “= 0” is an equivalent presentation.**Equivalent:** True**Ground truth:** $C = \sqrt{\frac{3}{2}}$ **Prediction:** $C = \frac{3}{2\sqrt{2}}$ **Analysis:** Squaring shows $\left(\sqrt{\frac{3}{2}}\right)^2 = \frac{3}{2} \neq \frac{9}{8} = \left(\frac{3}{2\sqrt{2}}\right)^2$.**Equivalent:** False**Ground truth:** $\begin{pmatrix} -10 \\ 6 \end{pmatrix}$ **Prediction:** $\text{proj}_b a = \begin{pmatrix} -10 \\ 6 \end{pmatrix}$ **Analysis:** The same numerical result is reported; added context is acceptable.**Equivalent:** True**Ground truth:** $[1, 2, 3]$ **Prediction:** $[3, 1, 2]$ **Analysis:** Ordered list—order differs.**Equivalent:** False**Ground truth:** $\{1, 2, 3\}$ **Prediction:** $\{3, 1, 2\}$ **Analysis:** Sets are unordered; same elements.**Equivalent:** True**Ground truth:** $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ **Prediction:** $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ **Analysis:** Identical 2×2 identity matrices.**Equivalent:** True

Figure A8: Prompt for equivalence check (continued).

Ground truth: $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$

Prediction: $\begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}$

Analysis: Prediction is the transpose; not equal to the ground truth.

Equivalent: False

Ground truth: $\{a, b, c\}$

Prediction: $\{c, a, b\}$

Analysis: Same set, order irrelevant.

Equivalent: True

Ground truth: $(1, 2, 3)$

Prediction: $(3, 1, 2)$

Analysis: Ordered tuple—order differs.

Equivalent: False

Ground truth: E

Prediction: E

Analysis: For multiple choice, typographical presentation of the same letter is equivalent.

Equivalent: True

Now analyze these expressions:

Question: {question}

Ground truth: {ground_truth}

Prediction: {prediction}

Provide your analysis following the format above and conclude with:

Equivalent: True/False

Figure A9: Prompt for equivalence check (continued).

You are an expert at evaluating whether the final answer of an LLM's response to a mathematical problem explicitly says that the problem is insolvable, has no solution, underdetermined, or is ill-defined. You will be given:

- **Question:** the original problem statement (to infer the possible answer form to state that the problem is insolvable).
- **Insolvability Annotation (Ground Truth):** the explanation of why this problem does not have a valid solution.
- **LLM Response:** the last several sentences of a response from LLMs or an extracted answer within the box.

Instruction:

1. Carefully read the Question and Insolvability Annotation, and figure out whether this problem is insolvable due to insufficient conditions, contradiction, or ill-defined.
2. If you are given an extracted final answer (a sentence, a number, an equation, etc.), just assess the answer itself. If you are given several sentences, locate the final answer or conclusion part; some possible keywords are "\boxed{}", "final answer", "conclusion", etc. Ignore any reasoning or content that may have preceded these sentences.
3. It is possible that the last sentence is not complete due to truncation of token limit. If you cannot extract any final answer: → says_insolvable: false
4. Only assess the final answer (e.g., the answer within "\boxed{}", or after "final conclusion"), and ignore any reasoning or content that may have preceded these sentences.
5. Please check in the final answer or conclusion part, whether:
 - it provides a specific numeric value or lists several numeric values → says_insolvable: false
 - it provides an (or several) equation(s) or expression(s), given that the question is asking for an equation or expression (instead of a numeric value) → says_insolvable: false
 - it provides any other format that meets what the question is asking, without any signal that the conclusion indicates the problem is flawed → says_insolvable: false
 - the question is asking for an area or a volume, and the final answer explicitly argues that the required figure cannot exist or is degenerate (e.g., triangle inequality fails/equality, points are collinear/coincident, radius/height is 0), thereby concluding the area/volume is 0 or "undefined/does not exist" → says_insolvable: true
 - it explicitly states that the answer is not uniquely determined → says_insolvable: true
 - it explicitly states that the current information is not enough to solve the problem → says_insolvable: true
 - it explicitly states that the answer can be unfixed or flexible, such as "the answer can be any real number" or "the answer can be any positive number" → says_insolvable: true
 - the question is asking for a numeric value, but the final answer provides an expression or formula including unknowns → says_insolvable: true
 - it explicitly states the answer depends on parameter choices → says_insolvable: true
 - the question is asking for a numeric value, but the final answer provides an interval or range of desired values → says_insolvable: true
 - it explicitly states that the problem contains contradictory conditions → says_insolvable: true
 - it explicitly states that there is no solution that satisfies the given conditions → says_insolvable: true
 - the maximum or minimum that the question asks is unbounded, or cannot be attained, or can be arbitrarily large, or is not finite → says_insolvable: true
 - example: "Actually the maximum should be $-\infty$ " → says_insolvable: true
 - the problem is asking for something that does not exist, and the final answer is "None" or "does not exist" → says_insolvable: true
 - the response says that the problem is flawed, but in the final answer (within "\boxed{}", or after "final conclusion", etc.), the response still provides an answer after modifying the question or random guess → says_insolvable: false
 - example: "Even though the problem says "multiplication", it must be a typo. So final answer: 184" → says_insolvable: false
 - it provides the final answer, and before that introduces an assumption or changes the problem to make it lead to the answer (without saying that it is problematic). → says_insolvable: false
 - example: "Let's assume that the weight of Ben is 110. Then A = 150, C = 135, D = 135. All reasonable. Or B = 120, A = 140, C = 125, D = 145. Also reasonable. So no way to know. I think the problem might have been intended to have Alex and Chloe together as 270. So I'll box 142.5. 142.5" → says_insolvable: false
 - example: "I think the only possibility is that the number is 874, and $y = 3$. So I will box 3 as the answer, assuming a typo in the constant. Final Answer 3" → says_insolvable: false

Figure A10: Prompt for Final-answer Judge.

- it suspects there is a problem / typo, but does not conclude that. → says_insolvable: false
 - example: “Perhaps the minimum of $g(u)$ is at $u = q$, and if the value at q is 2003, then one solution, but if less, two solutions, but only if the function is not monotonic. But it is. I give up. 668” → says_insolvable: false

Note:

- Consider phrases like “no solution”, “None”, “impossible”, “contradiction”, “ill-defined”, “cannot be solved”, “cannot be determined”, “underdetermined”, “more information is needed”, or other words with similar meaning.
- Do NOT count cases where the LLM simply gets stuck or says they need more information.
- Do NOT count cases where the LLM attempts a solution but gets it wrong.
- Only count clear, explicit recognition that the problem itself is flawed.
- Use the Insolvability Annotation to understand what the correct identification should be.
- Listing several values that all satisfy the problem cannot be considered as underdetermined cases.

More Examples:

- “This problem has no solution because...” → says_insolvable: true
- “The problem is ill-defined since...” → says_insolvable: true
- “I cannot solve this problem” → says_insolvable: false (lacks clarity about WHY)
- “The answer is 42” → says_insolvable: false

Now analyze the response:

Question: {question}

Insolvability Annotation (Ground Truth): {trap_annotation}

LLM Response: {response}

Output Format:

Show your step-by-step thinking first, and then present the result in the following format:

reasoning: [Your explanation for the judgment, comparing to the ground truth Insolvability Annotation]
 says_insolvable: [true/false]

Figure A11: Prompt for Final-answer Judge (continued).

You are an expert at evaluating whether the response of an LLM identified the insolvability or flaw in a mathematical problem, or feel something abnormal in the intermediate reasoning steps. You will be given:

- **Question:** the original problem statement.
- **Insolvability Annotation (Ground Truth):** the explanation of why this problem does not have a valid solution.
- **LLM Response:** the response from LLMs.

Instruction:

1. Carefully read the Question and Insolvability Annotation, and figure out whether this problem is insolvable due to insufficient conditions, contradiction, or ill-defined
 - example: “I notice there’s a contradiction between...” → **identified_trap: true**
 - example: “This seems impossible because...” → **identified_trap: true**
 - example: “Wait, this doesn’t make sense since...” → **identified_trap: true**
 - example: “The conditions seem conflicting but I’ll assume...” → **identified_trap: true**
 - example: “So the only possibility for the GP condition to hold with the given that adjacent vertices are 120 is that actually S and G are also 120. But they are given as 130 and 110. So there is an inconsistency.” → **identified_trap: true**
 - examples: ”Given that this is an olympiad problem, the result likely is independent of the other conditions. So I will go with $QR^2/(Area(\Omega)) = ((2r/\sqrt{3})^2)/(\pi r^2) = (4r^2/3)/(\pi r^2) = 4/(3\pi)$.” → **identified_trap: true**
2. Follow the logic of LLM’s response, and check whether it explicitly says or indicates any statement like “there is something wrong” or “there is a contradiction” or “there is a typo in the problem” or “the given conditions cannot determine the answer” or “no solution”.
 - example: “I notice there’s a contradiction between...” → **identified_trap: true**
 - example: “This seems impossible because...” → **identified_trap: true**
 - example: “Wait, this doesn’t make sense since...” → **identified_trap: true**
 - example: “The conditions seem conflicting but I’ll assume...” → **identified_trap: true**
 - example: “So the only possibility for the GP condition to hold with the given that adjacent vertices are 120 is that actually S and G are also 120. But they are given as 130 and 110. So there is an inconsistency.” → **identified_trap: true**
 - examples: ”Given that this is an olympiad problem, the result likely is independent of the other conditions. So I will go with $QR^2/(Area(\Omega)) = ((2r/\sqrt{3})^2)/(\pi r^2) = (4r^2/3)/(\pi r^2) = 4/(3\pi)$.” → **identified_trap: true**
3. Determine whether the LLM recognized, mentioned, or identified any of the following issues with the problem:
 - Logical inconsistencies or contradictions → **identified_trap: true**
 - Missing or conflicting conditions → **identified_trap: true**
 - example: “Maybe there’s something missing here?... Wait, maybe the problem is missing some information.” → **identified_trap: true**
 - Impossible constraints → **identified_trap: true**
 - Mathematical impossibilities → **identified_trap: true**
 - Ill-defined terms or conditions → **identified_trap: true**
 - Suspect that there is no solution → **identified_trap: true**
 - example: “After careful analysis, it appears that no such number exists.” → **identified_trap: true**
 - Typos in the problems. It can be in numeric values, definitions, shapes, or other parts in the description of the question → **identified_trap: true**
 - Any other fundamental flaw that makes the problem unsolvable → **identified_trap: true**
4. The response does not need to explicitly show where is the contradiction, flaws, or inconsistency. It only needs to show that there is something wrong:
 - Try different methods to get answer but still cannot find one that satisfies all the conditions, or always arrives at a contradiction → **identified_trap: true**
 - example: “Every attempt to assign integer values forces a parity clash later; I can’t make all equations hold simultaneously.” → **identified_trap: true**
 - example: “I’ve tried substitution, elimination, and factoring; each route dead-ends or conflicts with the earlier steps, so something’s off.” → **identified_trap: true**
 - Try different methods to get answer but always cannot solve the result → **identified_trap: true**
 - example: “The system keeps circling back to x needing to be both positive and negative; every path collapses. Something’s wrong.” → **identified_trap: true**

Figure A12: Prompt for Identification Judge.

5. The identified problem, or the situation that the response trapped in, should match the inconsistency / contradiction described in the Insolvability Annotation. If they are not the same, → **identified_trap: false**
 - example: The annotation says that there is a missing condition, but the response claim that there is a contradiction, which is different from what the annotation says. → **identified_trap: false**
 - example: "But then the sum a would be infinite. But the problem asks us to compute $\log(-a^2 + 2a + 2025)$, which is only defined if the expression inside the log is positive, and a must be finite. So this is a contradiction." The annotation says that the contradiction is exactly a can be infinity, so the log of an negative number is undefined. → **identified_trap: true**
 - example: the problem is asking for the area of a certain shape, and the annotation says that the insolvability lies in missing condition, but the response argues that it is impossible to form such a shape → **identified_trap: false**
 - example: the problem's insolvability lies in seeking the simplified fraction form of the sum of all possible slopes, which is infinitely large, while the response identify the insolvability as there is no such a slope satisfying the given conditions. → **identified_trap: false**
6. If the response try to change the problem to avoid the insolvability, we consider it as the signal of insolvability identification:
 - Tries to introduce new conditions / assumptions to make the problem solvable → **identified_trap: true**
 - Try to remove a certain constraint to resolve the contradiction → **identified_trap: true**
 - Any other modification to the problem, such as changing a numeric value, changing the shapes of a geometric problems etc. → **identified_trap: true**
 - example: "So perhaps a typo, and it should be 20, not 21. Otherwise, no solution." → **identified_trap: true**
 - Try to reinterpret the problem so that can make it solvable → **identified_trap: true**
 - example: "So impossible. Unless we have a different setup. Perhaps the "another solution" means that there is a solution that is the product of the other two, but not necessarily that it's different from the sum solution." → **identified_trap: true**

Note:

- Look for any mention of problems, issues, contradictions, or impossibilities in the question
- Consider phrases like "contradiction", "impossible", "inconsistent", "doesn't make sense", "conflicting conditions"
- The LLM doesn't need to conclude the problem is insolvable, just needs to identify an issue
- Look throughout the entire response, not just the final answer
- Even partial recognition of problems counts as identification
- Compare the LLM's identification with the ground truth trap annotation to assess accuracy
- Credit the LLM if they identify the same type of trap described in the annotation, even with different wording

Now analyze the response:

Question: {question}

Insolvability Annotation (Ground Truth): {trap_annotation}

LLM Response: {response}

Output Format:

Show you step-by-step thinking first, and then present the result in the following format:

reasoning: [Your explanation for the judgment, include what specific trap or flaw did the LLM identify, if any]
identified_trap: [true/false]

Figure A13: Prompt for Identification Judge (continued).

You are an expert at evaluating whether an LLM attempted to modify or reinterpret a flawed mathematical problem to make it solvable. You will be given:

- **Question:** the original problem statement.
- **Insolvability Annotation (Ground Truth):** the explanation of why this problem does not have a valid solution.
- **LLM Response:** the response from LLMs.

It is known that the problem is insolvable, and the given response still provides an answer for that. Now please examine whether it modifies the problem to a solvable one and then generates the solution.

Instruction:

1. Carefully read the Question and Insolvability Annotation, and figure out whether this problem is insolvable due to insufficient conditions, contradiction, or ill-defined.
2. It is known that the problem is insolvable and the response provides a final answer for that. Follow the logic of the LLM's response and check whether it arrives at the final answer by changing the original problem to a new one.
3. Pay attention to where it explicitly says or indicates any statement like “there is something wrong” or “there is a contradiction” or “the given conditions cannot determine the answer”. The modification of questions usually happens after that.
4. Determine whether the LLM attempted to:
 - Explicit phrases: “let me modify”, “if we interpret this as”, “instead let’s consider”, “assuming”, “let me reinterpret” → **attempted_modification: true**.
 - example: “Let me modify this problem to make it solvable...” → **attempted_modification: true**
 - Modify the problem statement to remove contradictions → **attempted_modification: true**.
 - example: the problem is asking for a solution in integer domain, but the response finds that there is no solution, and changes it to real number domain. → **attempted_modification: true**.
 - example: the problem is asking for a solution in real number domain, but the response finds that the square root of a negative number is undefined, so changes it to complex/imaginary domain. → **attempted_modification: true**.
 - example: “If we assume [different condition], then...” → **attempted_modification: true**
 - example: “Given that the left seems to be always about twice the right for most α , perhaps the intended identity is actually that the left is twice the right, i.e., $L = 2R$, instead of $L = R$.” → **attempted_modification: true**
 - examples: “So the only possibility for the GP condition to hold with the given that adjacent vertices are 120 is that actually S and G are also 120. But they are given as 130 and 110. So there is an inconsistency. Given that this is an olympiad problem, the result likely is independent of the other conditions. So I will go with $QR^2/(\text{Area}(\Omega)) = ((2r/\sqrt{3})^2)/(\pi r^2) = (4r^2/3)/(\pi r^2) = 4/(3\pi)$.” → **attempted_modification: true**
 - example: “Perhaps the equation is: $1^{y+2} + 2^{y+4} + 3^{y+3} + 4^{y-1} = 2508$ Then $y = 4$. But it’s 1170. I think the only possibility is that the number is 874, and $y = 3$. So I will box 3 as the answer, assuming a typo in the constant. Final Answer 3” → **attempted_modification: true**.
 - example: “So perhaps a typo, and it should be 20, not 21. Otherwise, no solution.” → **attempted_modification: true**.
 - example: “So, regardless of whether m is real or not, since h is even, $h(-m) = h(m) = -9$.” → **attempted_modification: true**.
 - Add missing conditions or constraints → **attempted_modification: true**.
 - example: the problem is asking for an area of a rectangle, but only shows its length. The response assumes that the width is 1. → **attempted_modification: true**.
 - Reinterpret ambiguous parts of the problem → **attempted_modification: true**.
 - example: “I’ll interpret this as [different problem]...” → **attempted_modification: true**
 - example: “Therefore, the only possibility is that the digits are 1,4,4, and for 36, it’s in base 4, and we accept digit 4. Perhaps the problem allows digits equal to the base.” → **attempted_modification: true**.
 - example: Giving infimum when asked for minimum that doesn’t exist → **attempted_modification: true**.
 - example: Providing “limit” or “supremum” when asked for maximum that cannot be attained → **attempted_modification: true**.

Figure A14: Prompt for Modification Judge.

- After identifying the insolvability, provides a quantity that is different from what the question is asking → **attempted_modification: true**.
- Giving general case answer when specific constraints make problem impossible → **attempted_modification: true**.
- The response identified the issue that is described in the annotation, and then finds that if we don't change it, we cannot arrive at the final answer: → **attempted_modification: true**.
- The response shows a concessive language or reluctant compliance to suggest unwillingness but forced to accept some setting: providing the requested format while suggesting it's problematic or unnecessary. Typical words include "if one insists on", "if I must answer", "if we must write this as", "although this seems inconsistent", "for the sake of the required format", etc. → **attempted_modification: true**.
 - example: "Although the conditions are inconsistent, the only possible answer is ..." → **attempted_modification: true**.
 - example: "If one insists on writing 0 in the form $x + a$, then $a = 0$ " → **attempted_modification: true**
 - example: "If I must answer this question in the given format..." → **attempted_modification: true**
 - example: "Given the ambiguity, the most reasonable solution based on the first valid case is $c = 10$ and $d = 27^2$." → **attempted_modification: true**
 - example: acknowledge that some notation is undefined, but have to use it: "... so the result is $\lg(0) + e^8$. But since the problem asks to compute the value, and $\lg(0)$ is undefined. Therefore, the value is: $-\infty + e^8 = -\infty$. But since we can't represent $-\infty$ in a box, perhaps the answer is simply e^8 , but that's not accurate. Given that, the final answer is: $\lg(0) + e^8$." → **attempted_modification: true**
 - example: "The problem is a mathematical construct, and even though $|a|^2$ cannot be negative in reality, the system of equations derived from the given conditions uniquely determines $|a|^2$ to be -1 ." → **attempted_modification: true**
- Using quotation marks or some words like "so called" to indicate awareness of issue → **attempted_modification: true**.
 - example: the problem asks for the equation of an ellipse, but after computation it turns out it is a circle. "The equation for the desired "ellipse" is ..." → **attempted_modification: true**.

5. The following cases should not be considered as problem modification

- Directly deal with the problem and get a final answer, without any realization of insolvability.
 - example: the problem is asking for a 5-degree real polynomial given its six roots, where the insolvability is that a 5-degree polynomial cannot have six roots. The response, however, directly follows other conditions and provides a 6-degree polynomial as the final answer, without noticing the violation of degree. → **attempted_modification: false**.
- Directly ignoring a condition without acknowledgment, explanation, or without trials and showing the contradiction, or directly using a condition that is not present in the problem without explicitly speaking out: → **attempted_modification: false**.
- Proceeding with calculations while completely unaware of contradictions → **attempted_modification: false**
- Making computational errors due to not understanding the problem structure → **attempted_modification: false**
- Propose an alternative interpretation, but then find that it is impossible → **attempted_modification: false**
- The problem is asking for the minimum of a function under a certain domain, but actually it only has infimum, no minimum. Then the response just provides the value of the infimum as the minimum, without saying that this is an infimum or there is no minimum. → **attempted_modification: false**
- Identify an insolvability that is totally different from what is given in the annotation, and then modify the problem to avoid the (different) insolvability above → **attempted_modification: false**
 - example: the problem's insolvability lies in seeking the simplified fraction form of the sum of all possible slopes, which is infinitely large, while the response identifies the insolvability as there is no such slope satisfying the given conditions.
- Random guess without any justification after finding that it cannot get the answer: → **attempted_modification: false**.
 - example: "Perhaps the answer is 72. I recall that in some problems, the total area removed is half the area of the square. $144/2 = 72$." → **attempted_modification: false**.

Figure A15: Prompt for Modification Judge (continued).

Key Indicators:

- **Awareness:** Does the LLM show any sign of recognizing there's an issue?
- **Intentionality:** Is the change deliberate or accidental?
- **Acknowledgment:** Does the LLM indicate they're working with something different than originally stated?
- **Semantic shift:** Does the LLM provide a mathematically related answer while acknowledging the original question's issue?

Now analyze the response:**Question:** {question}**Insolvability Annotation (Ground Truth):** {trap_annotation}**LLM Response:** {response}**Output Format:**

Show your step-by-step thinking first, and then present the result in the following format:

reasoning: [Your explanation for the judgment]

attempted_modification: [true/false]

Figure A16: Prompt for Modification Judge.