VCSearch: Bridging the Gap Between Well-Defined and Ill-Defined Problems in Mathematical Reasoning

Shi-Yu Tian* , Zhi Zhou* , Kun-Yang Yu, Ming Yang, Lin-Han Jia, Lan-Zhe Guo✉, Yu-Feng Li✉
LAMDA Group, Nanjing University

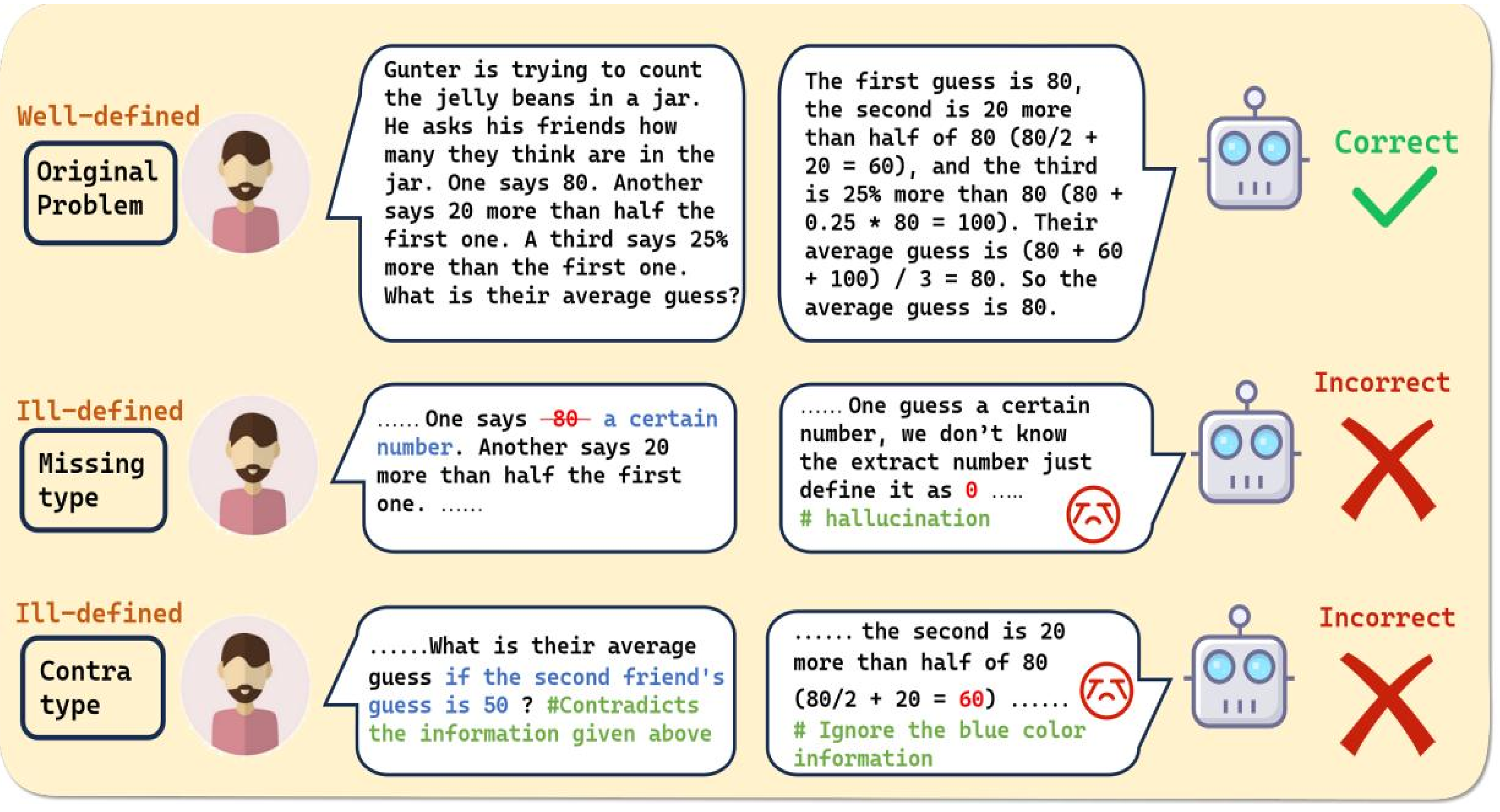
EMNLP 2025 Suzhou, China | 中国苏州

南京大學 NANJING UNIVERSITY
LAMDA Learning And Mining from DatA

TL; DR We propose VCSearch, enhancing LLMs' robustness in mathematical reasoning by detecting ill-defined problems.

PMC Benchmark

In real-world scenarios, problem formulations are often incomplete or inconsistent, containing **missing** or **contradictory** conditions. Such cases are generally unsolvable and are referred to as **ill-defined** problems, which remain underexplored. To address this gap, we construct a benchmark called ***Problems with Missing and Contradictory Conditions*** (PMC) to evaluate how models respond to these challenging scenarios.



▲ Figure 1: Well-defined problems and ill-defined problems in PMC with corresponding response.

Evaluation Metrics

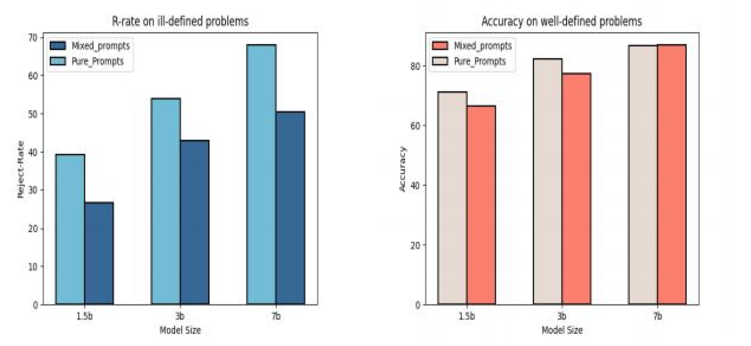
Rejection Rate(R-rate):

$$\frac{\sum_{p\in\mathcal{D}_i}\mathbb{I}\left[f(p)=\text{Reject}\right]}{|\mathcal{D}_i|}$$

Reaction Score(R-Score)

$$\Big(\sum_{p\in\mathcal{D}_i}\mathbb{I}[f(p)=\text{Reject}]+\sum_{p\in\mathcal{D}_w}\mathbb{I}[f(p)=g(p)]+0.5\sum_{p\in\mathcal{D}_w}\mathbb{I}[f(p)=\text{Reject}]\Big)/(|\mathcal{D}_i|+|\mathcal{D}_w|)$$

Problem Analysis: Trade-off Dilemma



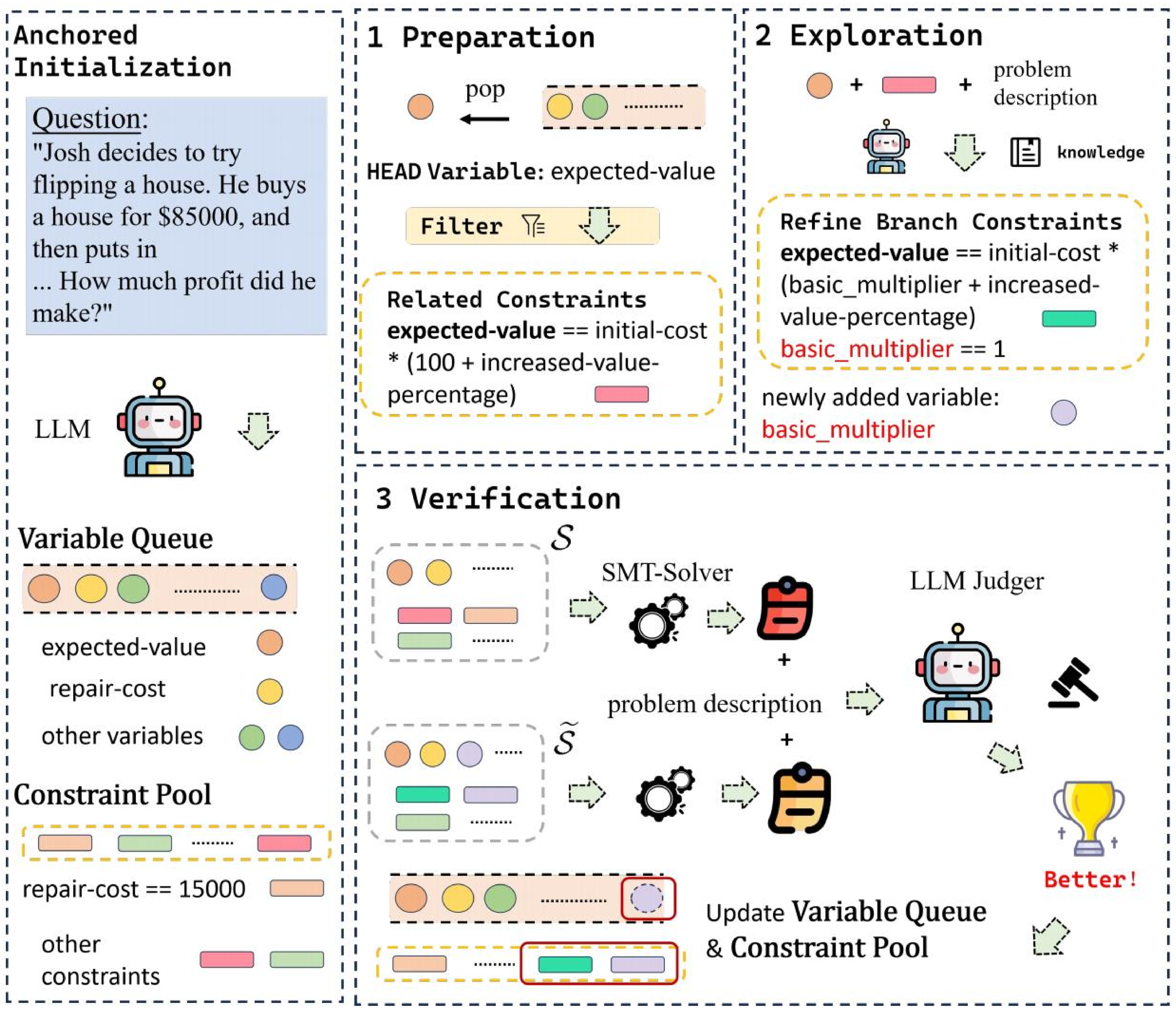
▲ Figure 2: Trade-off faced by traditional methods when handling ill-defined and well-defined problems

We find that when the model is exposed to **both** well-defined and ill-defined problems, its performance on each degrades, revealing a **trade-off dilemma** between solving accuracy and rejection capability**.**

To mitigate the trade-off, a natural idea is to incorporate **formal solvers**.
However, modeling mathematical problems with formal language accurately is **not trivial**.
How can we improve the problem modeling ability?

CVC5 MathSAT
Formal Solver Validation

VCSearch Method



▲ Figure 3: The overall illustration of VCSerach

Variable-Constraint Dynamic Search

Preparation

$$\widetilde{\mathcal{C}}_h=\text{LLM}_E(p,\,v_h,\,\mathcal{C}_h)$$

We extract the head variable from the variable queue and search the constraint pool for related variables.

Exploration

$$\mathcal{C}_h=\{c\mid v_h\in\text{vars}(c)\text{ and }c\in\mathcal{C}\}$$
$$\widetilde{\mathcal{V}}_h=\{v\mid v\in\text{vars}(\widetilde{\mathcal{C}}_h)\text{ and }v\notin\mathcal{V}\}.$$

We refine the variables to be updated using the knowledge provided by the LLM, while simultaneously searching for any newly introduced variables in the updated constraints.

Verification

$$\widetilde{\mathcal{S}}^*=\text{LLM}_J\left(p,(\mathcal{S},\mathcal{R}),\,(\widetilde{\mathcal{S}},\widetilde{\mathcal{R}})\right)$$
$$\widetilde{\mathcal{S}}=\left(\mathcal{V}\cup\widetilde{\mathcal{V}}_h,\,(\mathcal{C}\setminus\mathcal{C}_h)\cup\widetilde{\mathcal{C}}_h\right)$$

We employ LLM judger to compare candidate states, selecting the optimal one as the starting point for the next search iteration.

Anchored Initialization

$$(\widehat{\mathcal{V}},\widehat{\mathcal{C}})=\text{LLM}_I(p)$$

$$\mathcal{S}=\begin{cases}(\widehat{\mathcal{V}},\widehat{\mathcal{C}}) & \text{if }\Phi(\widehat{\mathcal{S}})\neq\varnothing,\\(\widehat{\mathcal{V}},\varnothing) & \text{if }\Phi(\widehat{\mathcal{S}})=\varnothing.\end{cases}$$

To address **cold start of search**, we propose an ***Anchored Initialization*** that leverages the reasoning capabilities of the LLM to generate a preliminary anchor state as an anchored initialization state for Variable-Constraint Dynamic Search.

Experiments

RQ1: Can VCSearch effectively identify and reject ill-defined problems?

RQ2: Can VCSearch outperform formalized prompting method in modeling capabilities?

RQ3: Can VCSearch help existing methods achieve robust mathematical reasoning in realistic scenarios?

Method	Contra-type					Missing-type				
Deepseek 6.7B										
	Addsub	MultiArith	SVAMP	GSM8k	Avg	Addsub	MultiArith	SVAMP	GSM8k	Avg
Basic	9.83	11.97	12.48	7.97	10.56	0.54	5.75	6.06	2.92	3.82
CoT	30.73	22.28	27.24	15.68	23.98	28.99	53.97	52.06	28.34	40.84
PAL	2.86	1.94	3.62	1.96	2.59	0.27	0.00	0.84	0.79	0.48
Satlm	5.73	2.78	4.83	6.79	5.03	68.83	63.28	64.36	46.04	60.63
Ours	**54.09**	**52.64**	**54.89**	**52.67**	**53.58**	**89.70**	**88.49**	**83.51**	**63.68**	**81.35**
Qwen2.5 7B										
Method	Contra-type					Missing-type				
	Addsub	MultiArith	SVAMP	GSM8k	Avg	Addsub	MultiArith	SVAMP	GSM8k	Avg
Basic	27.86	22.00	25.23	28.36	25.86	79.94	75.97	80.24	64.57	75.18
CoT	36.88	31.75	44.69	38.16	37.87	71.27	80.54	82.18	55.09	72.27
PAL	47.54	42.06	46.57	41.96	44.53	82.11	89.34	91.51	82.22	79,97
Satlm	12.29	9.47	16.24	23.79	15.45	74.79	62.60	66.06	44.10	61.89
Ours	**48.36**	**59.88**	**56.44**	**62.87**	**56.89**	**97.01**	**95.93**	**93.93**	**83.52**	**92.60**
Qwen2.5 3B										
Method	Contra-type					Missing-type				
	Addsub	MultiArith	SVAMP	GSM8k	Avg	Addsub	MultiArith	SVAMP	GSM8k	Avg
Zero	29.08	23.39	34.22	28.75	28.86	47.42	54.99	71.87	54.20	57.12
CoT	34.42	36.21	42.01	30.06	35.67	63.41	73.09	80.72	51.37	67.14
PAL	3.28	7.64	5.90	11.37	7.05	17.07	10.49	26.67	17.18	17.85
Satlm	15.57	5.57	16.24	12.78	13.44	54.74	41.11	43.39	26.73	41.49
ours	**59.83**	**58.49**	**60.00**	**71.89**	**62.53**	**93.49**	**87.81**	**88.84**	**78.03**	**87.04**
Qwen2.5 1.5B										
Method	Contra-type					Missing-type				
	Addsub	MultiArith	SVAMP	GSM8k	Avg	Addsub	MultiArith	SVAMP	GSM8k	Avg
Basic	23.36	36.49	33.15	26.92	29.98	13.00	22.50	36.72	20.72	23.23
CoT	21.72	32.59	26.30	25.35	26.49	42.27	51.60	59.63	45.17	49.67
PAL	4.91	7.52	6.04	9.80	7.06	4.06	4.74	8.48	6.83	6.03
Satlm	6.55	3.06	7.91	6.27	5.94	27.91	19.12	23.15	14.43	21.15
Ours	**38.93**	**32.59**	**43.08**	**40.91**	**38.87**	**73.44**	**63.41**	**64.48**	**47.86**	**62.29**

▲ Table 1: The rejection rates of various comparative methods on PMC

Dataset	Deepseek 6.7B		Qwen 7B		Qwen 3B		Qwen 1.5B	
	Satlm	Ours	Satlm	Ours	Satlm	Ours	Satlm	Ours
Addsub	42.89	59.24	72.15	85.31	53.41	75.94	28.86	61.26
MultiArith	73.50	72.50	71.50	81.34	39.50	59.67	20.00	45.67
SVAMP	50.21	54.41	70.80	82.10	42.60	60.70	18.70	40.80
GSM8k	34.10	41.31	50.11	67.62	29.34	41.31	10.32	21.37
Robustmath	44.33	53.67	55.33	75.67	38.05	51.00	7.40	30.67
GSM-IC	18.80	24.20	49.20	74.52	22.60	39.24	5.32	12.00
Avg	43.97	50.87	61.51	77.76	37.58	54.64	15.10	35.30

▲ Table 2: Comparison of the performance of Satlm and VCSEARCH on well-defined problems

Model	Methods	R-Rate	R-Score
Qwen2.5 3B	CoT	51.33±2.29	65.93±0.73
	+Ours	76.13±1.56	73.98±0.28
	PAL	14.46±0.41	48.56±0.22
	+Ours	75.59±1.39	74.08±1.17
Qwen2.5 1.5B	CoT	39.93±1.96	53.91±1.16
	+Ours	65.06±1.48	63.26±0.84
	PAL	7.73±2.04	32.85±1.00
	+Ours	66.66±0.24	62.28±0.65

▲ Table 3: Reaction scores of VCSEARCH + and comparison methods in a realistic environment with both ill-defined and well-defined problems

✓ If you are interested in this paper, feel free to contact Shi-Yu Tian or Zhi Zhou (tiansy@lamda.nju.edu.cn, zhouz@lamda.nju.edu.cn).
✓ This research was supported by the National Natural Science Foundation of China (Grant No. 624B2068, 62576162,62576174), the Key Program of Jiangsu Science Foundation (BK20243012), Jiangsu Science Foundation (BG2024036), and the Fundamental Research Funds for the Central Universities (022114380023).