MAXS: Meta-Adaptive Exploration with LLM Agents

Jian Zhang[1,2], **Zhiyuan Wang**[1,3], **Zhangqi Wang**[1,3], **Yu He**[1,2*],
Haoran Luo[4], **Li Yuan**[5], **Lingling Zhang**[1,3], **Rui Mao**[4], **Qika Lin**[6*], **Jun Liu**[1,2]

[1]School of Computer Science and Technology, Xi'an Jiaotong University
[2]Ministry of Education Key Laboratory of Intelligent Networks and Network Security
[3]Shaanxi Province Key Laboratory of Big Data Knowledge Engineering, Xi'an Jiaotong University
[4]Nanyang Technological University [5]South China University of Technology
[6]National University of Singapore

Abstract

Large Language Model (LLM) Agents exhibit inherent reasoning abilities through the collaboration of multiple tools. However, during agent inference, existing methods often suffer from (i) *locally myopic generation*, due to the absence of lookahead, and (ii) *trajectory instability*, where minor early errors can escalate into divergent reasoning paths. These issues make it difficult to balance global effectiveness and computational efficiency. To address these two issues, we propose **m**eta-**a**daptive e**x**ploration with LLM agent**s** (**MAXS**)[1], a meta-adaptive reasoning framework based on LLM Agents that flexibly integrates tool execution and reasoning planning. MAXS employs a lookahead strategy to extend reasoning paths a few steps ahead, estimating the advantage value of tool usage, and combines step consistency variance and inter-step trend slopes to jointly select stable, consistent, and high-value reasoning steps. Additionally, we introduce a trajectory convergence mechanism that controls computational cost by halting further rollouts once path consistency is achieved, enabling a balance between resource efficiency and global effectiveness in multi-tool reasoning. We conduct extensive empirical studies across three base models (MiMo-VL-7B, Qwen2.5-VL-7B, Qwen2.5-VL-32B) and five datasets, demonstrating that MAXS consistently outperforms existing methods in both performance and inference efficiency. Further analysis confirms the effectiveness of our lookahead strategy and tool usage.

1 Introduction

Large Language Model (LLM) Agents (Huang et al., 2024; Yuan et al., 2026) are built on the backbone of LLM, aiming to leverage tools such as search tools and code tools to assist in the reasoning process. LLM Agents are widely used in complex problem-solving (Renze and Guven, 2024; Yuan et al., 2025; Zhang et al., 2025), medical question-answering (Yang et al., 2024), search engines (Nie et al., 2024), and more. Typically, LLM agents generate queries based on reasoning requirements and invoke the search tool to obtain domain-specific knowledge and the latest information, and then use it to obtain the corresponding response (Jin et al., 2025). LLM Agents use the code tool to generate code based on the reasoning needs, which is then executed by an interpreter to return results for precise calculations (Wang et al., 2024). During the reasoning process, LLM Agents appropriately call on the search tool and the code tool to supplement its capabilities and derive the final result, as shown in Figure 1.

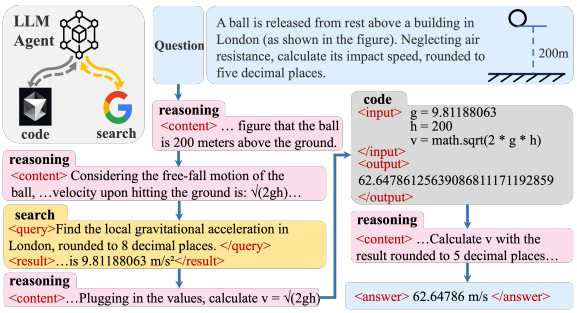

Figure 1: An example of LLM Agents solving a task via multi-step reasoning, dynamically leveraging search and code tools to obtain the final answer.

Various strategies are employed during test-time to improve the efficiency of LLM Agents. As shown in Figure 2, both *Chain of Thought (CoT)* (Wei et al., 2022; Choi et al., 2024) and *Tree of Thought (ToT)* (Yao et al., 2023; Haji et al., 2024) adopt step-by-step reasoning, following prompt-driven incremental trajectories. In contrast, Monte Carlo Tree Search (MCTS) (Luo et al., 2025; Gan et al., 2025) performs global exploration by simulating whole future paths, where each candidate step is evaluated by executing it to completion.

These methods face two major issues. The first

*Corresponding authors
[1]https://github.com/exoskeletonzj/MAXS

Findings of the Association for Computational Linguistics: ACL 2026, pages 13713–13731

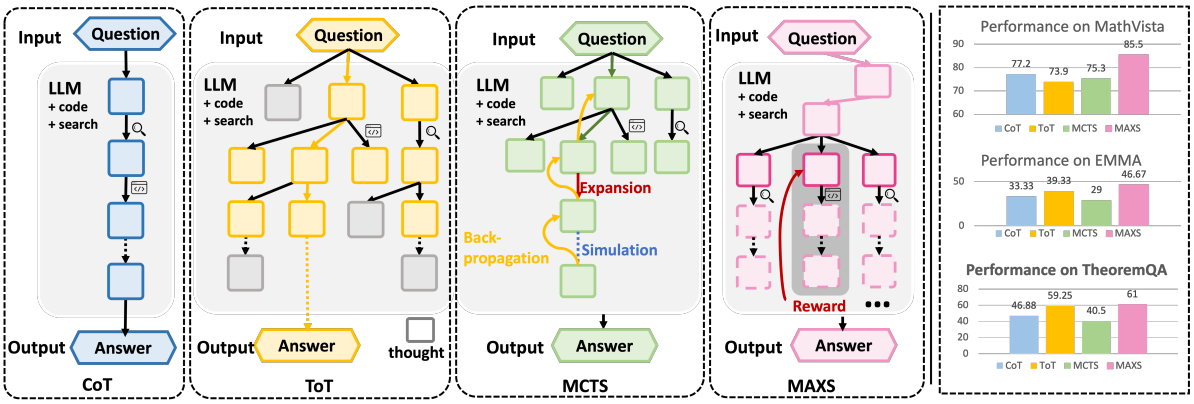

Figure 2: Comparison of test time reasoning strategies. CoT and ToT follow step by step generation with limited foresight, while MCTS conducts global simulation at a higher computational cost. On the right, MAXS uses MiMo-VL-7B-SFT as the backbone and consistently outperforms baseline methods across benchmarks.

is **locally myopic generation**. Whether using CoT or ToT, both approaches rely on the existing sequence for myopic generation. However, in the context of Agents, crucial factors such as whether a tool should be used, whether its use is appropriate, and whether it brings added value are not reflected in the decision-making process. The second issue is **trajectory instability**. Multi-tool reasoning paths are highly sensitive to early decisions, as small errors can accumulate and cause divergence. Tree-based methods like MCTS mitigate this by simulating multiple futures, but at high cost. As shown in Figure 4, MCTS often consume approximately one thousand times more tokens to reach similar performance, due to full-path expansion at each step.

To address these issues, we propose *meta-adaptive exploration with LLM agents* (MAXS), a meta-adaptive reasoning framework based on LLM Agents that flexibly integrates tool execution and reasoning planning. MAXS employs a **lookahead strategy** to extend reasoning paths by a few steps, estimating the potential value of tool usage. It combines step consistency variance and inter-step trend slopes to jointly select stable, consistent, and high-value reasoning steps. Additionally, we introduce a *trajectory convergence* mechanism to control computational costs and improve inference efficiency by halting further rollout once path consistency is achieved. MAXS strikes a balance between resource efficiency and global effectiveness within multi-tool reasoning trajectories.

We conduct extensive empirical studies across five datasets, including *MathVista* (Lu et al., 2023), *OlympiadBench* (He et al., 2024), *EMMA* (Hao et al., 2025), *TheoremQA* (Chen et al., 2023), and *MATH* (Hendrycks et al., 2021), using three LLM backbones: MiMo-VL-7B (Yue et al., 2025), Qwen2.5-VL-7B (Xu et al., 2025b), and Qwen2.5-VL-32B. As shown in the results in Figure 2 and Table 1, *MAXS* outperforms existing methods in both performance and inference efficiency. Ablation studies further validate the effectiveness of the lookahead strategy and tool usage design. Additional experiments confirm the robustness and adaptability of *MAXS* with multi-tool reasoning trajectories. The main contributions of this study are threefold:

- We propose a meta-adaptive agent reasoning framework, *MAXS*. To the best of our knowledge, it is the first method to apply *meta-adaptive exploration* during the inference-time of LLM Agents.
- A lookahead-based estimation strategy alleviates both locally myopic generation and trajectory instability by enabling foresighted, value-aware tool selection and promoting stable reasoning paths.
- Extensive experiments across multiple models and datasets demonstrate the effectiveness of *MAXS*, with ablations and further analyses confirming the key role of the lookahead strategy and tool usage design.

2 Methodology

The architecture is illustrated in Figure 3. In this section, we first introduce the preliminaries of LLM agents-based reasoning. We then present the three key components of MAXS: a lookahead strategy for simulating future steps, a value estimation mechanism for action scoring, and a trajectory con-

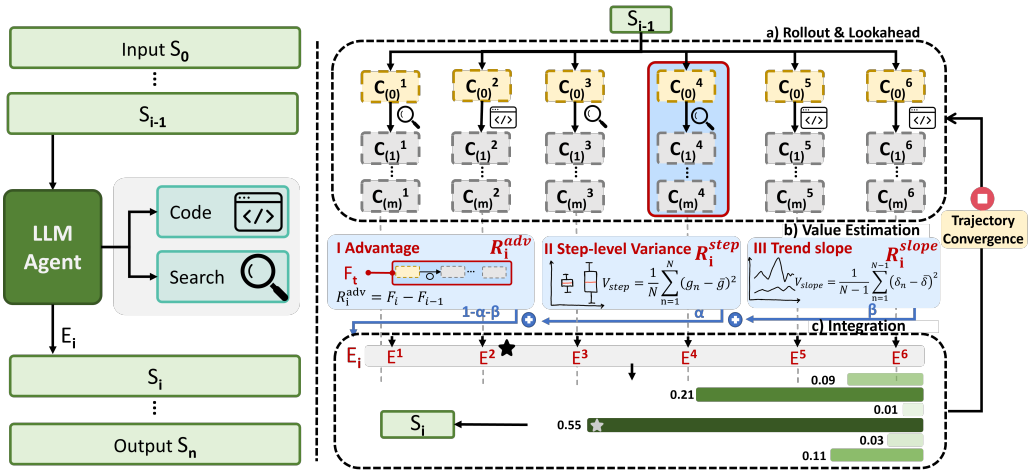

Figure 3: Illustration of the MAXS framework. Left: LLM Agents generates reasoning steps from input s_0 to final answer s_n. Right: At each step, MAXS performs (a) rollout & lookahead, (b) value estimation via advantage and two variance scores, and (c) integration. A trajectory convergence mechanism halts rollouts early to improve efficiency.

vergence module that improves inference efficiency via early rollout termination.

2.1 Preliminaries

Definition 1: Tool-Augmented Reasoning. LLM Agents reasoning is an iterative process where the agent generates steps s_i based on the reasoning history and input, including the question and prompt s_0:

$$s_i \sim \pi_\theta(\cdot \mid s_0, s_{\leq i-1}), \tag{1}$$

where π_θ is the policy of a pre-trained LLM with parameters θ, and $s_{\leq i}$ denotes all previous reasoning steps. In tool-augmented settings, the agent can choose to invoke external tools (e.g., search or code) at selected steps $\mathcal{I}_{\text{tool}} \subseteq \{1, \ldots, T\}$ to enhance reasoning. The final output s_n is generated by combining the input question s_0 with retrieved and computed results:

$$s_n \sim \pi_{\text{final}}\left(s_0;\ \{d_i, r_i\}_{i \in \mathcal{I}_{\text{tool}}}\right). \tag{2}$$

Definition 2: Test-Time Strategy. To improve reasoning quality, the agent may apply a selection policy $\mathcal{Q}$ to refine the next step:

$$\hat{s}_i \sim \mathcal{Q}(\cdot \mid s_0, s_{\leq i-1}), \tag{3}$$

where $\hat{s}_i$ is the selected optimal step, and $\mathcal{Q}$ denotes a test-time strategy such as MCTS.

Definition 3: Search Tool Invocation. At reasoning step i, the agent may generate a query to retrieve external knowledge based on input x:

$$q_i^{\text{search}} \sim \pi_{\text{search}}(s_0, s_i),\ d_i = \text{Search}(q_i^{\text{search}}). \tag{4}$$

The document d_i is used to update the next step.

Definition 4: Code Tool Invocation. At some steps, the agent may also invoke a code tool to perform computation based on the current state and input x:

$$c_i \sim \pi_{\text{code}}(s_0, s_i),\ r_i = \text{Exec}(c_i). \tag{5}$$

The result r_i is integrated into next reasoning process.

2.2 Lookahead Strategy

To mitigate the issue of locally myopic generation, we adopt lookahead via a rollout process. This approach evaluates the current step s_i and future steps $s_{>i}$ to determine the most optimal decision. The lookahead process is defined as:

$$\hat{s}_i \sim \pi_\theta(s_i \mid s_0, s_{<i}, s_{>i}), \tag{6}$$

where s_i is the current reasoning state, s_0 represents the input question and prompt, and $s_{>i}$ includes future steps to be evaluated.

According to the Bellman Optimality Principle (Barron and Ishii, 1989), the value of future steps $R(s_{>i})$ can be recursively estimated as:

$$R(s_0, s \leq i, s > i) = \mathbb{E}\left[\sum_{k=1}^{K} \gamma^{k-1} R(s_{i+k}) \mid s\right], \tag{7}$$

where γ is the discount factor for future steps, K is the maximum number of steps in the lookahead,

Methods	MathVista	OlympiadBench			EMMA				TheoremQA	MATH	Avg.	Tokens
		math	physics	avg.	Math	Phys.	Chem.	avg.				
MiMo-VL-7B-SFT												
CoT	77.20	47.25	30.57	41.57	31.00	33.00	36.00	33.33	46.88	65.67	52.93	2.67×10^7
ToT	73.90	48.51	32.40	43.03	39.00	39.00	40.00	39.33	59.25	69.67	57.04	6.40×10^{10}
MCTS	75.30	28.98	21.83	26.55	31.00	22.00	34.00	29.00	40.50	72.67	48.80	9.91×10^{10}
Guided Decoding	74.30	22.04	20.87	21.64	32.00	29.00	41.00	34.00	39.12	70.33	47.88	1.67×10^8
ϕ-Decoding	74.80	47.86	32.79	42.73	36.00	32.00	41.00	36.33	45.75	73.00	54.52	7.66×10^8
MAXS (ours)	**85.50**	**52.97**	**39.74**	**48.47**	**47.00**	**40.00**	**53.00**	**46.67**	**61.00**	**75.67**	**63.46**	9.86×10^8
Qwen2.5-VL-7B-Instruct												
CoT	49.20	21.32	11.09	17.84	33.00	21.00	19.00	24.33	34.00	50.67	35.21	6.70×10^6
ToT	52.00	20.03	9.48	16.44	25.00	19.00	22.00	22.00	31.00	50.00	34.29	1.37×10^{10}
MCTS	51.80	19.11	9.52	15.84	33.00	20.00	15.00	22.67	31.00	42.67	32.80	4.12×10^{10}
Guided Decoding	44.50	25.46	10.48	20.36	32.00	27.00	16.00	25.00	34.25	53.00	35.42	1.46×10^8
ϕ-Decoding	44.10	26.25	11.05	21.08	20.00	17.00	11.00	16.00	34.75	56.33	34.45	3.17×10^8
MAXS (ours)	**56.80**	**30.49**	**15.20**	**25.28**	**34.00**	**32.00**	**30.00**	**32.33**	**39.50**	**60.33**	**42.85**	4.02×10^8

Table 1: Main results across five benchmarks using different decoding methods, grouped by models. For OlympiadBench and EMMA, both overall averages and subset performances are reported. The 'avg.' column denotes the mean accuracy over MathVista, OlympiadBench(avg.), EMMA (avg.), TheoremQA, and MATH.

	Math	Chemistry	Physics	Avg.
CoT	23.00	33.00	27.00	27.67
ToT	25.00	22.00	24.00	23.67
MCTS	28.00	24.00	19.00	23.67
Guided Decoding	33.00	30.00	28.00	30.33
ϕ-Decoding	31.00	35.00	33.00	33.00
MAXS(ours)	**42.00**	**39.00**	**37.00**	**39.33**

Table 2: Generalization results on the EMMA dataset using Qwen2.5-VL-32B-Instruct.

and s is the whole steps. This allows us to incorporate future trajectory values into the decision-making process.

Proposition 1 (Bellman Recursion). *The optimal action at step i obeys $\hat{s}_i = \arg\max_{s_i} \big[R(s_i * \gamma\, \mathbb{E}_{s_{>i}} V^*(s_{>i})\big]$, hence the sequence's optimum is obtained by recursively combining current utility with the future optimal value.*

The detailed derivation can be found in Appendix A.1. Finally, the current step is selected based on the estimated future values $R(s_{>i})$ as:

$$\hat{s}_i \sim \pi_\theta(s_i \mid s_0, s_{<i})\, e^{\frac{R(s_0, s_{\le i}, s_{>i})}{\tau}}, \tag{8}$$

where τ controls the diversity of the generated steps. The complete algorithm and decoding pipeline are presented in Appendix C.

2.3 Value Estimation

To address trajectory instability, a composite value function evaluates candidate reasoning trajectories, incorporating advantage score, step-level variance, and slope-level variance to promote stable and consistent reasoning.

(1) Advantage Score. We adopt beam search to maintain K candidate paths. At each decoding step i, for each path, we perform M independent stochastic rollouts to simulate possible future trajectories and evaluate the expected lookahead return. Let F_i be the foresight probability at step i under the extended rollout:

$$F_i = \pi_\theta(s_{>i} \mid s_0, s_{\le i}), \tag{9}$$

where $s_{>i}$ denotes the future N steps after i. We define the global advantage as the relative improvement over the previous step:

$$A_i = F_i - F_{i-1}, \quad R_i^{\text{adv}} = \exp\left(\frac{A_i}{\tau}\right), \tag{10}$$

where τ is a temperature parameter controlling sensitivity. R_i^{adv} reflects the progress gained by choosing s_i.

(2) Step-Level Variance. Inspired by Lyapunov stability theory (Shevitz and Paden, 2002), we interpret the lookahead trajectory as a discrete-time dynamical system. Let g_n denote the log-probability of the n-th step in the lookahead segment $s_{>i}$, and define its mean over a rollout of length N as $\bar{g} = \frac{1}{N}\sum_{n=1}^{N} g_n$, and its variance as:

$$V_{\text{step}} = \frac{1}{N}\sum_{n=1}^{N}(g_n - \bar{g})^2. \tag{11}$$

Methods	MathVista	OlympiadBench			EMMA				TheoremQA	MATH	Avg.	Tokens
		math	physics	avg.	Math	Phys.	Chem.	avg.				
MiMo-VL-7B-SFT												
MAXS (ours)	**85.50**	**52.97**	**39.74**	**48.47**	**47.00**	**40.00**	**53.00**	**46.67**	**61.00**	**75.67**	**63.46**	9.86×10^8
w/o lookahead	78.20	49.12	30.96	42.94	42.00	36.00	49.00	42.33	58.38	70.67	58.50	2.44×10^8
w/o $score_{adv}$	81.60	51.74	36.68	46.61	43.00	38.00	51.00	44.00	59.25	73.33	60.96	9.88×10^8
w/o $score_{step}$	82.40	51.15	37.12	46.37	44.00	38.00	51.00	44.33	59.63	74.00	61.35	8.32×10^8
w/o $score_{slope}$	84.10	52.34	38.21	47.53	45.00	38.00	52.00	45.00	60.75	74.67	62.41	8.92×10^8
w/o T.C.	85.10	52.41	39.04	47.86	47.00	39.00	52.00	46.00	60.88	75.33	63.03	9.95×10^8
Qwen2.5-VL-7B-Instruct												
MAXS (ours)	**56.80**	**30.49**	**15.20**	**25.28**	**34.00**	**32.00**	**30.00**	**32.33**	**39.50**	**60.33**	**42.85**	4.02×10^8
w/o lookahead	46.30	23.46	10.17	18.94	24.00	23.00	22.00	23.00	28.50	50.33	33.41	1.76×10^8
w/o $score_{adv}$	48.10	27.96	12.45	22.68	29.00	26.00	25.00	26.67	33.25	54.00	36.94	4.01×10^8
w/o $score_{step}$	50.40	28.41	12.71	23.07	28.00	26.00	25.00	26.33	33.88	54.67	37.67	3.87×10^8
w/o $score_{slope}$	53.10	28.77	13.14	23.45	29.00	27.00	26.00	27.33	34.75	55.33	38.79	3.97×10^8
w/o T.C.	55.00	30.19	14.98	25.01	32.00	31.00	29.00	30.67	38.63	58.67	41.60	4.08×10^8

Table 3: Ablation results on different backbones. We individually ablate the lookahead module, three value estimation scores, and the trajectory convergence (T.C.) mechanism. *w/o* denotes experiments conducted without the specified module.

Lower V_{step} reflects bounded fluctuation across future steps, indicating that the trajectory remains stable and resists erratic deviations, akin to Lyapunov-stable behavior. Accordingly, we define the step consistency reward as $R_i^{\text{step}} = \exp\left(-\frac{V_{\text{step}}}{\tau}\right)$, where τ is a temperature parameter controlling sensitivity.

Proposition 2 (Deviation Bound). *If $V_{step} \leq \varepsilon$, then $|g_n - \bar{g}| \leq \sqrt{N\varepsilon}$ for every n. Bounding V_{step} therefore constrains state fluctuations and yields Lyapunov-like stability.*

The detailed derivation can be found in Appendix A.2. This variance serves as a regularizer to favor smoother forward reasoning paths.

(3) Slope-Level Variance. Inspired by Lipschitz continuity in mathematical analysis (Heinonen, 2005), we measure the directional smoothness of the lookahead trajectory by evaluating local slope variations. We define the first-order difference $\delta_n = g_{n+1} - g_n$. The average slope over a rollout of length N is $\bar{\delta} = \frac{1}{N-1}\sum_{n=1}^{N-1}\delta_n$, and its variance is given by:

$$V_{\text{slope}} = \frac{1}{N-1}\sum_{n=1}^{N-1}(\delta_n - \bar{\delta})^2. \tag{12}$$

Lower V_{slope} implies the trajectory's local increments are uniformly bounded, resembling Lipschitz-continuous behavior that avoids abrupt changes. Accordingly, we define the slope consistency reward as $R_i^{\text{slope}} = \exp\left(-\frac{V_{\text{slope}}}{\tau}\right)$, where τ controls sensitivity to local oscillations.

Proposition 3 (Lipschitz Bound). *If $V_{slope} \leq \varepsilon$, then for all m, n we have $|g_m - g_n| \leq \sqrt{(N-1)\varepsilon}\,|m-n|$. Hence bounding V_{slope} limits its worst-case jumps and enforces Lipschitz-like smoothness.*

The detailed derivation can be found in Appendix A.3. This reward encourages the model to prefer directionally coherent forward reasoning paths.

Combining Multiple Rewards. We combine the normalized scores of advantage, consistency, and slope into a unified reward:

$$R(s_0, s_{\leq i}, s_{>i}) = (1-\alpha-\beta)\cdot \text{Norm}(R_i^{\text{adv}}) + \alpha \cdot \text{Norm}(R_i^{\text{step}}) + \beta \cdot \text{Norm}(R_i^{\text{slope}}), \tag{13}$$

where each component is temperature-scaled and normalized by $\text{Norm}(R_i) = \frac{\exp(R_i/\tau)}{\sum_j \exp(R_j/\tau)}$, with $\tau = 0.6$.

Replacing this formulation of R into Eq. 8, the objective becomes sampling from the joint distribution that captures advantage, consistency, and directional smoothness.

2.4 Trajectory Convergence

To reduce computation and improve inference efficiency, we monitor the variance of candidate rewards $R(s_0, s_{\leq i}, s_{>i})$ at each step. Once the

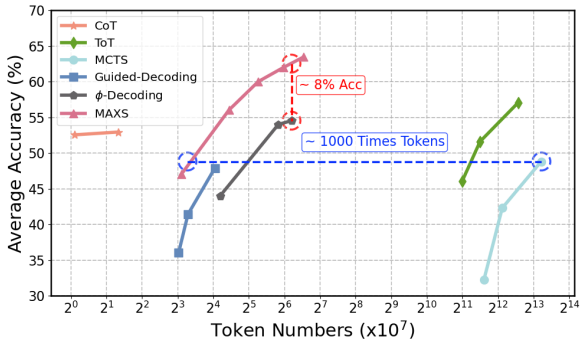

Figure 4: Inference-time scaling law: Accuracy vs. Token usage for different models during decoding.

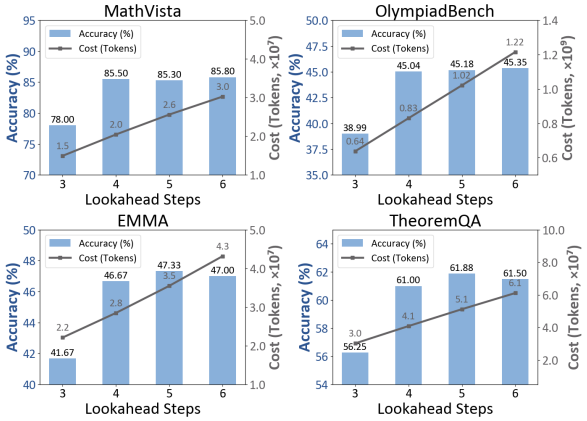

Figure 5: Accuracy–cost trade-off under varying lookahead steps across datasets.

variance falls below a threshold δ, we stop rollout and resume auto-regressive decoding. Let $\mathcal{R}_i = \{R^{(k)}(s_0, s^{(k)}_{\leq i}, s^{(k)}_{>i})\}_{k=1}^{K}$. The early stopping condition is:

$$\mathrm{Var}(\mathcal{R}_i) \leq \delta. \tag{14}$$

We terminate rollout at step i and resume decoding under the auto-regressive process. For all experiments, we set the convergence threshold $\delta = 0.002$ to balance efficiency and stability.

3 Experiments

3.1 Experimental Settings

Benchmarks. We evaluate our proposed method, MAXS, on five diverse and challenging reasoning benchmarks to assess its performance across both unimodal and multimodal domains. The selected datasets are MathVista, OlympiadBench, TheoremQA, MATH, and EMMA. More dataset details can be found in Appendix B.

Backbones and Hyperparameters. We conduct experiments using three multimodal language models: MiMo-VL-7B, Qwen2.5-VL-7B, and Qwen2.5-VL-32B, to evaluate the robustness and generalizability of MAXS across different architectures and model scales. All experiments are implemented on NVIDIA A800 GPUs with 80GB VRAM, using the vLLM (Kwon et al., 2023) inference engine. We keep the decoding configuration fixed for fair comparison, where K = 1, M = 4, and N = 4. Under this setting, the maximum step of reasoning considered is 13. The step scoring strategy is controlled by $\alpha = 0.3$ and $\beta = 0.2$, which balance different components of the score. The top-p value is set to 0.95 to ensure a good trade-off between diversity and precision in generation.

Metrics. We adopt the **pass@1** (Chen et al., 2021) rate as our primary accuracy (Acc.) metric to evaluate the correctness of the final generated answer. To measure computational efficiency, we also report the average number of **input and output tokens** consumed by the backbone model for generating each solution.

Tools. During inference, the LLM agents autonomously invoke external tools to support complex reasoning via code execution and knowledge retrieval. Specifically, a Python-based *Code Interpreter* executes model-generated code for accurate computations, while a *Search Engine* retrieves external knowledge-implemented via an LLM for convenience.

Baselines. We compare MAXS against five representative reasoning methods, including *CoT*, which generates a single step by step reasoning chain, *ToT* and *MCTS*, which explore reasoning trees with pruning via self evaluation or Monte Carlo rollouts, *Guided Decoding* (Xie et al., 2023), which uses stochastic search guided by self evaluation, and *ϕ-Decoding* (Xu et al., 2025a), which selects steps based on simulated foresight and path alignment.

3.2 Main Results

MAXS improves average performance across backbones. As shown in Table 1, MAXS consistently outperforms five strong baselines, achieving SOTA results. On MiMo-VL-7B, it reaches 63.46% accuracy-6.42% higher than ToT. On Qwen2.5-VL-7B, it surpasses Guided Decoding by 7.43%, demonstrating strong generalization.

MAXS balances effectiveness and efficiency. While tree-based methods like ToT and MCTS

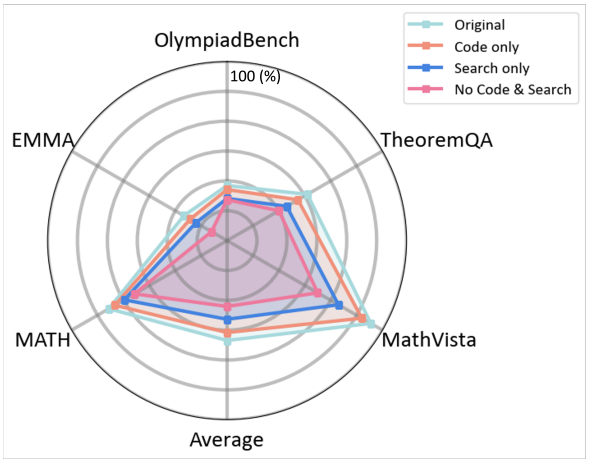

Figure 6: Radar plot of accuracy under different tool configurations across datasets.

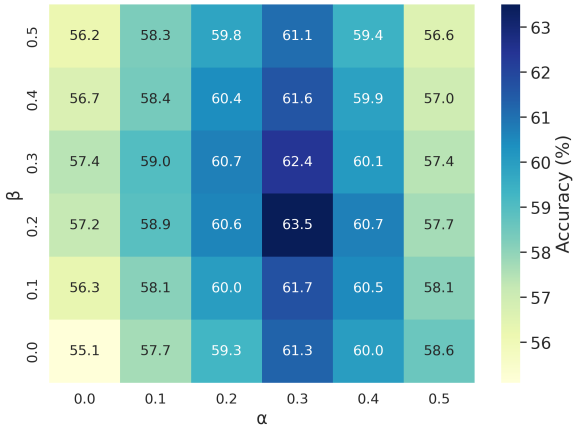

Figure 7: Accuracy heatmap under different value estimation weights (α, β) across datasets.

are competitive, they require up to 100× more tokens. On MiMo-VL-7B, MAXS uses 9.86×10^8 tokens, compared to ToT's 6.40×10^{10} and MCTS's 9.91×10^{10}. Compared to efficient methods like ϕ-Decoding, MAXS achieves notably higher accuracy with minimal additional cost, reflecting its superior allocation of computation for reasoning.

3.3 Generalization and Scalability

MAXS's superiority persists when scaling to the 32B model size. We conduct experiments on the EMMA benchmark using the Qwen2.5-VL-32B model. As shown in Table 2, MAXS yields even greater improvements on the larger model, surpassing the strongest baseline, ϕ-Decoding, by 6.33%. This confirms its ability to capitalize on the advanced reasoning potential of larger LLMs.

3.4 Inference-Time Scaling

MAXS method demonstrates a superior trade-off between performance and computational efficiency. As shown in Figure 4, MAXS consistently occupies the optimal top-left region, delivering the highest accuracy for any given token budget on the MiMo-VL-7B model. Horizontally, to achieve a comparable accuracy level of 49%, MAXS requires approximately 1,000 times fewer tokens than the MCTS baseline. Vertically, with a similar computational cost to ϕ-Decoding, MAXS achieves a higher accuracy, showcasing a performance advantage of nearly 8%.

4 Analysis

4.1 Ablation Studies

To assess the impact of each component in MAXS, we perform a systematic ablation study by removing one module at a time on MiMo-VL-7B and Qwen2.5-VL-7B. Results in Table 3 reveal the following key insights:

Lookahead is essential for globally-aware reasoning. Removing the lookahead module leads to the steepest performance drop (–4.96% on MiMo-VL, –9.44% on Qwen2.5-VL), highlighting its role in simulating future trajectories and escaping local optima. This aligns with the Bellman principle and confirms lookahead as fundamental.

Advantage score dominates value estimation. Among the three reward signals, ablating the advantage score yields the greatest degradation, proving it is the key driver of effective step selection. In contrast, step and slope variance mainly aid stability, with smaller impacts.

Trajectory convergence improves efficiency with little cost. Although its removal slightly affects accuracy, trajectory convergence reduces inference cost by terminating redundant rollouts, offering efficiency gains without sacrificing quality.

4.2 Analysis of Lookahead Steps

A 4-step lookahead offers the best balance between accuracy and efficiency. As shown in Figure 5, accuracy improves from 3 to 4 steps but plateaus at 85.3%–85.8% beyond that. Meanwhile, token usage rises sharply-from 2.05×10^7 at 4-step to 3.07×10^7 at 6-step-incurring a 49.8% over-

head. This confirms 4-step as the efficiency frontier, where further gains no longer justify the cost.

4.3 Analysis of Tool Utilization

Code and search are complementary, removing either harms performance. As shown in Figure 6, dropping code or search reduces accuracy from 63.46% (full model) to 60.81% (–2.65%) and 56.36% (–7.1%), respectively. The largest drop (52.07%, –11.4%) occurs when both are removed, underscoring their synergy in multi-tool reasoning.

Code is especially critical for symbolic reasoning. On MathVista, removing code drops accuracy from 85.5% to 73.0% (–14.7%), versus 82.0% (–4.1%) without search. While search aids information access, precise computation from code is key to correctness in complex tasks.

4.4 Analysis of Value Estimation Weights

Combining step and slope scores (α=0.3, β=0.2) yields the best overall performance. As shown in Figure 7, the model achieves peak accuracy (63.5%) when α=0.3 and β=0.2, validating the effectiveness of jointly weighting step-based and slope-based rewards in Equation 13. This configuration outperforms the advantage-only baseline (α=0, β=0, 55.2%) by +8.3%. Moreover, adjacent settings also yield competitive results, suggesting that the reward formulation is both robust and well-balanced.

4.5 Analysis of Reasoning Steps

Most problems are solved within 4–8 steps, validating the 13-step cap. As shown in Figure 8, most reasoning trajectories conclude between steps 4 and 8 across datasets. OlympiadBench peaks later at steps 7–8 (23% each), suggesting greater complexity, while MathVista, EMMA, and TheoremQA concentrate around steps 5–6, covering 58–65% of cases. Kernel density curves show OlympiadBench spans a broader range (6–9 steps), whereas others are more tightly clustered. Reasoning rarely exceeds 13 steps, justifying our choice of a 13-step cap. These trends confirm that moderate-length trajectories suffice for most problems, with deeper steps reserved for harder cases.

Appendix D provides additional analysis on rollout, beam size, value estimation methods and significance test, while Appendix E presents successful and failure cases.

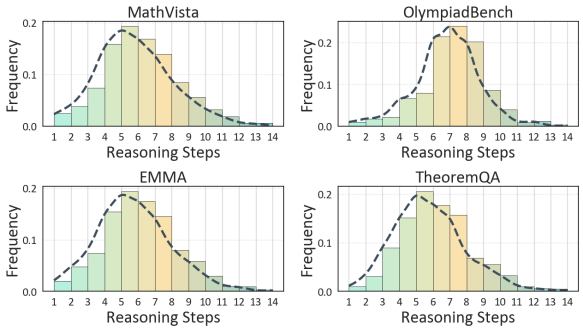

Figure 8: Distribution of reasoning steps across datasets.

5 Related Works

LLM Agents and Tool-Augmented Reasoning. LLM Agents enhance language models by dynamically invoking tools (e.g., search, code) to support complex reasoning (Renze and Guven, 2024; Yang et al., 2024; Zhang et al., 2026b,a; Chen et al., 2026). Early approaches insert API calls to improve factual accuracy (Jin et al., 2025; Wang et al., 2024; Xu et al., 2026; Li et al., 2026), while recent frameworks integrate planning and tool selection into multi-step decision-making (Baker et al., 2019; Torreno et al., 2017; Zhang et al., 2024; Fu et al., 2026; Liu et al., 2025, 2026). However, most rely on locally greedy decoding and lack long-term tool utility estimation. We address this gap via lookahead-based evaluation and stability-aware step selection.

Inference-Time Scaling and Optimization. Inference-time methods like ToT (Yao et al., 2023), MCTS (Gan et al., 2025), and Best-of-N (Gui et al., 2024) improve answer quality by exploring multiple paths, but often at high computational cost. Efficiency-focused approaches introduce early stopping (Chen et al., 2024; Yan et al., 2025) or pruning (Xu et al., 2025a; Yan et al., 2026). Our method complements them by combining lightweight value estimation with convergence-aware rollouts for efficient multi-tool reasoning.

6 Conclusion

In this work, we propose *MAXS*, a meta-adaptive exploration framework that mitigates local myopia and trajectory instability in LLM agents. MAXS integrates lookahead rollouts and a composite value function that incorporates advantage, step variance, and slope variance to guide stable, efficient decision making. A trajectory convergence mechanism further reduces redundant rollouts. Experiments on five benchmarks and three backbones

demonstrate improved reasoning performance and reduced cost, with ablations confirming the synergy between lookahead and value-based guidance.

Limitations

MAXS is an attempt to mitigate locally myopic generation and trajectory instability in LLM agents via meta-adaptive lookahead and stability aware step selection. The advantage based value estimation with step and slope consistency signals and the trajectory convergence mechanism achieve a strong effectiveness efficiency trade off. However, we acknowledge two limitations: (1) MAXS relies on reasonably reliable tool execution such as search and code, and when available tools are weak or poorly aligned with the task, advantage estimation and the resulting trajectories may degrade ,thereby limiting overall performance. (2) MAXS relies on internal rollout based estimates without incorporating environment feedback or outcome level signals during inference, which may constrain robustness in more interactive or long horizon settings.

Acknowledgments

This work was supported by Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (JYB2025XDXM116), National Natural Science Foundation of China (No. 62137002, 62293553, 62293554, 62450005, 62437002, 62577043), the Youth Innovation Team of Shaanxi Universities "Multi-modal Data Mining and Fusion".

References

Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn Powell, Bob McGrew, and Igor Mordatch. 2019. Emergent tool use from multi-agent autocurricula. In *International conference on learning representations*.

EN Barron and H Ishii. 1989. The bellman equation for minimizing the maximum cost. *Nonlinear Anal. Theory Methods Applic.*, 13(9):1067–1090.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Wenhu Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony Xia. 2023. Theoremqa: A theorem-driven question answering dataset. *arXiv preprint arXiv:2305.12524*.

Yanxi Chen, Xuchen Pan, Yaliang Li, Bolin Ding, and Jingren Zhou. 2024. Ee-llm: Large-scale training and inference of early-exit large language models with 3d parallelism. In *International Conference on Machine Learning*, pages 7163–7189. PMLR.

Zhihui Chen, Kai He, Qingyuan Lei, Bin Pu, Jian Zhang, Yuling Xu, and Mengling Feng. 2026. Medforge: Interpretable medical deepfake detection via forgery-aware reasoning. *arXiv preprint arXiv:2603.18577*.

Wonje Choi, Woo Kyung Kim, Minjong Yoo, and Honguk Woo. 2024. Embodied cot distillation from llm to off-the-shelf agents. In *Proceedings of the 41st International Conference on Machine Learning*, pages 8702–8721.

Yumeng Fu, Jiayin Zhu, Lingling Zhang, Bo Zhao, Shaoxuan Ma, Yushun Zhang, Yanrui Wu, and Wenjun Wu. 2026. Geolaux: A benchmark for evaluating mllms' geometry performance on long-step problems requiring auxiliary lines. *The 64rd Annual Meeting of the Association for Computational Linguistics*.

Bingzheng Gan, Yufan Zhao, Tianyi Zhang, Jing Huang, Li Yusu, Shu Xian Teo, Changwang Zhang, and Wei Shi. 2025. Master: A multi-agent system with llm specialized mcts. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 9409–9426.

Lin Gui, Cristina Gârbacea, and Victor Veitch. 2024. Bonbon alignment for large language models and the sweetness of best-of-n sampling. *Advances in Neural Information Processing Systems*, 37:2851–2885.

Fatemeh Haji, Mazal Bethany, Maryam Tabar, Jason Chiang, Anthony Rios, and Peyman Najafirad. 2024. Improving llm reasoning with multi-agent tree-of-thought validator agent. *arXiv preprint arXiv:2409.11527*.

Yunzhuo Hao, Jiawei Gu, Huichen Will Wang, Linjie Li, Zhengyuan Yang, Lijuan Wang, and Yu Cheng. 2025. Can mllms reason in multimodality? emma: An enhanced multimodal reasoning benchmark. *arXiv preprint arXiv:2501.05444*.

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*.

Juha Heinonen. 2005. *Lectures on Lipschitz analysis*. 100. University of Jyväskylä.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Xu Huang, Weiwen Liu, Xiaolong Chen, Xingmei Wang, Hao Wang, Defu Lian, Yasheng Wang, Ruiming Tang, and Enhong Chen. 2024. Understanding the planning of llm agents: A survey. *arXiv preprint arXiv:2402.02716*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626.

Yifei Li, Weidong Guo, Lingling Zhang, Rongman Xu, Muye Huang, Hui Liu, Lijiao Xu, Yu Xu, and Jun Liu. 2026. Locomo-plus: Beyond-factual cognitive memory evaluation framework for llm agents. *arXiv preprint arXiv:2602.10715*.

Wenjin Liu, Haoran Luo, Xin Feng, Xiang Ji, Lijuan Zhou, Rui Mao, Jiapu Wang, Shirui Pan, and Erik Cambria. 2025. Lexgenius: An expert-level benchmark for large language models in legal general intelligence. *arXiv preprint arXiv:2512.04578*.

Wenjin Liu, Haoran Luo, Xueyuan Lin, Haoming Liu, Tiesunlong Shen, Jiapu Wang, Rui Mao, and Erik Cambria. 2026. Prompt-r1: Collaborative automatic prompting framework via end-to-end reinforcement learning. *arXiv preprint arXiv:2511.01016*.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. 2023. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*.

Haoran Luo, Yikai Guo, Qika Lin, Xiaobao Wu, Xinyu Mu, Wenhao Liu, Meina Song, Yifan Zhu, Luu Anh Tuan, et al. 2025. Kbqa-o1: Agentic knowledge base question answering with monte carlo tree search. *arXiv preprint arXiv:2501.18922*.

Guangtao Nie, Rong Zhi, Xiaofan Yan, Yufan Du, Xiangyang Zhang, Jianwei Chen, Mi Zhou, Hongshen Chen, Tianhao Li, Ziguang Cheng, et al. 2024. A hybrid multi-agent conversational recommender system with llm and search engine in e-commerce. In *Proceedings of the 18th ACM Conference on Recommender Systems*, pages 745–747.

Matthew Renze and Erhan Guven. 2024. Self-reflection in llm agents: Effects on problem-solving performance. *arXiv preprint arXiv:2405.06682*.

Daniel Shevitz and Brad Paden. 2002. Lyapunov stability theory of nonsmooth systems. *IEEE Transactions on automatic control*, 39(9):1910–1914.

Alejandro Torreno, Eva Onaindia, Antonín Komenda, and Michal Štolba. 2017. Cooperative multi-agent planning: A survey. *ACM Computing Surveys (CSUR)*, 50(6):1–32.

Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. 2024. Executable code actions elicit better llm agents. In *Forty-first International Conference on Machine Learning*.

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.

Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, James Xu Zhao, Min-Yen Kan, Junxian He, and Michael Xie. 2023. Self-evaluation guided beam search for reasoning. In *Advances in Neural Information Processing Systems*, volume 36, pages 41618–41650. Curran Associates, Inc.

Fangzhi Xu, Hang Yan, Chang Ma, Haiteng Zhao, Jun Liu, Qika Lin, and Zhiyong Wu. 2025a. ϕ-decoding: Adaptive foresight sampling for balanced inference-time exploration and exploitation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13214–13227, Vienna, Austria. Association for Computational Linguistics.

Fangzhi Xu, Hang Yan, Qiushi Sun, Jinyang Wu, Zixian Huang, Muye Huang, Jingyang Gong, Zichen Ding, Kanzhi Cheng, Yian Wang, et al. 2026. Odysseyarena: Benchmarking large language models for long-horizon, active and inductive interactions. *arXiv preprint arXiv:2602.05843*.

Jin Xu, Zhifang Guo, Jinzheng He, Hangrui Hu, Ting He, Shuai Bai, Keqin Chen, Jialin Wang, Yang Fan, Kai Dang, et al. 2025b. Qwen2. 5-omni technical report. *arXiv preprint arXiv:2503.20215*.

Hang Yan, Xinyu Che, Fangzhi Xu, Qiushi Sun, Zichen Ding, Kanzhi Cheng, Jian Zhang, Tao Qin, Jun Liu, and Qika Lin. 2026. Tide: Trajectory-based diagnostic evaluation of test-time improvement in llm agents. *arXiv preprint arXiv:2602.02196*.

Hang Yan, Fangzhi Xu, Rongman Xu, Yifei Li, Jian Zhang, Haoran Luo, Xiaobao Wu, Luu Anh Tuan, Haiteng Zhao, Qika Lin, et al. 2025. Mur: Momentum uncertainty guided reasoning for large language models. *arXiv preprint arXiv:2507.14958*.

Hang Yang, Hao Chen, Hui Guo, Yineng Chen, Ching-Sheng Lin, Shu Hu, Jinrong Hu, Xi Wu, and Xin Wang. 2024. Llm-medqa: Enhancing medical question answering through case studies in large language models. *arXiv preprint arXiv:2501.05464*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Li Yuan, Yi Cai, Xudong Shen, Qing Li, Qingbao Huang, Zikun Deng, and Tao Wang. 2025. Collaborative multi-lora experts with achievement-based multi-tasks loss for unified multimodal information extraction. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, pages 6940–6948.

Li Yuan, Qingfei Huang, Bingshan Zhu, Yi Cai, Qingbao Huang, Changmeng Zheng, Zikun Deng, and Tao Wang. 2026. Hybrid-dmkg: A hybrid reasoning framework over dynamic multimodal knowledge graphs for multimodal multihop qa with knowledge editing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 28032–28040.

Zihao Yue, Zhenru Lin, Yifan Song, Weikun Wang, Shuhuai Ren, Shuhao Gu, Shicheng Li, Peidian Li, Liang Zhao, Lei Li, et al. 2025. Mimo-vl technical report. *CoRR*.

Jian Zhang, Shihao Qi, Yuxuan Dong, Li Yuan, Tiesunlong Shen, Weiping Fu, Bifan Wei, Haiping Zhu, and Jun Liu. 2025. Gkg-llm: A unified framework for generalized knowledge graph construction. *Information Fusion*, page 103956.

Jian Zhang, Zhangqi Wang, Haiping Zhu, Kangda Cheng, Kai He, Bo Li, Qika Lin, Jun Liu, and Erik Cambria. 2026a. Mars: Multi-agent adaptive reasoning with socratic guidance for automated prompt optimization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(19):16307–16315.

Jian Zhang, Zhiyuan Wang, Zhangqi Wang, Fangzhi Xu, Qika Lin, Lingling Zhang, Rui Mao, Erik Cambria, and Jun Liu. 2026b. Maps: Multi-agent personality shaping for collaborative reasoning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(19):16316–16324.

Zeyu Zhang, Quanyu Dai, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. 2024. A survey on the memory mechanism of large language model based agents. *ACM Transactions on Information Systems*.

A Proof of Proposition

A.1 Proof of Proposition 1: Bellman Recursion

We aim to prove that the optimal decision at step i satisfies:

$$\hat{s}_i = \arg\max_{s_i} \left[R(s_i) + \gamma \, \mathbb{E}_{s_{>i}} V^*(s_{>i}) \right], \quad (15)$$

where $R(s_i)$ is the immediate utility, $\gamma \in (0, 1)$ is a discount factor, and $V^*(s_{>i})$ is the expected future value under the optimal policy.

Step 1: Define global optimal value. Let the total expected return under the optimal policy starting from the initial input s_0 be:

$$V^*(s_0) = \max_{s_1,\ldots,s_T} \mathbb{E}\left[\sum_{t=1}^{T} \gamma^{t-1} R(s_t) \right]. \quad (16)$$

We can rewrite this recursively as:

$$V^*(s_0) = \max_{s_1} \left[R(s_1) + \gamma \cdot \mathbb{E}_{s_2} V^*(s_{\geq 2}) \right]. \quad (17)$$

Step 2: Bellman decomposition at step i. At an arbitrary step i, given history $s_0, \ldots, s_{i-1}$, the value function is:

$$V^*(s_{\leq i}) = \max_{s_{>i}} \mathbb{E}\Big[\sum_{k=1}^{K} \gamma^{k-1} R(s_{i+k}) \;\Big|\; s_{\leq i} \Big], \quad (18)$$

which can again be written recursively as:

$$V^*(s_{\leq i}) = \max_{s_{i+1}} \Big[R(s_{i+1}) + \gamma \, \mathbb{E}_{s_{>i+1}} V^*(s_{>i+1}) \Big]. \quad (19)$$

Step 3: Local decision refinement. Now consider choosing s_i to maximize the full downstream return:

$$\hat{s}_i = \arg\max_{s_i} \mathbb{E}_{s_{>i}} \Big[R(s_i) + \sum_{k=1}^{K} \gamma^k R(s_{i+k}) \Big]. \quad (20)$$

Let us define:

$$Q(s_i) := R(s_i) + \gamma \cdot \mathbb{E}_{s_{>i}} V^*(s_{>i}), \quad (21)$$

then

$$\hat{s}_i = \arg\max_{s_i} Q(s_i). \quad (22)$$

Step 4: Relation to lookahead rollout. In rollout-based approximation, we generate a set of candidate continuations $\{s_{>i}^{(k)}\}_{k=1}^{M}$, then use Monte Carlo estimate:

$$\mathbb{E}_{s_{>i}} V^*(s_{>i}) \approx \frac{1}{M} \sum_{k=1}^{M} \sum_{j=1}^{K} \gamma^{j-1} R(s_{i+j}^{(k)}), \quad (23)$$

which retains consistency with the Bellman optimal formulation.

Conclusion. Thus, our decision strategy:

$$\hat{s}_i = \arg\max_{s_i} \left[R(s_i) + \gamma \cdot \mathbb{E}_{s_{>i}} V^*(s_{>i}) \right] \quad (24)$$

recursively links current utility with foresighted trajectory values, consistent with Bellman's Principle of Optimality.

A.2 Proof of Proposition 2: Deviation Bound

We aim to show that if the step-level variance of a rollout trajectory is bounded by ε, then each individual log-probability score g_n is tightly concentrated around its mean $\bar{g}$:

$$V_{\text{step}} \leq \varepsilon \quad \Rightarrow \quad |g_n - \bar{g}| \leq \sqrt{N\varepsilon}, \quad (25)$$

$\forall n \in \{1, \ldots, N\}$.

Step 1: Definition of variance. By definition, the step-level variance of the rollout is:

$$V_{\text{step}} = \frac{1}{N} \sum_{n=1}^{N} (g_n - \bar{g})^2. \quad (26)$$

This measures the dispersion of log-probabilities across the trajectory.

Step 2: Bounding the ℓ_2 norm. Let $\delta_n := g_n - \bar{g}$ be the deviation from the mean at step n. Then:

$$\sum_{n=1}^{N} \delta_n^2 = N \cdot V_{\text{step}} \leq N\varepsilon. \quad (27)$$

This implies the squared ℓ_2 norm of the deviation vector $\boldsymbol{\delta} = [\delta_1, \ldots, \delta_N]$ is bounded.

Step 3: Derive pointwise bound via inequality. Using the fact that:

$$\|\boldsymbol{\delta}\|^2 = \sum_{n=1}^{N} \delta_n^2 \geq \max_n \delta_n^2, \quad (28)$$

it follows that for each n:

$$|g_n - \bar{g}| = |\delta_n| \leq \|\boldsymbol{\delta}\| \leq \sqrt{N\varepsilon}. \quad (29)$$

Step 4: Alternative probabilistic interpretation. Suppose the log-probability sequence $\{g_n\}$ arises from a bounded stochastic process. Then $\bar{g}$ is the empirical mean, and by applying Chebyshev's inequality:

$$\mathbb{P}(|g_n - \bar{g}| \geq \lambda) \leq \frac{V_{\text{step}}}{\lambda^2} \leq \frac{\varepsilon}{\lambda^2}, \tag{30}$$

which shows that the deviation from the mean is highly improbable beyond scale $\sqrt{\varepsilon}$.

Step 5: Connection to discrete Lyapunov stability. The result implies that the rollout trajectory is uniformly bounded within a $\sqrt{N\varepsilon}$-ball around the mean, which is a sufficient condition for bounded-input bounded-state (BIBS) stability in discrete-time systems. That is, $\forall g_n, \quad |g_n - \bar{g}| \leq \mathcal{O}(\sqrt{N\varepsilon}) \quad \Rightarrow \quad$ bounded trajectory.

Conclusion. The variance bound implies that the trajectory exhibits global uniform boundedness, which is analogous to Lyapunov stability in dynamical systems. This supports the interpretation that minimizing V_{step} leads to smoother and more predictable reasoning behavior.

A.3 Proof of Proposition 3: Lipschitz Bound

We aim to show that if the slope-level variance of the log-probability sequence $\{g_n\}_{n=1}^N$ is bounded by ε, then for any two positions $m, n \in \{1, \ldots, N\}$, their cumulative difference is linearly bounded in $|m - n|$:

$$\begin{aligned} &V_{\text{slope}} \leq \varepsilon \\ &\Rightarrow \quad |g_m - g_n| \leq \sqrt{(N-1)\varepsilon}\,|m - n|. \end{aligned} \tag{31}$$

Step 1: Define local slope sequence. Let $\delta_n := g_{n+1} - g_n$ be the first-order discrete derivative (slope) between adjacent log-probability values:

$$\delta_n = g_{n+1} - g_n, \quad \text{for } n = 1, \ldots, N-1. \tag{32}$$

Let the average slope be:

$$\bar{\delta} = \frac{1}{N-1} \sum_{n=1}^{N-1} \delta_n. \tag{33}$$

Step 2: Define slope-level variance. The slope variance is defined as:

$$V_{\text{slope}} = \frac{1}{N-1} \sum_{n=1}^{N-1} (\delta_n - \bar{\delta})^2. \tag{34}$$

This measures the local fluctuation in directional progress. Let $\Delta_n := \delta_n - \bar{\delta}$ denote the deviation from average slope.

Then,

$$\sum_{n=1}^{N-1} \Delta_n^2 = (N-1) \cdot V_{\text{slope}} \leq (N-1)\varepsilon. \tag{35}$$

Step 3: Express global difference via telescoping sum. Let $m < n$ without loss of generality. Then we have:

$$g_n - g_m = \sum_{k=m}^{n-1} \delta_k = (n-m)\bar{\delta} + \sum_{k=m}^{n-1} \Delta_k. \tag{36}$$

The first term captures the trend, and the second term reflects local irregularity.

Step 4: Bound the deviation term. By Cauchy–Schwarz inequality:

$$\begin{aligned} \left| \sum_{k=m}^{n-1} \Delta_k \right|^2 &\leq (n-m) \cdot \sum_{k=m}^{n-1} \Delta_k^2 && (37) \\ &\leq (n-m) \cdot \sum_{k=1}^{N-1} \Delta_k^2 && (38) \\ &\leq (n-m)(N-1)\varepsilon. && (39) \end{aligned}$$

Hence,

$$\left| \sum_{k=m}^{n-1} \Delta_k \right| \leq \sqrt{(n-m)(N-1)\varepsilon}. \tag{40}$$

Step 5: Final bound on log-probability difference. From Eq. (36), we have:

$$\begin{aligned} |g_n - g_m| \leq |n-m|\,|\bar{\delta}| \\ + \sqrt{(n-m)(N-1)\varepsilon}. \end{aligned} \tag{41}$$

In worst-case or centered-slope settings (e.g., $\bar{\delta} \approx 0$), the term simplifies to:

$$|g_n - g_m| \leq \sqrt{(N-1)\varepsilon} \cdot |n-m|, \tag{42}$$

which mimics the discrete Lipschitz condition with constant $\sqrt{(N-1)\varepsilon}$.

Step 6: Discrete Lipschitz analogy. A function $f(x)$ is Lipschitz continuous if:

$$|f(x) - f(y)| \leq L|x - y|, \quad \forall x, y. \tag{43}$$

Here, the sequence $\{g_n\}$ exhibits analogous behavior, where the bounded variance on discrete slopes constrains global oscillation across the trajectory.

Dataset	Category	Size
MathVista	Overall	1000
OlympiadBench	OE_TO_maths_zh_CEE	1240
	OE_MM_maths_zh_CEE	1910
	OE_TO_physics_en_COMP	236
	OE_MM_maths_en_COMP	150
	OE_MM_physics_en_COMP	456
	OE_TO_maths_en_COMP	674
	OE_TO_maths_zh_COMP	408
	OE_MM_physics_zh_CEE	1483
	OE_MM_maths_zh_COMP	56
	OE_TO_physics_zh_CEE	115
	maths (subset total)	4438
	physics (subset total)	2290
	Overall	6728
EMMA	Math	100
	Physics	100
	Chemistry	100
	Overall	300
TheoremQA	Overall	800
MATH	Sampled	300

Table 4: Detailed composition of the five datasets used in our study: MathVista, OlympiadBench, EMMA, TheoremQA, and MATH. For OlympiadBench, we present its fine-grained subsets along with their corresponding sizes. We also report the total number of problems in the math- and physics-related subsets, where applicable. For EMMA, we adopt its MINI version, and for MATH, we sample 300 problems from the full dataset.

Conclusion. The slope variance V_{slope} directly governs the rate of directional fluctuation. Bounding it enforces path regularity, controls local curvature, and promotes globally smooth reasoning progress. This justifies the slope-consistency reward in our value function as a surrogate for discrete Lipschitz continuity.

B Datasets

As illustrated in Table 4, this study utilizes five publicly available datasets: *MathVista*, *OlympiadBench*, *EMMA*, *TheoremQA*, and *MATH*. These benchmarks cover a wide range of science problems and are widely used for evaluating reasoning abilities of large language models.

MathVista. MathVista is a large-scale scientific reasoning dataset that spans multiple reasoning types such as algebraic, geometric, statistical, scientific, numeric commonsense, and logical reasoning, aiming to assess the comprehensive capabilities of machine learning models in solving complex scientific problems. The dataset(*testmini*) contains 1,000 data points covering various issues across multiple disciplines, designed with varying difficulty levels to help researchers evaluate model reasoning abilities. The release of MathVista supports interdisciplinary scientific research.

OlympiadBench. OlympiadBench consists of two subdomains, maths and physics, and is specifically designed for Mathematical and Physical Olympiads, featuring a wide range of challenging problems to assess models' performance on high-level scientific tasks. The dataset contains two difficulty levels: competition level and college level, reflecting the diversity and depth of real-world Olympiad problems. It includes two types of questions: open-ended questions and theorem-proof questions. To focus on evaluating generative mathematical reasoning abilities, we select the 6,728 open-ended(OE) questions for our experiments.

EMMA. EMMA is a multimodal scientific reasoning dataset covering three subsets: Math, Physics, and Chemistry. By integrating mathematical expressions, physical formulas, and chemical symbols with natural language descriptions, it focuses on testing models' abilities in interdisciplinary scientific reasoning. This version uses the EMMA dataset, which contains 100 data points from each subdomain (mathematics, physics, and chemistry).

TheoremQA. TheoremQA is a benchmark dataset designed to evaluate the ability of language models to perform theorem-based reasoning. It contains 800 high-quality question-answer pairs grounded in over 350 unique theorems, covering fields such as mathematics, physics, electrical engineering, computer science, and finance. The dataset focuses on assessing whether models can correctly apply formal theorems to solve advanced problems, making it a valuable resource for studying scientific reasoning in large language models.

MATH. MATH is a benchmark dataset designed to evaluate the advanced mathematical reasoning capabilities of language models. It comprises 12,500 high school competition-level problems drawn from sources such as AMC, AIME, and other standardized exams. The dataset spans seven mathematical domains: Prealgebra, Algebra, Number Theory, Counting & Probability, Geometry, Intermediate Algebra, and Precalculus. Each prob-

Algorithm 1 MAXS Decoding with Lookahead and Value Estimation

Input: Input prompt s_0

Parameter: Model π_θ, beam size K, temperature τ, threshold δ, rollout size M, lookahead size N

Output: Final reasoning trajectory $s = \{s_1, \ldots, s_T\}$

Initialize $t \leftarrow 1$, $s \leftarrow \{s_0\}$
while not end-of-sequence **do**
  Sample K candidates $\{s_t^{(m)}\}_{m=1}^{M} \sim \pi_\theta(s_t \mid s_{<t})$
  for each candidate $s_t^{(m)}$ **do**
    Rollout $s_{>t}^{(m)} \sim \pi_\theta$ up to length N
    Compute foresight $F_t^{(k)} = \pi_\theta(s_{>t}^{(k)} \mid s_{\leq t}^{(k)})$
    Compute advantage R_t^{adv}, step variance R_t^{step}, slope variance R_t^{slope}
    Aggregate reward $R^{(k)}$ via Eq. (13)
  end for
  if $\text{Var}(\{R^{(k)}\}) \leq \delta$ **then**
    Break rollout, continue auto-regressive decoding
  end if
  Select $\hat{s}_t \sim \text{softmax}(R^{(k)}/\tau)$
  Append $\hat{s}_t$ to s, update $t \leftarrow t + 1$
end while
return sequence s

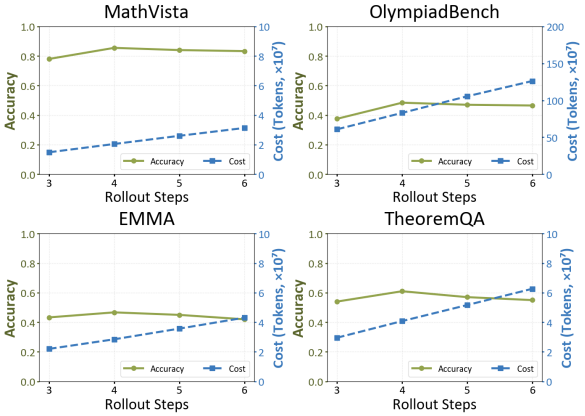

Figure 9: Accuracy–cost trade-off under varying rollout steps across datasets.

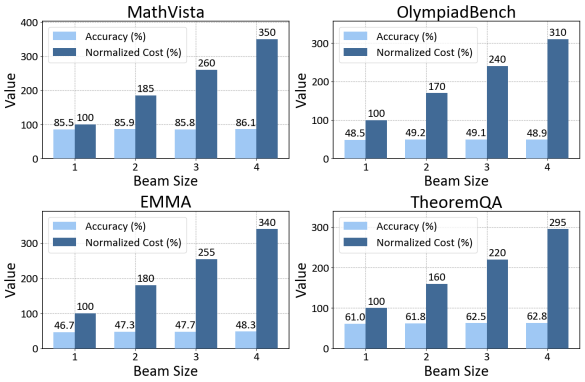

Figure 10: Accuracy vs. relative cost under varying beam sizes (1-beam normalized to 100%).

lem includes a detailed step-by-step solution, final answer, subject label, and difficulty rating, allowing for fine-grained analysis of model performance across diverse mathematical topics. We randomly sampled 300 problems from the MATH dataset, selecting 60 problems from each difficulty level (Levels 1 through 5) to ensure an evenly balanced coverage across difficulty tiers.

C MAXS Decoding Algorithm

We summarize the full decoding process in Algorithm 1. At each step t, the model samples K candidate actions $\{s_t^{(k)}\}_{k=1}^{K}$ from the policy π_θ. For each candidate, a stochastic rollout generates future steps $s_{>t}^{(k)}$, from which the foresight probability $F_t^{(k)}$ is estimated.

We compute the composite reward $R^{(k)}$ using advantage score, step-level variance, and slope-level variance, combined via Eq. (13). If the reward variance $\text{Var}(\{R^{(k)}\})$ falls below threshold δ, we terminate rollout early and resume auto-regressive decoding. Otherwise, the next step $\hat{s}_t$ is sampled according to $\text{softmax}(R^{(k)}/\tau)$ and appended to the sequence. This process iterates until an end-of-sequence token is generated.

D Supplement Analysis

D.1 Analysis of Rollout Steps

Rollout steps beyond 4 incur excessive cost with no accuracy gain. As shown in Figure 9, accuracy on OlympiadBench improves from 0.375 to 0.484 when increasing the rollout steps from 3 to 4, but declines thereafter. Meanwhile, token cost rises sharply-from 332M at 3-step to 564M at 5-step and 661M at 6-step. This confirms 4-step as the efficiency frontier, where further rollout yields diminishing or even negative returns.

D.2 Analysis of Beam Size

1-beam strikes the best balance between accuracy and cost. Figure 10 shows that 1-beam maintains normalized computational cost at 100% (leftmost dark blue bars). Increasing to 4-beam

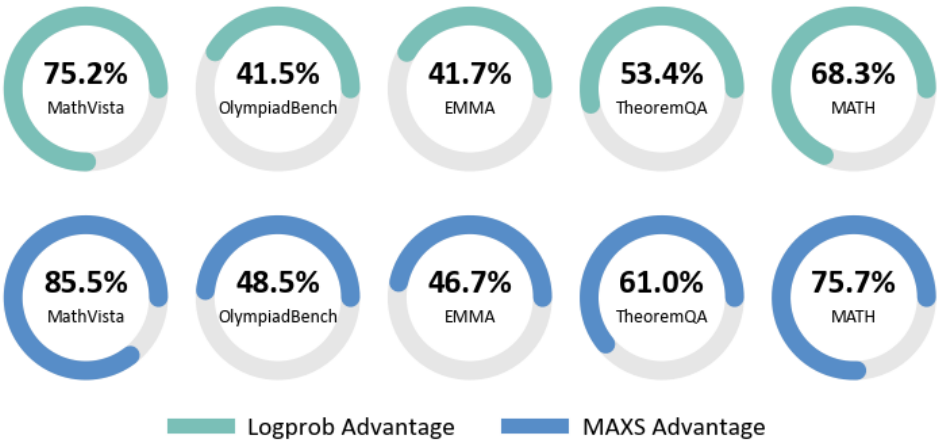

Figure 11: Comparison of different value estimation methods across datasets.

dramatically raises costs-by +250% on MathVista, +195% on TheoremQA, and +180% on EMMA-while accuracy gains remain marginal ($< 1.5\%$). On OlympiadBench, accuracy rises by only 0.46% despite a 210% cost increase. These results confirm that larger beams yield diminishing returns, with 1-beam offering the most efficient trade-off.

D.3 Comparison of Value Estimation Methods

MAXS consistently outperforms Logprob–based value estimation. As shown in Figure 11, MAXS achieves 5.0–10.3% higher accuracy across all five reasoning benchmarks, with the largest gains observed on MathVista and TheoremQA. This confirms our value estimation method's superiority in modeling complex reasoning trajectories, especially in symbolic tasks where log-probability fails to capture structural value. The stable margin of 5.0–7.3% on OlympiadBench, EMMA, and MATH further demonstrates MAXS's robustness across diverse reasoning formats.

D.4 Significance Test

To determine whether the gains achieved by MAXS are statistically significant, we perform McNemar's test for paired comparisons between MAXS and each baseline method. Table 5 reports the results on two backbones, MiMo-VL-7B-SFT and Qwen2.5-VL-7B-Instruct. Across all comparisons, including strong baselines such as ToT and ϕ-Decoding, MAXS achieves $p < 0.001$, which is well below the significance threshold $\alpha = 0.05$. These results indicate that the improvements of MAXS over existing decoding strategies are statistically significant and consistent across model architectures.

Comparison	p-value	Significance
MiMo-VL-7B-SFT		
MAXS vs. CoT	< 0.001	✓
MAXS vs. ToT	< 0.001	✓
MAXS vs. MCTS	< 0.001	✓
MAXS vs. Guided Decoding	< 0.001	✓
MAXS vs. ϕ-Decoding	< 0.001	✓
Qwen2.5-VL-7B-Instruct		
MAXS vs. CoT	< 0.001	✓
MAXS vs. ToT	< 0.001	✓
MAXS vs. MCTS	< 0.001	✓
MAXS vs. Guided Decoding	< 0.001	✓
MAXS vs. ϕ-Decoding	< 0.001	✓

Table 5: Results of McNemar's Test for Statistical Significance. We compare our proposed MAXS method against all baseline methods across two base models. A p-value < 0.05 indicates a statistically significant difference. As shown, MAXS demonstrates significant improvement over all baselines.

E Case Study

In this section, we present a successful case (Figure 12) and a failure case (Figure 13), respectively.

E.1 Successful Case

Figure 12 presents an example of problem-solving using the MAXS method, with the question sourced from the TheoremQA dataset. As shown in steps 2 and 3, MAXS performs a rollout at each reasoning step, exploring multiple candidate reasoning paths. After generating beam candidates, the model conducts foresight for each path. Although the foresight depth is set to 4, in later stages of the reasoning process, the solution may be completed within fewer than four steps-thus not every step features a full four-step foresight chain. Following this,

MAXS evaluates each rollout plus foresight chain using the three advantage metrics proposed in this paper (Advantage Score, Step-Level Variance, and Slope-Level Variance) and selects the candidate with the highest overall score as the action for the current step. This process continues iteratively until the final solution is reached. Notably, each candidate or foresight step may involve different types of operations such as reasoning, search, or code execution. The model dynamically invokes external tools to ensure high-quality reasoning throughout the problem-solving process.

E.2 Failure Case

Figure 13 presents a failure case of MAXS on MathVista, illustrating how an early recognition error can derail multi-step reasoning. The task asks for the age difference between two individuals shown in an image. At the initial stage (Meta step 0), MAXS performs a rollout and generates two beam candidates. Beam 1 attempts to use the search tool to identify the individuals, but the returned results are ambiguous and do not yield a reliable match, leading to low confidence and a lower evaluation score (-0.205). Beam 2 instead relies on the model's internal visual recognition. Although it misidentifies the individuals as Rex Tillerson and Tânia Sągescu, it produces a coherent explanation and receives a higher score (-0.123). MAXS therefore selects Beam 2 and commits to an incorrect premise.

This initial mistake propagates through later steps. In Meta steps 1-3, the model retrieves birth information for the misidentified subjects and performs the arithmetic correctly, but the final answer is necessarily wrong: it outputs 15 years instead of the ground-truth 7 years. This case highlights a limitation of the system: when tool-based retrieval is uncertain or ineffective, the model may prefer a more confident but incorrect internal hypothesis, which can dominate the downstream reasoning process.

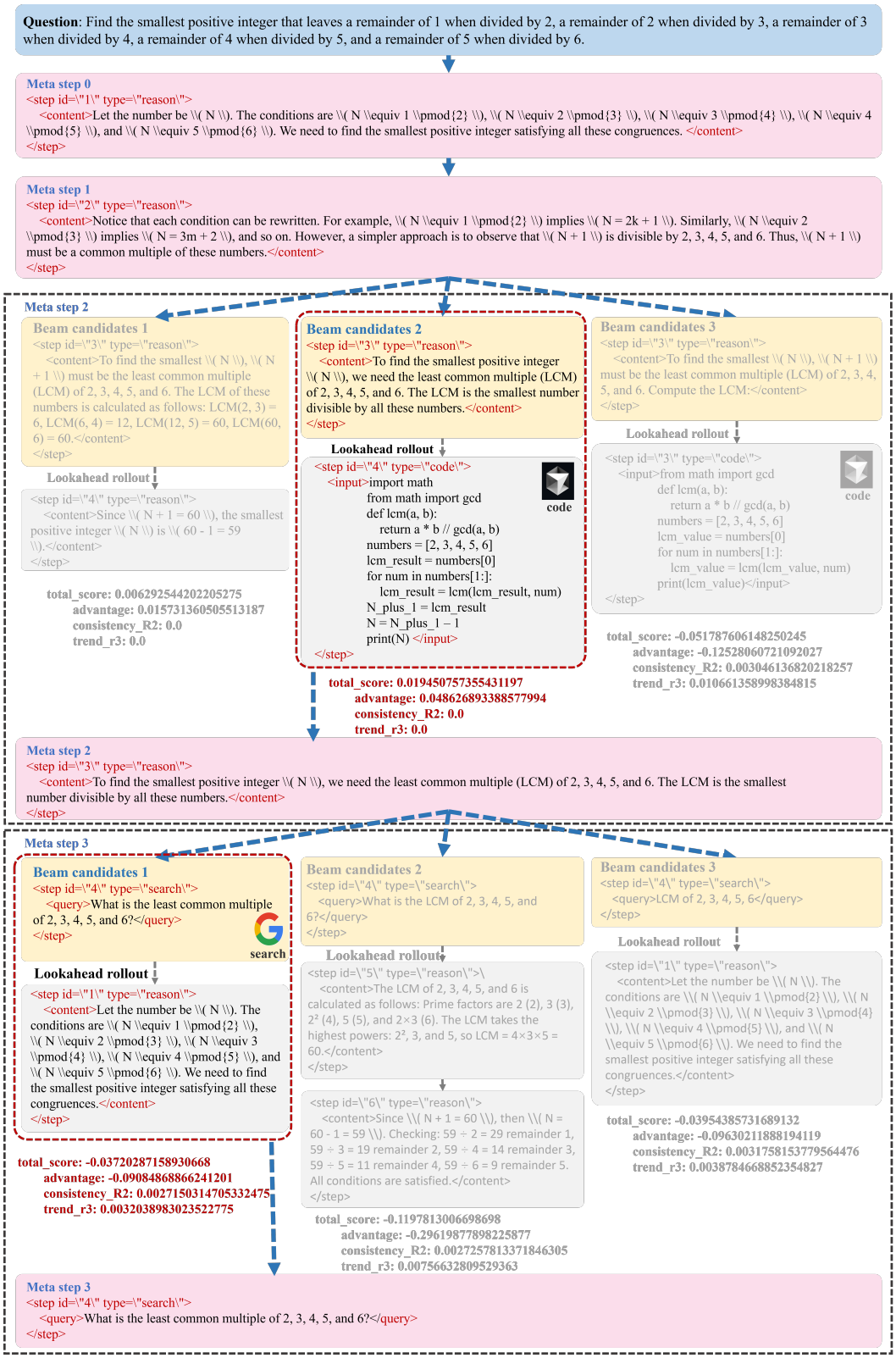

Figure 12: Successful case of MAXS solving a TheoremQA problem. At each step, it performs rollout and foresight (up to four steps), evaluates candidates via three advantage metrics, and iteratively selects the best path. The process dynamically integrates reasoning, search, and tool use.

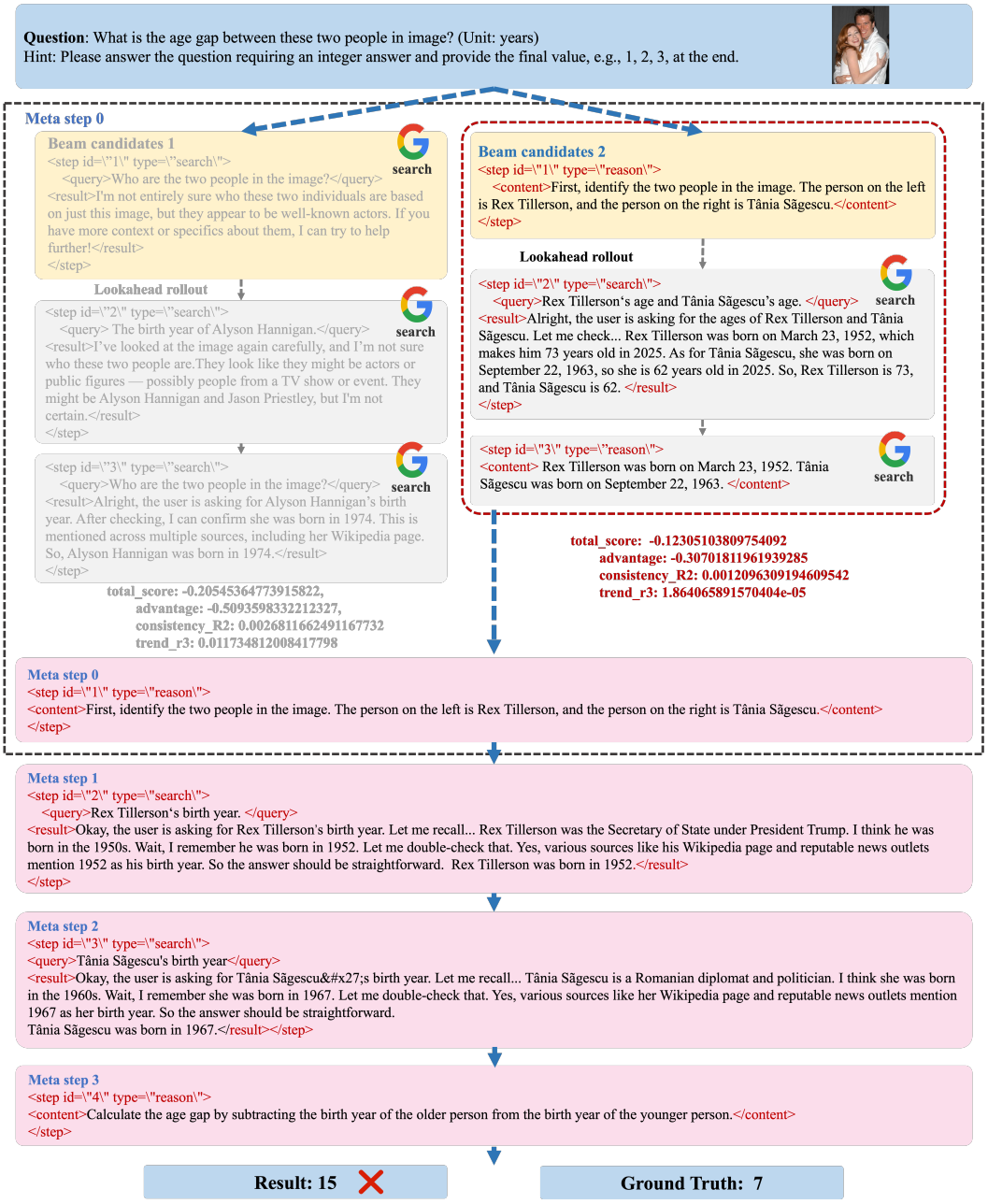

Figure 13: A failure case on the MathVista dataset where MAXS selects an incorrect visual recognition path due to the low confidence of search tool results. The initial misidentification of the individuals propagates through the reasoning chain, leading to an erroneous final answer despite valid subsequent calculations.