Stepwise Informativeness Search for Improving LLM Reasoning

Siyuan Wang[1*], **Enda Zhao**[2*], **Xiang Ren**[1]
[1]University of Southern California, [2]Purdue University.
sw_641@usc.edu; zhao1462@purdue.edu

Abstract

Advances in Large Language Models (LLMs) have improved multi-step reasoning by generating free-text rationales, but these models tend to lose focus over the middle of long contexts. This raises concerns that as reasoning progresses, LLMs may overlook information in earlier steps when decoding subsequent steps, leading to unreliable and redundant rationales. To address this, we propose guiding LLMs to generate more accurate and concise rationales by (1) proactively referencing information from underutilized prior steps, and (2) minimizing redundant information between new and existing steps. We introduce *stepwise informativeness search*, an inference-time tree search framework incorporating two selection heuristics: grounding-guided selection which prioritizes steps paying higher attention over underutilized steps; and novelty-guided selection which encourages steps with novel conclusions. We further utilize a self-grounding strategy that prompts LLMs to explicitly reference relevant prior steps as premises before deduction at each step, mitigating distraction from irrelevant content. Experiments on five reasoning datasets across five LLMs show the effectiveness and efficiency of our approach to improve reasoning with reduced errors and redundancy [1].

1 Introduction

Large Language Models (LLMs) (OpenAI, 2023; Team et al., 2023) have shown remarkable performance in reasoning tasks through Chain-of-Thought (CoT) (Wei et al., 2022) prompting, which elicits step-by-step rationales to derive answers. However, complex multi-step reasoning remains challenging, particularly for smaller-scale models (Dziri et al., 2024). Recent advances in tree-search algorithms (Wang et al., 2024b; Yao et al., 2024; Zhang et al., 2024b) improve this by generating step-level candidates [2] and using scoring mechanisms to select the most promising ones iteratively, thereby improving overall generated rationales. However, they typically rely on domain-specific reward models or more powerful LLMs to assess candidate validity (Luo et al., 2024).

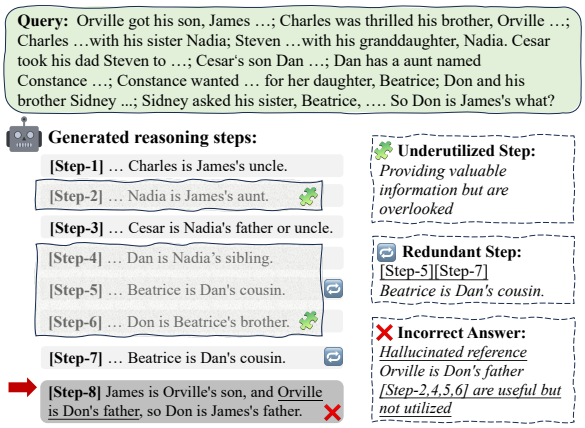

Figure 1: An example illustrating LLMs' difficulty in referencing early-step information (e.g., underutilization of [Step-2,4,5,6]), and the inclusion of redundant steps (e.g., repeated conclusions in [Step-5, 7]). The rightward red arrow indicates the focus is on generating [Step-8] with [Step 1-7] have been generated.

Moreover, LLMs tend to focus on leading and recent contexts while losing attention in the middle (Hsieh et al., 2024). As reasoning progresses, this causes difficulty in referencing useful intermediate conclusions from earlier steps when decoding subsequent ones, leading to unreliable and redundant rationales. For example, in Fig. 1, [Step 2,4,5,6] provide useful information for deriving the final answer but are not effectively utilized. This results in redundant steps (e.g., [Step-7] and [Step-5] have repeated conclusions) and incorrect answer (e.g., [Step-8]). Consequently, LLMs risk

[*] Equal contribution.
[1]Code and data are available at https://github.com/SiyuanWangw/Informativeness-Search.
[2]A reasoning step in this paper refers to a sentence in generated rationales, delimited by the end-of-line token "/n".

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 25280–25298
November 4-9, 2025

getting trapped in repetitive reasoning loops (Chen et al., 2024) and generating unnecessarily lengthy rationales, increasing the likelihood of cumulative errors (Furuta et al., 2024).

To address this, we propose to guide LLMs in generating more accurate and concise step-by-step rationales by (1) proactively referencing intermediate conclusions generated from underutilized steps, and (2) minimizing redundancy between new and existing steps. With higher-quality rationales generated, we can improve answer accuracy and reduce decoding costs. Underutilized steps are those whose intermediate conclusions have been less frequently referenced before the current step, suggesting untapped potential to offer useful information for subsequence reasoning. Meanwhile, reducing redundancy across steps can contribute novel information, enabling more efficient exploration of the reasoning space toward final answers.

We introduce *stepwise informativeness search*, an inference-time tree search framework that prioritizes steps based on informativeness, either by leveraging underutilized steps or generating novel content. The framework follows a stepwise beam search paradigm (Xie et al., 2024), generating multiple candidate steps at each iteration. Based on standard cumulative step-level likelihood, it incorporates two heuristics to guide candidate selection. (1) *Grounding-guided selection* identifies underutilized steps by computing each step's reference degree so far to estimate its information gain for subsequent reasoning. As LLMs naturally assign higher attention to grounding context (Zhang et al., 2023), we prioritize candidate steps with higher attention scores over underutilized steps. (2) *Novelty-guided selection* ranks candidates based on the novelty of their intermediate conclusions relative to prior steps. A trigram-based similarity measure filters out highly similar candidates for simplicity.

We empirically validate that encouraging the grounding of underutilized steps improves reasoning, though LLMs inevitably generate unnecessary steps during reasoning. Specifically, for LLaMA-3-8B self-generated rationales on CLUTRR (Sinha et al., 2019) problems requiring 8 steps, the model exhibits significantly higher accuracy 57.89% when all steps build upon at least one underutilized step, compared to only 22.39% when they do not. To further prevent grounding on underutilized yet irrelevant steps, we introduce a *self-grounding strategy* that elicits LLMs' ability to identify relevant prior steps to provide premises before each deduction. This process enables connecting with distant underutilized steps by first specifying their step numbers, and reinforcing the generation of well-supported new steps through explicit grounding. We implement our informativeness search framework both with and without self-grounding strategy.

Experiments on four multi-step reasoning and one commonsense reasoning datasets validate the effectiveness of the informativeness search framework and self-grounding strategy across five LLMs of varying families and scales. Overall, our framework can generate more effective solutions with improved accuracy and reduced tokens. Moreover, the two heuristics leverage the model's internal outputs and attention to guide step search, making the approach domain-agnostic and efficient by eliminating the need for exhaustive interactions with external scorers or self-evaluation during decoding.

2 Stepwise Beam Search for Reasoning

In this work, we formulate multi-step reasoning as a stepwise beam search process considering its generation parallelizability can accelerates search process (Xie et al., 2024). This contrasts with another common tree-search practice, Monte Carlo Tree Search (MCTS) methods (Feng et al., 2023; Zhang et al., 2024a), which involve extensive rollout simulations and are computationally expensive.

Specifically, at each iteration, the model generates a set of reasoning steps in parallel, each delimited by a special end-of-line token "/n". A beam of the top N steps are selected according to various criteria, where N is the beam size. Unlike step-level evaluation, stepwise beam search ranks candidates by their cumulative rewards (e.g., likelihood) across the sequence generated so far.

Formally, the generation of a reasoning sequence $R = [s_1, s_2, \ldots, s_T]$ with T steps is formulated as

$$P(R = s_{1:T}|x) = \prod_t P(s_t|s_{1:t-1}, x),$$

where s_t is the t-th step and x is the input query. Stepwise generation and selection are performed with beam size N and sample size k as follows: starting with N sequences at step $t-1$, it generates k continuations from $P(s_t|s_{1:t-1}, x)$ for each sequence $s_{1:t-1}$, forming a candidate set C_t containing Nk reasoning chains of length t. The top N sequences are then selected based on a scoring criteria $\phi(C_t, \gamma(\cdot)) = \{s^1, s^2, \ldots, s^N\}$. ϕ is the selection function (e.g., $topk(\cdot)$) and $\gamma(s_{1:t})$ eval-

uates the sequence so far $s_{1:t}$. Initially, given only an input x, we generate Nk candidates.

A standard scoring criteria is the cumulative likelihood of a sequence, defined as: $\gamma_L(s_{1:t}) = \log \prod_t P(s_t|s_{1:t-1}, x)$. Alternative scoring functions $\gamma(s_{1:t})$ are employed in self-evaluation (Xie et al., 2024) and deductive beam search (Zhu et al., 2024). The former prompts the backend LLM to provide a correctness score $\gamma_c(s_t)$ to assess whether s_t is correct given $s_{1:t-1}$, which is then combined with likelihood: $\gamma_E(s_{1:t}) = \log \prod_t P(s_t|s_{1:t-1}, x)\ \gamma_c(s_t)$. The latter trains an external deductive verifier f to assess whether each step s_t is logically entailed by previous contexts, and replaces the sequence likelihood with a cumulative deductive score: $\gamma_D(s_{1:t}) = \prod_t f(\text{entails}|s_t, s_{1:t-1}, x)$.

While these methods improve performance, they require additional annotations or prompts to obtain domain-specific scoring models. They also incur interaction overhead by waiting for scorer response at each decoding step, yet failing to address aforementioned grounding and redundancy challenges.

3 Informativeness Search Framework

Unlike iteration-based scoring functions described above, we introduce stepwise informativeness search framework with two scoring heuristics that utilize model's intrinsic outputs and attention scores. This reduces reliance on off-the-shelf scorers and iterative interactions during decoding. It prioritizes steps based on informativeness, assessed by grounding-guided and novelty-guided heuristics that determine whether new decoded steps ground on underutilized steps and generate novel content.

3.1 Grounding-Guided Selection

To ground each deduction upon underutilized steps to maximally leverage useful information, we design an algorithm to identify underutilized ones among all prior steps. The candidate sequences, denoted as $C_t = \{s^1_{1:t}, s^2_{1:t}, \ldots, s^{Nk}_{1:t}\}$, are then evaluated and selected based on whether each current step s^i_t is well derived from corresponding underutilized steps. Further empirical analysis on the correlation between underutilized steps grounding and improved reasoning is provided in Appendix C.

Identifying Underutilized Steps At each reasoning step, underutilized steps are those referenced less frequently up to that point, offering higher untapped potential for contributing information to subsequent reasoning. At the current step s_t, the immediately preceding step s_{t-1} is by default considered underutilized since it represents the most recent addition to the reasoning path. For additional underutilized steps, we perform a backward traversal from step s_{t-2} to s_1, calculating the reference degree of each step to assess its information gain to subsequent reasoning.

Specifically, for each prior step $s_j \in \{s_{t-2}, ..., s_2, s_1\}$, we first extract its intermediate conclusion c_j by segmenting it using special clause delimiters (e.g., "so", "thus" and commas). We then compare c_j with each subsequent step $s_m \in \{s_{j+1}, \ldots, s_{t-1}\}$ before the current step using a trigram-based similarity measure. The information gain of s_j is computed as follows:

$$\text{InfoGain}(s_j) = 1 - \max_{m \in j+1,\ldots,t-1} \text{Sim}_{\text{tri}}(c_j, s_m)$$

We classify a prior step as underutilized if its information gain exceeds a predefined threshold τ. The set of underutilized steps at step t is:

$$\mathcal{I}_t = \{s_{t-1}\} \cup \{s_j \mid \text{InfoGain}(s_j) > \tau\}, \quad j \in \{1, \ldots, t-2\}$$

Grounding on Underutilized Steps After identifying the set of underutilized steps $\mathcal{I}^i_t$ for each candidate sequence $s^i_{1:t}$ in the candidate set $C_t = \{s^1, s^2, \ldots, s^{Nk}\}$ (with subscripts omitted for simplicity), we prioritize candidates that more effectively ground their reasoning in s^i_t upon their respective underutilized steps.

LLMs typically assign higher attention scores to their grounding context (Zhang et al., 2023). We leverage attention to assess how well each candidate focuses on its identified underutilized steps $\mathcal{I}^i_t$ when constructing step s^i_t. Given that there may be multiple underutilized steps ($|\mathcal{I}^i_t| > 2$), we do not enforce grounding on all of them. Instead, we apply a soft attention mechanism that prioritizes candidates assigning higher weights to a subset of these steps. Specifically, we compute the grounding score of s^i_t over $\mathcal{I}^i_t$ as $\gamma_a(s^i_t)$ by applying mean pooling across all tokens in s^i_t and only highly attended tokens within $\mathcal{I}^i_t$. Detailed calculations are provided in Appendix D. This approach maintains robustness to irrelevant or noisy steps while capturing valuable signals from underutilized steps.

We then integrate this attention-based measure into the original cumulative likelihood scoring function to obtain an grounding-enhanced score:

$$\gamma_G(s_{1:t}) = \gamma_L(s_{1:t}) + \alpha \cdot \gamma_a(s_t)$$

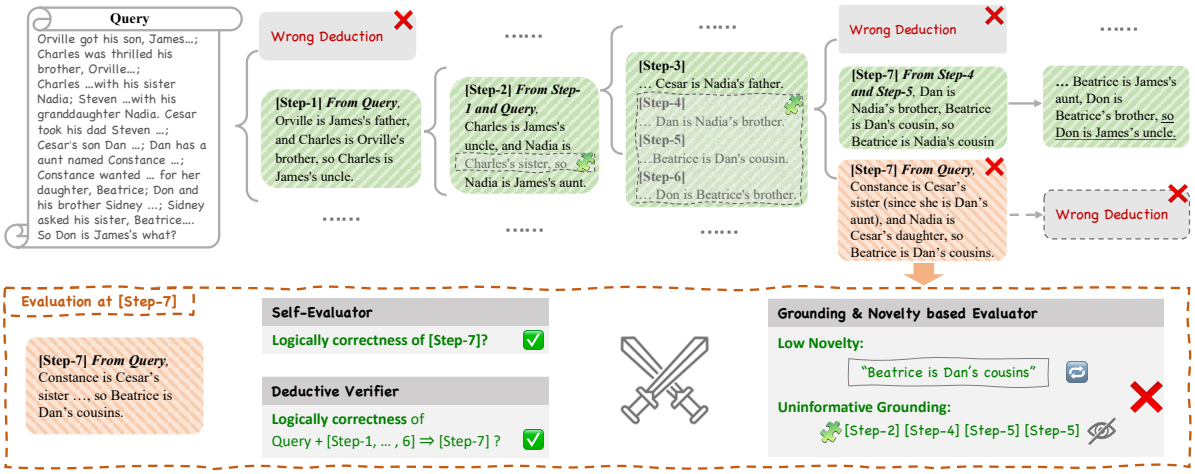

Figure 2: **Upper:** Overview of our informativeness search framework illustrated with beam size of 1. Green blocks indicate selected steps, red-crossed blocks denote discarded steps, and gray blocks contain incorrect deductions. The orange block ([Step-7]) highlights a low-quality step that only can be filter by our method. **Bottom: Evaluation comparison of [Step-7].** Previous methods accept [Step-7] as *logically correct* at both step and sequence levels, whereas our framework filters it out based on its low novelty and poor grounding on underutilized steps.

where $\gamma_L(s_{1:t}) = \log \prod_t P(s_t|s_{1:t-1}, x)$ and α is a weighted hyperparameter. Then N candidates are selected from $C_t = \{s^1, s^2, \ldots, s^{Nk}\}$ with the highest $\gamma_G(s_{1:t})$. We validate this attention-based operation in Sec. 5.3 by analyzing the consistency between highly attended content and actual grounded information.

3.2 Novelty-Guided Selection

To reduce redundancy across multiple intermediate steps, we assess the conclusion novelty of each newly generated step s_t^i in a candidate sequence $s_{1:t}^i$, and select candidates with higher novelty. We extract intermediate conclusions from s_t^i and all its prior steps $\{s_1^i, \ldots, s_{t-1}^i\}$ by segmenting the corresponding sentences using special clause delimiters (e.g., "so", "thus" and commas), forming a set of conclusions $\{c_1^i, \ldots, c_{t-1}^i, c_t^i\}$. We then calculate the trigram-based similarity between the newly generated conclusion c_t^i and all preceding conclusions $\{c_1^i, \ldots, c_{t-1}^i\}$. The novelty score of s_t^i is then obtained as follows:

$$N(s_t^i) = 1 - \max_{j \in 1,\ldots,t-1} \text{Sim}_{\text{tri}}(c_t^i, c_j^i)$$

where $\text{Sim}_{\text{tri}}(\cdot,\cdot)$ measures trigram-based similarity with justification discussed in Appendix E. We also explore embedding-based similarity measure in Sec. 5.4. To incorporate novelty into candidate selection, we calibrate the grounding-enhanced scoring function with novelty score. At step t, candidates with low-novelty conclusions (i.e., $N(s_t) \leq \theta$) are filtered out, retaining only diverse and meaningful candidates. The adjusted scoring function is defined as below, where θ is a predefined threshold.

$$\gamma_N(s_{1:t}) = \begin{cases} \gamma_G(s_{1:t}), & \text{if} N(s_t) > \theta, \\ -100, & \text{otherwise.} \end{cases}$$

3.3 Self-Grounding Strategy

To handle irrelevant steps that may arise during reasoning and prevent grounding-guided selection from focusing on irrelevant prior steps, especially when contexts contain distracting information, we introduce a self-grounding strategy. This approach leverages LLMs' inherent ability to anchor reasoning in relevant prior information, either from prior steps or the input query, that serve as necessary premises for each new deduction. The strategy explicitly prompts LLMs to reason step by step, structuring each step in the format: "`[Step-i] From <source>, <deduction>.`" where "<source>" refers to either relevant prior steps or the input query that provide premises for deducing new conclusions in "<deduction>". For example, "*[Step-1] From Query, we know ...*", "*[Step-2] From Step-1 and Query, we know ...*" and "*[Step-3] From Step-1 and Step-2, because ...*". This explicit step-grounding process ensures that each new step directly builds upon established information, maintaining logical coherence while minimizing irrelevant or unsupported conclusions. Moreover, ex-

Models	Methods	FOLIO	ProofWriter	MMLU-Pro	GPQA-Diamond	Avg.
Llama3.2-3B-Instruct	*Few-shot CoT*	38.73%	40.00%	28.57%	21.72%	32.25%
	Self-Grounding CoT	45.59%	43.33%	28.57%	22.73%	35.06%
	Best-of-N	45.59%	37.00%	30.00%	22.73%	33.83%
	Self-Consistency	46.57%	47.67%	29.64%	22.73%	36.65%
	Tree-of-Thought	44.12%	44.17%	26.43%	22.73%	34.36%
	Self-Eval Beam Search	45.10%	47.00%	30.71%	19.19%	35.50%
	Deductive Beam Search	48.04%	38.17%	25.71%	24.75%	34.17%
	MCTS + Math-PRM	/	/	26.07%	22.22%	/
	Informativeness Search	46.57%	50.33%	33.57%	**27.27%**	39.44%
	Informativeness Search (w/ SG)	**51.96%**	**53.67%**	**33.93%**	24.24%	**40.95%**
Llama3-8B-Instruct	*Few-shot CoT*	54.90%	55.33%	37.50%	29.29%	44.25%
	Self-Grounding CoT	55.39%	57.00%	38.57%	30.30%	45.32%
	Best-of-N	56.86%	50.00%	39.29%	30.30%	44.11%
	Self-Consistency	57.84%	60.17%	39.29%	31.31%	47.15%
	Tree-of-Thought	55.88%	53.33%	39.29%	27.78%	44.07%
	Self-Eval Beam Search	59.31%	56.17%	35.00%	29.80%	45.07%
	Deductive Beam Search	54.90%	48.83%	37.50%	27.78%	42.25%
	MCTS + Math-PRM	/	/	27.14%	28.28%	/
	Informativeness Search	58.33%	61.33%	40.00%	33.33%	48.25%
	Informativeness Search (w/ SG)	**59.80%**	**62.00%**	**40.71%**	**35.35%**	**49.46%**

Table 1: Experimental results (accuracy %) of different methods on Llama3.2-3B-Instruct and LLaMA3-8B-Instruct. *SG* denotes the *Self-Grounding* strategy. Shaded rows present results from our proposed method.

plicitly referencing step numbers facilitates connections with distant underutilized steps. Further details on the prompts and few-shot demonstrations are provided in Appendix B.

4 Experiments

4.1 Setup

Baselines We evaluate against both sequence-level CoT methods and step-level search methods. Sequence-level methods include: (1) Few-shot CoT (Wei et al., 2022) performs step-by-step reasoning. (2) Self-Grounding CoT is our proposed self-grounding strategy without search. (3) Best-of-N (Lightman et al., 2023) samples Nk rationales and selects the best via LLM self-evaluation as we lack general reward models for diverse tasks. (4) Self-Consistency (Wang et al., 2022) samples Nk rationales and uses majority voting for the final answer. Step-level methods include: (5) Tree-of-thought (Yao et al., 2024) performs breadth-first tree search with self-evaluation at each step. (6) Self-Eval Beam Search (Xie et al., 2024) and (7) Deductive Beam Search (Zhu et al., 2024) both use stepwise beam search, with the former relying on self-evaluation and the latter on deductive scoring trained on synthesized datasets. (8) MCTS (Zhang et al., 2024a) where we use the minimum score across all steps from Qwen2.5-Math-PRM-7B (Zhang et al., 2025) to evaluate simulated solutions. As this is a mathematical PRM, we report MCTS results only on MMLU-Pro and GPQA-Diamond. We evaluate our informativeness search with and without the self-grounding (SG) strategy.

Implementation Details We evaluate our framework on four multi-step reasoning datasets: FOLIO (Han et al., 2022), ProofWriter (Tafjord et al., 2020), MMLU-Pro (Wang et al., 2024c) and GPQA (Rein et al., 2023). We mainly evaluate our method and baselines on Llama3.2-3B-Instruct and Llama3-8B-Instruct, using a two-shot prompting strategy with a 1024-token generation limit. We set $N = 3$ and $k = 2$ for all stepwise beam search methods. The parameter α is set to 2 and the threshold τ to 0.7. θ is set to 0.5 for FOLIO and ProofWriter, 0.4 for MMLU-Pro and GPQA-Diamond. Further details and search configurations are provided in Appendix A.

4.2 Main Results

Table 1 presents the overall performance comparison across four benchmark datasets. Our method consistently outperforms all baseline methods across both deductive and diverse reasoning datasets when implemented with either Llama3.2-3B-Instruct or Llama3-8B-Instruct. This demonstrates the general superiority of our informativeness search framework and self-grounding strategy. Notably, our method yields more substantial improvements on Llama3.2-3B-Instruct, suggesting its particular effectiveness in enhancing reasoning for lower-performing models. Additionally,

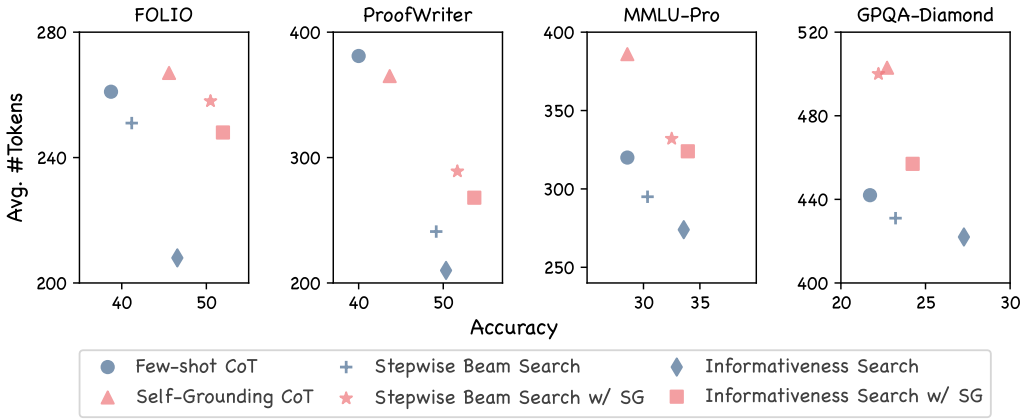

Figure 3: Accuracy and average token count (Avg. # Tokens) of final predicted rationales using different methods on Llama3.2-3B-Instruct.

self-grounding further enhances informativeness search, except when using Llama3.2-3B-Instruct on GPQA-Diamond. We attribute this to Llama3.2-3B-Instruct's inability to perform self-grounding effectively for the challenging GPQA-Diamond task. Step-level methods like tree-of-thought, deductive beam search and MCTS show moderate performance due to their reliance on specialized reward model or verifiers, limiting their generalizability. In contrast, informativeness search is broadly applicable without requiring task-specific customization.

4.3 Efficiency Analysis

Average Rationale Length We analyze the average token count of final predicted rationales using different methods on Llama3.2-3B-Instruct to examine the relationship between rationale length and accuracy. As shown in Figure 3, our method generates shorter rationales with fewer tokens than few-shot CoT and stepwise beam search while achieving higher accuracy, both with and without the self-grounding strategy. Notably, our approach exhibits greater token reduction in deductive reasoning, correlating with more significant performance improvements. We attribute this to our informativeness search framework can effectively reduce redundancy by combining grounding-guided and novelty-guided selection. This minimizes cumulative errors and prevents circular reasoning loops, ultimately leading to better performance.

Total Inference Cost We further analyze the total inference cost, including candidate step generation and evaluation throughout the search process for all methods involving stepwise beam search. As shown in Table 2, we compare token consumption, inference time and memory usage across all methods, evaluated consistently on the same NVIDIA RTX A6000 GPU with batch size 1. Our method effectively reduces token usage and inference time compared to the baseline and other beam search methods while maintaining comparable memory usage, exhibiting superior efficiency. The high cost of Self-Eval and deductive beam search stems from additional interactions to obtain evaluation feedback after each step. Moreover, deductive beam search requires additional computational resources for training a domain-specific deductive verifier.

Search Method	Token Num.	Inference Time	Memory
Self-Eval	$75.0\times K$	589 *min*	7348 *MiB*
Deductive	$46.3\times K$	346 *min*	9789 *MiB*
Stepwise	$2.40\times K$	123 *min*	10918 *MiB*
Stepwise w/ SG	$2.48\times K$	135 *min*	11052 *MiB*
Informativeness	$1.89\times K$	103 *min*	10895 *MiB*
Informativeness w/ SG	$2.17\times K$	112 *min*	11256 *MiB*

Table 2: Efficiency comparison of different stepwise beam search methods on FOLIO. Baseline stepwise beam search uses only cumulative likelihood scoring.

4.4 Results on Additional LLMs

To further validate the broad effectiveness of our method, we implement it on Phi-4 (Abdin et al., 2024), Qwen2.5-7B-Instruct (Qwen et al., 2025) (14B and 7B-parameter models from different families), and DeepSeek-R1-Distill-Llama-8B (Guo et al., 2025), a slow-thinking Llama3-8B variant distilled from DeepSeek-R1. We evaluate performance on FOLIO, ProofWriter, and MMLU-Pro, comparing against few-shot CoT, self-grounding, and self-consistency baselines using corresponding backbones. A one-shot prompting strategy is used with $N = 3$ and $k = 1$, and we extend the generation limit to 2048 tokens to accommodate

long CoT from R1-Distill-Llama-8B. As shown in Table 3, our framework consistently improves performance on more powerful LLMs, though self-grounding fails on R1-Distill-Llama-8B, as it learns to generate free-form CoT and struggles to follow a structured response format. Despite this, our informativeness search still yields significant improvements, notably reducing redundant tokens in final rationales (Table 4). This aligns with DeepSeek-R1's over-thinking problems as pointed by (Chen et al., 2024; Cuadron et al., 2025). These results, along with Table 1 demonstrate our method's robustness across models.

Method	FOLIO	ProofWriter	MMLU-Pro
Phi-4			
Few-shot CoT	73.67%	72.55%	71.79%
Self-Grounding CoT	73.50%	72.06%	72.14%
Self-Consistency	71.17%	72.55%	72.50%
Informativeness Search w/ SG	76.67%	77.94%	72.86%
Qwen2.5-7B-Instruct			
Few-shot CoT	65.20%	54.17%	53.21%
Self-Grounding CoT	67.16%	54.83%	53.57%
Self-Consistency	69.12%	56.33%	54.29%
Informativeness Search w/ SG	70.59%	58.00%	55.36%
DeepSeek-R1-Distill-Llama-8B			
Few-shot CoT	61.76%	48.67%	38.57%
Self-Grounding CoT	53.92%	38.17%	35.36%
Self-Consistency	62.25%	63.50%	46.07%
Informativeness Search	70.10%	66.50%	47.50%

Table 3: Additional results on Phi-4, Qwen2.5-7B-Instruct and R1-Distill-Llama-8B.

Method	FOLIO	ProofWriter	MMLU-Pro
Few-shot CoT	1105	1861	1636
Informativeness Search	588	1023	1001

Table 4: Average token count of the final predicted reasoning paths from R1-Distill-Llama-8B.

4.5 Applicability to General Reasoning Task

To validate the generalization ability of our method on general reasoning tasks beyond multi-step reasoning datasets, we further evaluate it on a randomly sampled subset (100 instances) from the ARC Challenge dataset, a benchmark for commonsense reasoning. We report the results for Llama3.2-3B-Instruct, DeepSeek-R1-Distill-Llama-8B and Phi-4, as shown in Table 5. The consistent performance gains across all models demonstrate the applicability of our method beyond the original four multi-step reasoning datasets.

Methods	Llama3.2-3B	R1-Llama-8B	Phi-4
Few-shot CoT	71%	68%	94%
Self-Grounding CoT	73%	74%	94%
Self-Consistency	70%	75%	95%
Stepwise Beam Search	75%	76%	94%
Informativeness Search w/ SG	78%	83%	97%

Table 5: Performance on ARC Challenge Dataset.

5 Further Analysis

5.1 Ablation Study

To investigate the contribution of our proposed novelty-guided and grounding-guided selection, we conduct an ablation study using LLama3.2-3B-Instruct on FOLIO and MMLU-Pro datasets. Starting with stepwise beam search as our baseline, we separately incorporate each component individually, and compare them with our final *Informative Search*). Table 6 reveals that each selection heuristic contributes substantially to performance improvements, with their integration producing our best-performing search framework overall. Notably, novelty-based selection proves especially effective on FOLIO, addressing the challenge of redundant step generation in deductive reasoning tasks, while grounding-guided selection shows particular strength on MMLU-Pro. These findings validate the standalone effectiveness of each heuristic and their complementary nature when integrated into our complete framework.

Methods	FOLIO	MMLU-Pro
Stepwise Beam Search	41.18%	30.36%
w/ only Novelty-Guided Heuristic	45.10%	32.14%
w/ only Grounding-Guided Heuristic	42.65%	32.50%
Informativeness Search	46.57%	33.57%

Table 6: Ablation study using LLaMA3.2-3B-Instruct.

5.2 Redundant Step Analysis

In complex multi-step reasoning tasks, LLMs tend to generate repeated intermediate conclusions, either from same or different premises, which can trap reasoning in circular loops. For detailed investigation, we measure the average number of repeated conclusions across steps per rationale generated by our method compared to few-shot CoT and self-grounding CoT baselines using LLama3.2-3B-Instruct. Specifically, we split rationales into steps using end-of-line token "/n" and extract intermediate conclusions based on special clause delimiters as operated in Sec. 3.2. A step is considered redundant if its conclusion shares over 70% tri-word

overlap with any previous conclusions in the same rationale. As shown in Figure 4, LLMs exhibit a pronounced tendency to produce redundant steps, particularly in deductive reasoning tasks. This occurs because deductive contexts often contain verbally similar information, causing LLMs to lose track of logical progression and become stuck in circular reasoning. In contrast, our self-grounding strategy and informativeness search substantially reduce redundant steps, enabling more effective and efficient multi-step reasoning.

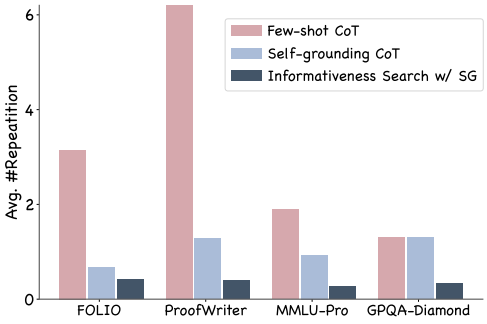

Figure 4: Average count of redundant steps whose conclusions have over 70% tri-word overlap with any previous conclusions in the same rationale.

5.3 Validity of Attention-Based Selection

To validate our attention-based implementation in grounding-guided selection, we examine whether LLMs naturally assign higher attention to grounded steps than other steps. Using the CLUTRR dataset, which provides well-annotated reasoning paths, we conduct a teacher-forcing analysis where all previous ground-truth steps are fed into the model to prompt the next step. We then compute the average attention score over both grounded and non-grounded steps. This analysis is performed both with and without self-grounding, using Llama3.2-3B-Instruct and Llama3-8B-Instruct. As shown in Fig. 5, LLMs exhibit significantly higher attention over grounded steps. This demonstrates the consistency of LLMs' attention patterns and their grounding behavior, and confirms the validity of our attention-based implementation.

5.4 Effectiveness using Embedding-based Similarity

We further explore the effectiveness of our method when utilizing more robust embedding-based similarity instead of simple trigram-based similarity to measure novelty and information gain. Specifically, we employ a lightweight sentence-transformers

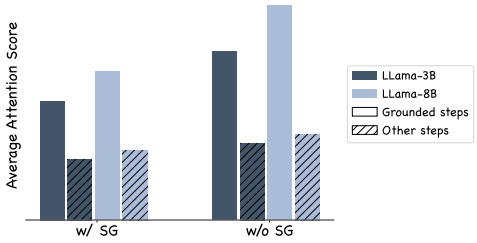

Figure 5: Average attention on grounded and other steps.

model, all-MiniLM-L6-v2 (with only 22.7M parameters), which maps each sentence into a 384-dimensional embedding for cosine similarity computation. We compare this embedding-based method to our original trigram-based measure on FOLIO and MMLU-Pro using Llama3.2-3B-Instruct and Llama3-8B-Instruct. Results in Table 7 suggest that the embedding-based method provides comparable or slightly improved performance on MMLU-Pro, which contains more free-form text. The slight drop on FOLIO may stem from its structured and templated format, where trigram matching more precisely captures exact phrase overlap.

Method	FOLIO	MMLU-Pro
Llama3.2-3B-Instruct		
Few-shot CoT	38.73%	28.57%
Informativeness Search w/SG (Trigram)	51.96%	33.93%
Informativeness Search w/SG (Embedding)	51.47%	35.00%
Llama3-8B-Instruct		
Few-shot CoT	54.90%	37.50%
Informativeness Search w/SG (Trigram)	59.80%	40.71%
Informativeness Search w/SG (Embedding)	59.31%	40.36%

Table 7: Additional results using embedding-based similarity.

6 Related Work

LLMs (OpenAI, 2023; Abdin et al., 2024; Guo et al., 2025) and Chain-of-Thought (CoT) prompting (Wei et al., 2022; Zhou et al., 2022) have demonstrated remarkable performance in reasoning tasks by generating step-by-step rationales. However, for complex multi-step reasoning problems, LLMs often underutilize earlier critic information as rationale getting longer due to tendency to lose focus on middle-context content (See et al., 2017; Peysakhovich and Lerer, 2023; Junqing et al., 2023; Hsieh et al., 2024). They also frequently generate repeated sub-conclusions, leading to redundant reasoning and error accumulation (Dziri et al., 2024; Furuta et al., 2024). These difficulties are pronounced in smaller LLMs with limited reasoning

capacity (Fu et al., 2023). An intuitive method is to simply prompt for concise outputs. However, LLMs often struggle to maintain output quality under length constraints, leaving grounding and redundancy issues unresolved (Nayab et al., 2024; Han et al., 2024).

This inspires generating multiple rationales and selecting the most likely solution via majority voting (Wang et al., 2022) or best-of-N (Wang et al., 2024b). However, they are computationally costly due to exponentially growing search space. To reduce search space, tree search techniques use scoring mechanisms to prioritize promising candidates at each step, such as stepwise beam search (Xie et al., 2024), Tree-of-Thought (Yao et al., 2024), and Monte Carlo Tree Search (Jiang et al., 2024; Feng et al., 2023; Zhang et al., 2024a). While effective, they face practical limitations, relying on extensive rollouts (Wang et al., 2024b,a) and annotations (Lightman et al., 2023) for training specialized reward models. Besides, they introduce latency due to interactions with external or self-evaluators during autoregressive decoding (Xie et al., 2024; Yao et al., 2024), and ignore the grounding and redundancy issues in this work.

7 Conclusion

In this work, we address the challenge of LLMs losing focus on intermediate steps during multi-step reasoning, which can lead to unreliable and redundant rationales. To mitigate this, we propose an inference-time tree search framework incorporating grounding-guided and novelty-guided heuristics, enhancing rationale generation by proactively grounding underutilized steps and minimizing redundancy between reasoning steps. Our self-grounding strategy further prompts LLMs to explicitly reference relevant prior steps before each deduction. Experiments show that our method improves reasoning accuracy and efficiency with fewer errors and reduced redundancy.

Limitations

Our work has several limitations to address in future research. First, due to computational constraints, our main experiments operate within a limited search space with beam size 3 and sample size 2, and use LLM backbones of at most 14B parameters. Future work can explore larger search spaces and more powerful LLMs to further unlock the potential of our framework. Second, while our method currently relies solely on stepwise beam search with standard cumulative likelihood, incorporating our selection heuristics with other scoring mechanism, such as self-evaluation and process reward models, as well as other tree-search algorithms like MCTS could be potential future work.

Acknowledgments

This research is supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the HIATUS Program contract #2022-22072200006, the Defense Advanced Research Projects Agency with award HR00112220046, and NSF IIS 2048211. We would like to thank all the collaborators in USC INK research lab for their constructive feedback on the work.

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, et al. 2024. Do not think that much for 2+ 3=? on the overthinking of o1-like llms. *arXiv preprint arXiv:2412.21187*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Alejandro Cuadron, Dacheng Li, Wenjie Ma, Xingyao Wang, Yichuan Wang, Siyuan Zhuang, Shu Liu, Luis Gaspar Schroeder, Tian Xia, Huanzhi Mao, Nicholas Thumiger, Aditya Desai, Ion Stoica, Ana Klimovic, Graham Neubig, and Joseph E. Gonzalez. 2025. The danger of overthinking: Examining the reasoning-action dilemma in agentic tasks. *Preprint*, arXiv:2502.08235.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, et al. 2024. Faith and fate: Limits of transformers on compositionality. *Advances in Neural Information Processing Systems*, 36.

Xidong Feng, Ziyu Wan, Muning Wen, Stephen Marcus McAleer, Ying Wen, Weinan Zhang, and Jun Wang. 2023. Alphazero-like tree-search can guide large language model decoding and training. *arXiv preprint arXiv:2309.17179*.

Yao Fu, Hao Peng, Litu Ou, Ashish Sabharwal, and Tushar Khot. 2023. Specializing smaller language models towards multi-step reasoning. In *International Conference on Machine Learning*, pages 10421–10430. PMLR.

Hiroki Furuta, Yutaka Matsuo, Aleksandra Faust, and Izzeddin Gur. 2024. Exposing limitations of language model agents in sequential-task compositions on the web. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*.

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

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, et al. 2022. Folio: Natural language reasoning with first-order logic. *arXiv preprint arXiv:2209.00840*.

Tingxu Han, Chunrong Fang, Shiyu Zhao, Shiqing Ma, Zhenyu Chen, and Zhenting Wang. 2024. Token-budget-aware llm reasoning. *arXiv preprint arXiv:2412.18547*.

Cheng-Yu Hsieh, Yung-Sung Chuang, Chun-Liang Li, Zifeng Wang, Long T Le, Abhishek Kumar, James Glass, Alexander Ratner, Chen-Yu Lee, Ranjay Krishna, et al. 2024. Found in the middle: Calibrating positional attention bias improves long context utilization. *arXiv preprint arXiv:2406.16008*.

Jinhao Jiang, Zhipeng Chen, Yingqian Min, Jie Chen, Xiaoxue Cheng, Jiapeng Wang, Yiru Tang, Haoxiang Sun, Jia Deng, Wayne Xin Zhao, et al. 2024. Technical report: Enhancing llm reasoning with reward-guided tree search. *arXiv preprint arXiv:2411.11694*.

He Junqing, Pan Kunhao, Dong Xiaoqun, Song Zhuoyang, Liu Yibo, Liang Yuxin, Wang Hao, Sun Qianguo, Zhang Songxin, Xie Zejian, et al. 2023. Never lost in the middle: Improving large language models via attention strengthening question answering. *arXiv preprint arXiv:2311.09198*.

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

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

Sania Nayab, Giulio Rossolini, Giorgio Buttazzo, Nicolamaria Manes, and Fabrizio Giacomelli. 2024. Concise thoughts: Impact of output length on llm reasoning and cost. *arXiv preprint arXiv:2407.19825*.

OpenAI. 2023. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Alexander Peysakhovich and Adam Lerer. 2023. Attention sorting combats recency bias in long context language models. *arXiv preprint arXiv:2310.01427*.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

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

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Koustuv Sinha, Shagun Sodhani, Jin Dong, Joelle Pineau, and William L Hamilton. 2019. Clutrr: A diagnostic benchmark for inductive reasoning from text. *arXiv preprint arXiv:1908.06177*.

Oyvind Tafjord, Bhavana Dalvi Mishra, and Peter Clark. 2020. Proofwriter: Generating implications, proofs, and abductive statements over natural language. *arXiv preprint arXiv:2012.13048*.

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

Chaojie Wang, Yanchen Deng, Zhiyi Lyu, Liang Zeng, Jujie He, Shuicheng Yan, and Bo An. 2024a. Q*: Improving multi-step reasoning for llms with deliberative planning. *arXiv preprint arXiv:2406.14283*.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024b. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439.

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

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. 2024c. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*.

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. 2024. Self-evaluation guided beam search for reasoning. *Advances in Neural Information Processing Systems*, 36.

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

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

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

Qingru Zhang, Chandan Singh, Liyuan Liu, Xiaodong Liu, Bin Yu, Jianfeng Gao, and Tuo Zhao. 2023. Tell your model where to attend: Post-hoc attention steering for llms. *arXiv preprint arXiv:2311.02262*.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

Tinghui Zhu, Kai Zhang, Jian Xie, and Yu Su. 2024. Deductive beam search: Decoding deducible rationale for chain-of-thought reasoning. *arXiv preprint arXiv:2401.17686*.

A Implementation Details

A.1 Baseline Details

For Best-of-N and Self-Consistency, we adopt a sampling configuration with temperature $T = 1.0$ and top-40 token truncation. For tree-of-thought (ToT) and self-eval beam search (Self-Eval BS), we prompt LLMs to conduct self-evaluation. For deductive beam search that provide a general verifier checkpoint and two data subsets for training a commonsense and a mathematical verifier, we select the best-performing verifier for each dataset. Specifically, we use the general or commonsense verifier for FOLIO, ProofWriter, and MMLU-Pro, and the general or mathematical verifier for GPQA. For MCTS which operates in a iterative four-stage manner: selection, expansion, simulation and backprogation, we use the minimum score across all steps from Qwen2.5-Math-PRM-7B (Zhang et al., 2025) to evaluate simulated rollout.

Comparison among ToT, MCTS, and Self-evaluation Beam Search ToT and Self-evaluation beam search both perform stepwise tree search using a scorer function at each step. The key difference lies in granularity: Self-eval scores and accumulates at the individual step level, while ToT evaluates each thought generated so far as a unit. In contrast, MCTS follows a fundamentally different approach. It performs simulation-based search, where each node is evaluated through multiple rollouts, and scores are provided by a reward model. While this enables deeper exploration, it also introduces significantly higher computational cost.

A.2 Dataset Details

We evaluate our framework on four multi-step reasoning datasets: FOLIO (Han et al., 2022), ProofWriter (Tafjord et al., 2020), MMLU-Pro (Wang et al., 2024c) and GPQA (Rein et al., 2023), and and one commonsense dataset ARC Challenge (Clark et al., 2018). FOLIO and ProofWriter focus on deductive reasoning, requiring 1-8 and 1-6 reasoning steps respectively, with test sets of 204 and 600 cases. MMLU-Pro covers 14 domains, including math, physics, chemistry, engineering, law, and psychology, from which we uniformly sample 280 cases. GPQA specializes in biology, physics, and chemistry, and we use its Diamond subset containing 198 expert-answered but non-expert-failed questions.

A.3 Implementation Details

We apply grounding-guided and novelty-guided selection primarily after the third step, as the initial steps typically ground on the given context and contain minimal redundant steps. Additionally, our prompt requires summarizing the final answer after the reasoning process, separated by the special token "[END]". We restrict grounding-guided and novelty-guided selection to the reasoning phase only, ensuring that the final answer output, which often overlaps with the last reasoning step, remains unaffected.

A.4 Varying Search Configurations

For step-level candidate generation in stepwise beam search, we explore both temperature sampling and tokenwise beam search. As shown in Table 8, our method with grounding and novelty-guided selection consistently outperforms stepwise beam search baseline (with cumulative likelihood scoring), regardless of whether self-grounding is applied. Additionally, tokenwise beam search for candidate generation yields slightly better performance than temperature sampling.

Methods	FOLIO	MMLU-Pro
Beam Search		
Stepwise Beam Search	41.18%	30.36%
Informativeness Search	46.57%	33.57%
Stepwise Beam Search (w/ SG)	50.49%	32.86%
Informativeness Search (w/ SG)	51.96%	33.93%
Temperature Sampling		
Stepwise Beam Search	41.67%	29.64%
Informativeness Search	44.12%	31.43%
Stepwise Beam Search (w/ SG)	47.55%	29.64%
Informativeness Search (w/ SG)	48.53%	32.50%

Table 8: Different candidate step generation methods.

We further evaluate the impact of varying beam sizes in our informativeness search, using both tokenwise beam search and temperature sampling for candidate step generation. Specifically, we set the sample size to 2 and vary the beam size from 1 to 4. As shown in Fig. 6, both alternatives consistently outperform the few-shot CoT baseline. Additionally, our informativeness search continues to improve as beam size increases. Notably, when the search space is constrained (i.e., with a smaller beam size), tokenwise beam search performs better. Based on these findings, we adopt tokenwise beam search for all stepwise beam search methods in our reported results (Table 1$\sim$ 4) considering its better performance and accelerated computational speed.

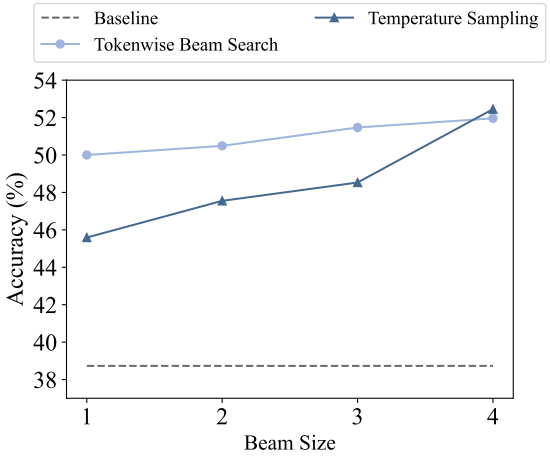

Figure 6: The impact of beam size on our utility-based search for the FOLIO dataset on Llama3.2-3B-Instruct.

A.5 Comparison to Tokenwise Beam Search

We further compare our informativeness search (beam size $N = 3$, sample size $k = 2$) with naive tokenwise beam search for whole rationale generation using beam size 3 and 6. Table 9 demonstrate the effectiveness of our method.

Method	FOLIO	ProofWriter	MMLU-Pro	GPQA-D
Few-shot CoT	38.73%	40.00%	28.57%	21.72%
Tokenwise BS (3)	43.63%	45.00%	28.93%	21.72%
Tokenwise BS (6)	46.08%	42.17%	31.07%	19.19%
Informativeness Search	46.57%	50.33%	33.93%	27.27%

Table 9: Comparison with tokenwise beam search using Llama3.2-3B-Instruct for whole rationale generation. Numbers in parentheses denote the beam size.

B Framework Prompts

Table 10, 11, 12 and 13 present the prompts used in our informativeness search framework without self-grounding strategy for the FOLIO, ProofWriter, MMLU-Pro and GPQA-Diamond datasets. For illustration, Table 14 provides the prompt used in our informativeness search framework with self-grounding strategy on GPQA-Diamond [3].

C Correlation between of Grounding Challenge and Reasoning Performance

We provide a detailed illustration of the grounding challenge that LLMs face when referencing

[3]We use GPT-4o and Claude to adjust prompts manually.

prior reasoning steps. Specifically, we analyze all instances involving 8-9 reasoning steps from CLUTRR (Sinha et al., 2019), a dataset with well-annotated rationales. We evaluate the performance of Llama3-8B-Instruct across instances grouped by the maximum distances between referencing and referenced steps. As shown in Fig. 7, performance degrades as the distance to the referenced prior steps grows. This demonstrates the inherent difficulty of grounding distant prior step, with longer distances (steps accumulating) progressively degrade reasoning performance.

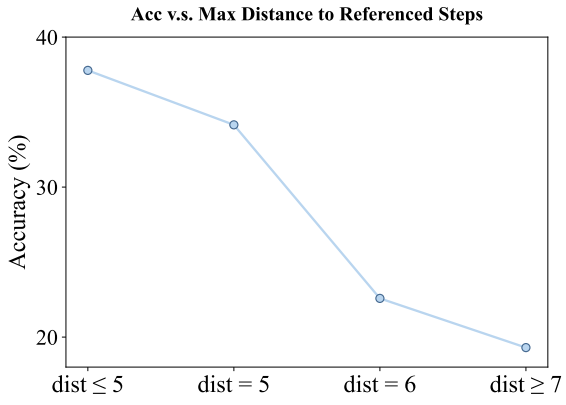

Figure 7: Accuracy versus maximum distance between referencing and referenced steps on CLUTRR.

D Attention Calculation

To compute the grounding score $\gamma_a(s_t^i)$ that measures how effectively step s_t^i attends to the underutilized steps $\mathcal{I}_t^i$, we use a selective attention mechanism that emphasizes the most relevant connections while remaining robust to noisy information. Our approach proceeds in three stages. First, we collect attention weights from all L layers and H heads of the model. For each generated step, we extract the attention scores from the N tokens in the current step s_t^i to the M tokens in the underutilized prior steps $\mathcal{I}_t^i$, yielding $L \times H \times M \times N$ attention values. Next, we take the mean over the N tokens in the current step s_t^i, reducing it to $L \times H \times M$ scores to measure how much the step s_t^i as a whole attends to each token in underutilized steps. Finally, we select the top-K scores (we experiment and conclude that $K = 50 \sim 100$ yield similar results) and compute their mean to obtain the final attention score $\gamma_a(s_t^i)$. This selective averaging maintains robustness to irrelevant or noisy steps while capturing most valuable signals from underutilized steps.

E Justification of Using Trigram-based Similarity

We simply adopt trigram-based similarity to measure novelty and information gain for each step for the following reasons. (1) Similarity computation includes both semantics-based similarity and N-gram-based similarity. However, the former often requires additional model inference, introducing substantial computational overhead. To reduce costs, we opted for N-gram-based similarity, which can be computed efficiently without model calls. (2) Among N-gram-based metrics, uni-gram similarity performs poorly, as it only captures word overlap and lacks sentence-level representation. In contrast, higher-order N-grams (e.g., 4-gram) are overly strict and fail to recognize semantically similar sentences with slightly different wording. Through empirical comparisons on bi-gram and trigram similarity, we found that trigram-based similarity provides the best trade-off between robustness and flexibility, making it the preferred choice in our implementation. Alternatively, lightweight embedding-based measures can further enhance semantic fidelity and merit further exploration.

F Further Discussion on Handling Intermediate Noise or Repetition

LLMs inevitably generate irrelevant or incorrect steps during reasoning, making not all underutilized steps useful. However, we argue that our method encouraging the grounding of underutilized steps still improve reasoning. As discussed in Sec. 1, empirical evidence shows that when LLaMA-3-8B is prompted to self-generate multi-step rationales on CLUTRR instances requiring eight ground-truth steps, accuracy is significantly higher (57.89%) in cases where each step grounds on at least one underutilized prior step, compared to 22.39% when they do not. Besides, our proposed self-grounding strategy further mitigate the impact of irrelevant steps by prompting the model to identify and state relevant prior steps before each deduction, leveraging its internal reasoning to surface useful content. Finally, we adopt a soft attention mechanism prioritizes grounding on a selective subset of underutilized steps, maintaining robustness against noise while capturing valuable signals.

Additionally, our framework can extend to long-thinking models that generate intermediate conclusions similar to their final answers. Unlike short-thinking models that terminate reasoning upon

reaching likely answer, long-thinking models usually explore multiple alternative solutions or perform verification. However, valuable alternative solutions and verifications often appear in varied expressions. For instance, the conclusion "the answer to 2 + 3 is 5" may emerge through diverse intermediate forms such as "two apples plus three apples equals five apples" or "2 is II and 3 is III in Roman numerals, adding them gives V". Similarly, useful verifications include phrases like "which is the same as 2 plus 3" or "this aligns with 2 plus 3 equals 5". Our trigram-based similarity metric recognizes these varied expressions as semantically novel, encouraging diverse reasoning paths that increase information richness and model confidence in its answer. The strong performance of DeepSeek-R1-Distill-Llama-8B in Table 3 validates the generalization of our approach to long-thinking models. Moreover, we can apply in-block novelty-guided selection, filtering within reasoning blocks segmented by special transition tokens (e.g., "Wait," or "Alternatively") to maintain diversity while mitigating the risk of filtering meaningful content.

Prompt without Self-Grounding (FOLIO)

You are a helpful assistant.
You will receive a query. Your task is to answer the query.

Examples
Query: LanguageA is a universal language. If a universal language exists, then for every two people if they both know the same universal language they can communicate. Katya cannot communicate with Danil. Katya knows LanguageA. Based on the above information, is the following statement true, false, or uncertain? Danil knows LanguageA.
Thought:
Because LanguageA is a universal language, and if a universal language exists, then for every two people if they both know the same universal language they can communicate, so every two people that know LanguageA can communicate.
Because every two people that know LanguageA can communicate, and Katya knows LanguageA, so Katya can communicate with others that know LanguageA.
Because Katya can communicate with others that knows LanguageA, and Katya cannot communicate with Danil, so Danil does not know LanguageA.
Therefore, the statement "Danil knows LanguageA." is False.
END.
So the answer is: False.

Query: All eels are fish. No fish are plants. A thing is either a plant or animal. Nothing that breathes is paper. All animals breathe. If a sea eel is either an eel or a plant, then a sea eel is an eel or an animal. Based on the above information, is the following statement true, false, or uncertain? Sea eel breathes or is paper.
Thought:
Because all eels are fish, so a sea eel is a fish.
Because no fish are plants, a thing is either a plant or animal, so a fish is an animal.
Because a sea eel is a fish, and a fish is an animal, so a sea eel is an animal.
Because a sea eel is an animal, and all animals breathe, so a sea eel breathes.
Because a sea eel breathes and nothing that breathes is paper, so a sea eel is not paper.
Therefore, the statement "Sea eel breathes or is paper." is True.
END.
So the answer is: True.

Here's what you need to do. Please first think step-by-step, give out each of your step in a newline, then end your thought with "END.". Finally respond "True", "False" or "Uncertain" in a newline, strictly starting with "So the answer is: ".

Table 10: The prompt without self-grounding on FOLIO.

Prompt without Self-Grounding (ProofWriter)

You are a helpful assitant.
You will receive a query. Your task is to answer the query.

Examples
Query: Bob is big. Dave is big. Dave is rough. Erin is nice. Erin is white. Gary is nice. Gary is white. Red things are white. All big things are green. All red, white things are nice. All green things are blue. If something is nice then it is big. All blue, green things are rough. All rough things are red. If something is blue then it is nice. If something is red then it is blue. Based on the above information, is the following statement true, false, or unknown? Gary is not red.
Thought:
Because Gary is nice, and if something is nice then it is big, so Gary is big.
Because Gary is big and all big things are green, so Gary is green.
Because Gary is green and all green things are blue, so Gary is blue.
Because Gary is green and Gary is blue, and all blue, green things are rough, so Gary is rough.
Because Gary is rough and all rough things are red, so Gary is red.
Therefore, the statement "Gary is not red." is false.
END.
So the answer is: False.

Query: Anne is nice. Anne is smart. Charlie is green. Fiona is nice. Fiona is round. Fiona is white. Harry is blue. White, kind things are nice. If something is smart and kind then it is green. If something is round and kind then it is white. Smart things are kind. Nice, white things are kind. Round things are kind. If something is nice then it is smart. All white things are round. If Charlie is green then Charlie is white. Based on the above information, is the following statement true, false, or unknown? Charlie is smart.
Thought:
Because Charlie is green, and if Charlie is green then Charlie is white, so Charlie is white.
Because Charlie is white and all white things are round, so Charlie is round.
Because Charlie is round and round things are kind, so Charlie is kind.
Because Charlie is white and Charlie is kind, and white, kind things are nice, so Charlie is nice.
Because Charlie is nice, and if something is nice then it is smart, so Charlie is smart.
Therefore, the statement "Charlie is smart." is true.
END.
So the answer is: True.

Here's what you need to do. Please first think step-by-step, give out each of your step in a newline, then end your thought with "END.". Finally respond "True", "False" or "Unknown" in a newline, strictly starting with "So the answer is: ".

Table 11: The prompt without self-grounding on ProofWriter.

Prompt without Self-Grounding (MMLU-Pro)

You will receive a query and ten options. Your task is to select an option to answer the query.

Examples
Query: Kylar went to the store to buy glasses for his new apartment. One glass costs $5, but every second glass costs only 60% of the price. Kylar wants to buy 16 glasses. How much does he need to pay for them?
Options: A.24, B.54, C.40, D.32, E.64, F.8, G.16, H.60, I.100, J.74
Thought:
Because one glass costs $5, and every second glass costs only 60% of the price, so the discount price of every second glass is 60/100 * 5 = $3.
Because every second glass is discounted at $3, and Kylar wants to buy 16 glasses, so Kylar is going to buy 16 / 2 = 8 discounted glasses and 16 - 8 = 8 regular-priced glasses.
Because Kylar is going to buy 8 discounted glasses, and every discounted glass is $3, so Kylar is going to pay 8 * 3 = $24.
Because Kylar is also going to buy 8 regular-priced glasses, and one glass costs $5, so Kylar will pay 8 * 5 = $40.
Because Kylar will pay $24 for 8 discounted glasses, and $40 for 8 regular-priced glasses, so in total Kylar needs to pay 24 + 40 = $64 for the glasses he wants to buy.
END.
So the answer is: E.

——

Query: A refracting telescope consists of two converging lenses separated by 100 cm. The eye-piece lens has a focal length of 20 cm. The angular magnification of the telescope is ?
Options: A.10, B.40, C.6, D.25, E.15, F.50, G.30, H.4, I.5, J.20
Thought:
Because in a refracting telescope both lenses are converging, so their focus must be between the two lenses.
Because the focus of both lenses must lie between them, so their focal lengths must add up to their separation.
Because the two lenses are separated by 100 cm, and one lens has a focal length of 20 cm, so the other lens must have a focal length of 80 cm.
Because one lens has a focal length of 20 cm and the other 80 cm, so the magnification is the ratio of their focal lengths, which is 4.
END.
So the answer is: H.

Here's what you need to do. Please first think step-by-step, presenting each of your step in a new line. Then end your thought with "END.". Finally respond with an option from "A", "B", "C", "D", "E", "F", "G", "H", "I" or "J" in a newline, strictly starting with "So the answer is: ".

Table 12: The prompt without self-grounding on MMLU-Pro.

Prompt without Self-Grounding (GPQA-Diamond)

You will receive a query along with four options. Your task is to select an option to answer the query.

Examples
Query: Kylar went to the store to buy glasses for his new apartment. One glass costs $5, but every second glass costs only 60% of the price. Kylar wants to buy 16 glasses. How much does he need to pay for them?
Options:
(A) 24
(B) 54
(C) 40
(D) 64
Thought:
Because one glass costs $5, and every second glass costs only 60% of the price, so the discount price of every second glass is 60/100 * 5 = $3.
Because every second glass is discounted at $3, and Kylar wants to buy 16 glasses, so Kylar is going to buy 16 / 2 = 8 discounted glasses and 16 - 8 = 8 regular-priced glasses.
Because Kylar is going to buy 8 discounted glasses, and every discounted glass is $3, so Kylar is going to pay 8 * 3 = $24.
Because Kylar is also going to buy 8 regular-priced glasses, and one glass costs $5, so Kylar will pay 8 * 5 = $40.
Because Kylar will pay $24 for 8 discounted glasses, and $40 for 8 regular-priced glasses, so in total Kylar needs to pay 24 + 40 = $64 for the glasses he wants to buy.
END.
So the answer is: D.

——

Query: A refracting telescope consists of two converging lenses separated by 100 cm. The eye-piece lens has a focal length of 20 cm. The angular magnification of the telescope is ?
Options:
(A) 10
(B) 6
(C) 4
(D) 25
Thought:
Because in a refracting telescope both lenses are converging, so their focus must be between the two lenses.
Because the focus of both lenses must lie between them, so their focal lengths must add up to their separation.
Because the two lenses are separated by 100 cm, and one lens has a focal length of 20 cm, so the other lens must have a focal length of 80 cm.
Because one lens has a focal length of 20 cm and the other 80 cm, so the magnification is the ratio of their focal lengths, which is 4.
END.
So the answer is: C.

Here's what you need to do. Please first think step-by-step, give out each of your step in a newline. Then end all your thought with "END.". Finally respond with an option from "A", "B", "C" or "D" in a newline, strictly starting with "So the answer is: ".

Table 13: The prompt without self-grounding on GPQA-Diamond.

Prompt with Self-Grounding (GPQA-Diamond)

You will receive a query along with four options. Your task is to select an option to answer the query.

Examples
Query: Kylar went to the store to buy glasses for his new apartment. One glass costs $5, but every second glass costs only 60% of the price. Kylar wants to buy 16 glasses. How much does he need to pay for them?
Options:
(A) 24
(B) 54
(C) 40
(D) 64
Thought:
[Step-1] From Query, because one glass costs $5, and every second glass costs only 60% of the price, so the discount price of every second glass is 60/100 * 5 = $3.
[Step-2] From Step-1 and Query, because every second glass is discounted at $3, and Kylar wants to buy 16 glasses, so Kylar is going to buy 16 / 2 = 8 discounted glasses and 16 - 8 = 8 regular-priced glasses.
[Step-3] From Step-1 and Step-2, because Kylar is going to buy 8 discounted glasses, and every discounted glass is $3, so Kylar is going to pay 8 * 3 = $24.
[Step-4] From Step-2 and Query, because Kylar is also going to buy 8 regular-priced glasses, and one glass costs $5, so Kylar will pay 8 * 5 = $40.
[Step-5] From Step-3 and Step-4, because Kylar will pay $24 for 8 discounted glasses, and $40 for 8 regular-priced glasses, so in total Kylar needs to pay 24 + 40 = $64 for the glasses he wants to buy.
END.
So the answer is: D.

——

Query: A refracting telescope consists of two converging lenses separated by 100 cm. The eye-piece lens has a focal length of 20 cm. The angular magnification of the telescope is ?
Options:
(A) 10
(B) 6
(C) 4
(D) 25
Thought:
[Step-1] From Query, because in a refracting telescope both lenses are converging, so their focus must be between the two lenses.
[Step-2] From Step-1, because the focus of both lenses must lie between them, so their focal lengths must add up to their separation.
[Step-3] From Step-2 and Query, because the two lenses are separated by 100 cm, and one lens has a focal length of 20 cm, so the other lens must have a focal length of 80 cm.
[Step-4] From Step-3 and Query, because one lens has a focal length of 20 cm and the other 80 cm, so the magnification is the ratio of their focal lengths, which is 4.
END.
So the answer is: C.

Here's what you need to do. Please first think step-by-step, give out each of your step in a newline starting with [Step-i], and cite the sources (e.g., Step-i, Query) of your premises at the beginning of each step. Then end all your thought with "END.". Finally respond with an option from "A", "B", "C" or "D" in a newline, strictly starting with "So the answer is: ".

Table 14: The prompt with self-grounding on GPQA-Diamond.