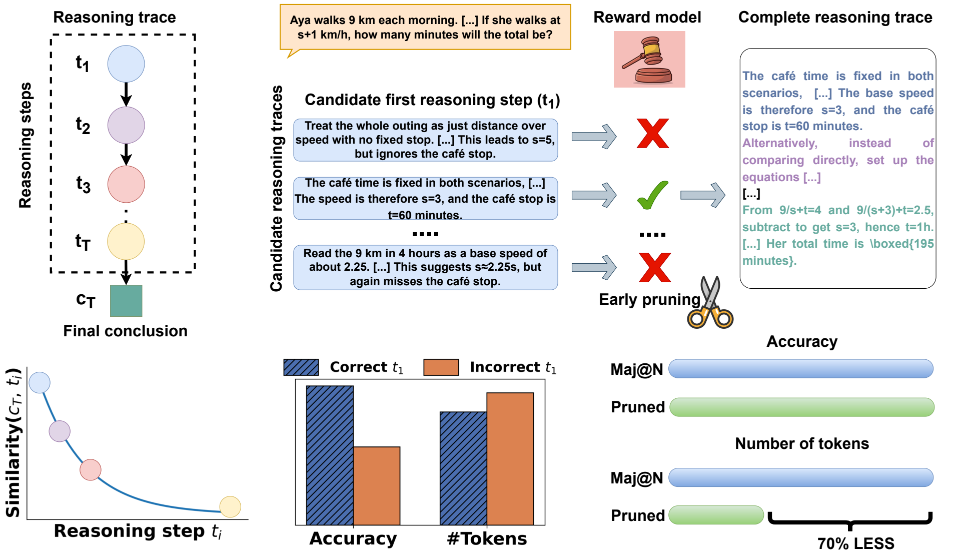
LOST AT THE BEGINNING OF REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent advancements in large language models (LLMs) have significantly advanced complex reasoning capabilities, particularly through extended chain-of-thought (CoT) reasoning that incorporates mechanisms such as backtracking, self-reflection, and self-correction. Despite these developments, the self-correction abilities of LLMs during long CoT reasoning remain underexplored. And recent findings on overthinking suggest that such models often engage in unnecessarily redundant reasoning. In this work, we empirically show that the first reasoning step exerts a disproportionately large influence on the final prediction—errors introduced at this stage can substantially degrade subsequent reasoning quality. This phenomenon is consistently observed across various state-of-the-art open- and closed-source reasoning models. Leveraging this insight, we propose an efficient sampling strategy that leverages a reward model to identify and retain high-quality first reasoning steps while discarding suboptimal ones, achieving up to a 70% reduction in inference cost without sacrificing accuracy.



Aya walks 9 km each morning. [...] If she walks at s+1 km/h, how many minutes will the total be?

Treat the whole outing as just distance over speed with no fixed stop. [...] This leads to s=5, but ignores the café stop.

The café time is fixed in both scenarios, [...] The speed is therefore s=3, and the café stop is t=60 minutes.

....

Read the 9 km in 4 hours as a base speed of about 2.25. [...] This suggests s≈2.25s, but again misses the café stop.

The café time is fixed in both scenarios, [...] The base speed is therefore s=3, and the café stop is t=60 minutes.
Alternatively, instead of comparing directly, set up the equations [...]
[...]
From 9/s+t=4 and 9/(s+3)+t=2.5, subtract to get s=3, hence t=1h. [...] Her total time is \boxed{195 minutes}.

Figure 1: Overview of our observation and efficient sampling. The first reasoning step t_1 heavily shapes the entire reasoning trajectory: a strong first step typically yields accurate solutions with fewer tokens (bottom left). Building on this observation, we propose to generate multiple candidate first steps, evaluate them with a reward model, and discard weaker candidates early (top right). This method maintains accuracy while substantially reducing token consumption by 70% (bottom right).

1 INTRODUCTION

Large language models (LLMs) have demonstrated remarkable performance across a variety of reasoning tasks, ranging from mathematical problem solving to multi-hop question answering (Hestness et al., 2017; Kaplan et al., 2020; Hoffmann et al., 2022). More recently, the advent of reasoning-oriented LLMs capable of performing long chain-of-thought (long-CoT) reasoning at test time has led to substantial advancements in these domains (Brown et al., 2020; Hurst et al., 2024; Anthropic, 2025; Team et al., 2024; Guo et al., 2025; Yang et al., 2025a; Wen et al., 2025; He et al., 2025). A widely held hypothesis attributes this progress to the models' ability to backtrack, self-reflect, and self-correct, effectively leveraging contextual feedback to iteratively refine their responses.

However, recent studies suggest that long-CoT reasoning can also introduce inefficiencies. Models often "overthink" by producing unnecessarily extended reasoning processes to solve problems (Chiang & Lee, 2024; Zhang et al., 2024a; Wang et al., 2025b; Liao et al., 2025b;a). This observation raises questions about the model's capacity for backtracking, self-reflection, and self-correction. It suggests that LLMs may lack awareness of the information they have already processed, leading to redundant or inefficient reasoning. Moreover, Liu et al. (2024a) demonstrate that LLMs are prone to the "lost-in-the-middle" phenomenon, wherein information located in the middle of a long context is often overlooked. While their analysis is conducted in the context of information retrieval, it remains an open question whether similar positional biases affect long CoT as well.

In this work, we introduce a novel and previously underexplored perspective on long-CoT reasoning: many reasoning failures in long-CoT LLMs stem not from errors made mid-chain, but rather from flaws at the beginning of reasoning. Our experiments demonstrate that the first reasoning step has the most significant influence on the final prediction. When this first step is incorrect, the model is considerably more likely to arrive at an incorrect final answer (40% accuracy drop), highlighting the limited self-correction capabilities of current long-CoT LLMs. Notably, this phenomenon is consistently observed from five open- and closed-source long-CoT LLM families.

Motivated by this insight, we propose an efficient early pruning algorithm that exploits the predictive power of the first reasoning step. Specifically, by evaluating the quality of the first step, we identify and discard less promising reasoning traces early, continuing generation only for the more promising ones. This approach significantly reduces inference cost. Across five open-sourced long-CoT LLM families and five challenging mathematical, scientific reasoning and programming benchmarks, our method maintains accuracy while reducing inference budget by up to 70%. Our results show that the first step is not just the beginning of reasoning, but a key factor that influences both accuracy and efficiency, making it an important focus for future reasoning models.

Contributions. Our main contributions are as follows: (1) To the best of our knowledge, we firstly empirically establish a strong positive correlation between the first reasoning step and the final prediction across various open- and closed-sourced long-CoT LLM families (§3); (2) Inspired by this observation, we propose an efficient early pruning algorithm that halts generation for less promising initial steps, thereby improving inference efficiency while maintaining the accuracy (§4); (3) Both observation and proposed efficient sampling method are extensively validated on various long-CoT LLMs across different reasoning tasks, with necessary control experiments to disentangle the confounding factors.

2 RELATED WORK

Lost in the middle. Liu et al. (2024a) introduced the "lost in the middle" effect, demonstrating that LLMs tend to overlook information in the middle of long contexts, performing better when relevant content appears at the beginning or end. This positional bias is evident across tasks like arithmetic reasoning (Shen et al., 2023; Liao & Monz, 2024), multiple-choice QA (Zheng et al., 2024; Pezeshkpour & Hruschka, 2023), text evaluation (Wang et al., 2024; Shi et al., 2024), passage ranking (Zhang et al., 2024b), and instructional prompt positioning (Liu et al., 2024b; Chen et al., 2024b). Additionally, studies have documented primacy and recency biases, where models disproportionately allocate attention to the first or final tokens, independent of their semantic relevance (Xiao et al., 2024; Qin et al., 2023; Barbero et al., 2025). While previous studies have primarily examined positional biases in external context, we investigate whether analogous biases emerge in internal reasoning trajectories of long chain-of-thought models. Different from attention-level analyses that focus on how the first input token shapes representations, our work shows that the first generated reasoning step greatly influences subsequent reasoning and final outcomes.

Efficient test-time reasoning. Test-time scaling methods aim to improve the accuracy–compute trade-off by adapting sampling and aggregation. One line of work increases self-consistency efficiency by reducing sample counts (Li et al., 2024; Wan et al., 2025; Aggarwal et al., 2023; Xue et al., 2023), while another shortens chain-of-thought depth via fine-tuning or inference-only optimizations (Chen et al., 2024a; Luo et al., 2025; Hou et al., 2025; Fu et al., 2025a; Yang et al., 2025b). These methods, however, still rely on generating full reasoning traces. DeepConf (Fu et al., 2025b) instead uses local confidence to filter low-quality traces and terminate generation early. Our method takes a different focus: we assess the quality of the initial reasoning step, which strongly shapes subsequent reasoning, and prune weak starts before long traces unfold.

3 LOST AT THE BEGINNING OF REASONING

Motivated by the finding of Liu et al. (2024a), which demonstrates that query-relevant information is more impactful when positioned at either the beginning or end of an LLM's context window, we first investigate whether a similar positional effect exists in long-CoT reasoning (§3.1). Our analysis reveals that the first reasoning step has great impact to the final conclusion. To validate this observation, we further perform two ablation studies, confirming the critical role of the first step in determining the model's final prediction (§3.2 and §3.3).

Notation. Let p represent the input prompt, consisting of both a system instruction and a user query. A reasoning model $\mathcal{M}$ produces a sequence of CoT reasoning steps $t = [t_1, t_2, ..., t_T]$, followed by a final conclusion c_T, such that the complete model output is given by $t \oplus c_T = \mathcal{M}(p)$, where $\oplus$ means concatenation. In models such as DeepSeek-R1 (Guo et al., 2025) and Qwen3 (Team, 2025), the input-output format adheres to the following:

$$p \text{ <think> } t_1, t_2, \ldots, t_T \text{ </think> } c_T$$

The final prediction q_T is then derived by applying an extraction function g to the conclusion, i.e., $q_T = g(c_T)$, where g may, for example, extract values enclosed within `\boxed{}`.

The conclusion c_T can be interpreted as a summary of the essential reasoning steps leading to the final prediction. This raises an interesting question: *Is there a positional bias in how reasoning steps contribute to the conclusion?* In other words, do certain steps have a disproportionately greater influence on c_T than others?

3.1 SIMILARITY BETWEEN REASONING STEPS AND THE FINAL CONCLUSION

To understand how different reasoning steps contribute to the final conclusion, we measure the semantic similarity between each reasoning step $\{t_i\}_{i=1}^{T}$ and the final conclusion c_T.

To assess how intermediate reasoning contributes to the final outcome, we measure the semantic similarity between each reasoning step $\{t_i\}_{i=1}^{T}$ and the final conclusion c_T. This analysis reveals whether the reasoning process gradually aligns with the correct answer or diverges along the way.

Experimental setup. We evaluate 60 questions from AIME24 and AIME25 (MAA Committees, 2025) using DeepSeek-R1-Distill-Qwen-7B (abbreviated as DS-R1-Qwen-7B in the remainder of this paper) (Guo et al., 2025), Qwen3-8B (Yang et al., 2025a), Claude-3.7-Sonnet with thinking (Anthropic, 2025), GPT-OSS-20B (Agarwal et al., 2025), and Magistral-Small (Rastogi et al., 2025).[1] Generations are produced with `temperature=1.0`, `top_p=0.9`, `min_p=0.05`, and `max_tokens=32768`; for Claude-3.7-Sonnet, only `max_tokens` is set. All subsequent experiments adopt this hyperparameter configuration.

Segmentation of reasoning steps. We define a reasoning step as a complete logical leap or self-contained unit (Xiong et al., 2025), and segment reasoning traces with GPT-5.[2] To complement this setup, we also employ heuristic segmentation based on reasoning switch keywords (Wang et al., 2025a), with details provided in Appendix C.

Similarity computation. We compute semantic similarity between each step t_i and the conclusion c_T by taking the cosine similarity of their embeddings obtained from all-MiniLM-L6-v2 (Reimers & Gurevych, 2019; Wang et al., 2020). To avoid inflated similarity from problem restatement, we use GPT-5 mini to remove question-overlap text at the beginning of traces. As a robustness check,

---

[1] For reproducibility, the exact model identifiers are provided in Appendix D.

[2] By default, we use GPT-5 mini for step segmentation; for GPT-OSS-20B, we instead use GPT-5, as the mini variant is incompatible.

we also report results with SPLADE similarity (Formal et al., 2021) in Appendix E, confirming that our findings are not specific to dense embeddings. Since traces vary in length, similarity curves are interpolated to a fixed number of steps (either the average or maximum length) for visualization.

This setup allows us to capture how reasoning trajectories semantically converge toward—or deviate from—the final answer across different models.

Result. Figure 2 shows that the first reasoning step exhibits the highest similarity to the final conclusion, after which similarity drops sharply. Beyond the initial few steps, similarity stabilizes at a lower level, with only minor fluctuations across the remainder of the reasoning process. These results suggest that the first step t_1 is most closely aligned with the final conclusion and likely sets the overall direction of the reasoning. Subsequent steps appear to introduce exploratory or redundant content that deviates from the final answer. Additional results using SPLADE similarities (Figure E.2) show the same trend, confirming that this pattern is robust across similarity metrics. Taken together, these findings show that the first reasoning step plays a disproportionately important role in shaping the final conclusion.

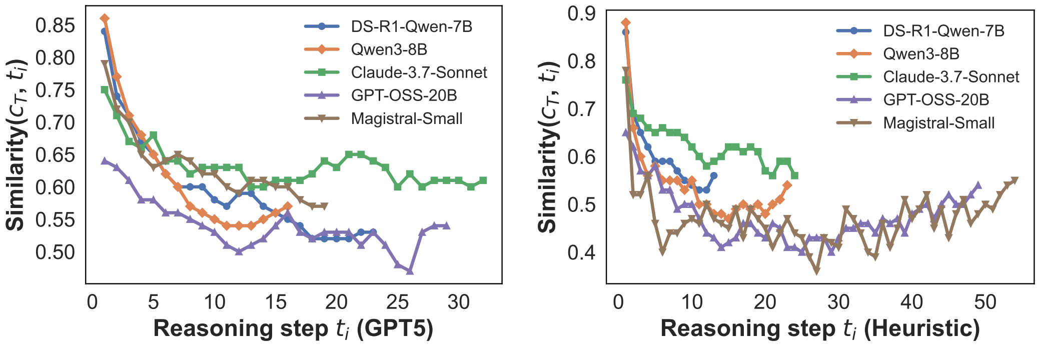


Figure 2: Cosine similarity between the embeddings of the i-th reasoning step t_i and the final conclusion c_T, using the average number of reasoning steps for interpolation. The reasoning steps are segmented either by GPT-5 (left) or by heuristic rules (right). See Figure E.1 for results based on the maximum number of reasoning steps used for interpolation.

Given the strong alignment between early reasoning steps—particularly the first—and the final conclusion, we hypothesize that the first step may significantly influence whether the reasoning model can arrive at a correct prediction.

3.2 CORRELATION BETWEEN THE FIRST REASONING STEP AND THE FINAL PREDICTION

Given that the first reasoning step closely resembles the final conclusion, we investigate whether the essential reasoning required for the final prediction is already encapsulated in the first step. To this end, we analyze the prediction when conditioned solely on the first reasoning step. Specifically, we compute $c_1 = \mathcal{M}(p\text{<think>}t_1\text{</think>})$, and derive the corresponding prediction $q_1 = g(c_1)$, which we compare against the ground truth a. Based on this comparison, we categorize each first reasoning step as either `first correct` (if $q_1 = a$) or `first incorrect` (otherwise).

Experimental setup. To better analyze the correlation, we sample 64 CoT traces per question using the same datasets in §3.1. We exclude questions for which all 64 CoT traces result in either correct or incorrect predictions, as these are considered either too easy or too difficult, respectively, yielding 38 questions for DS-R1-Qwen-7B and 37 for Qwen3-8B. For each remaining question and its corresponding first reasoning step t_1, we obtain the initial prediction q_1 as previously described. While GPT-5 provides more reliable segmentation, it is costly and difficult to reproduce. We therefore adopt the heuristic segmentation method in all subsequent experiments, which is shown to have compariable results with GPT5 segmentation in §3.1. To better visualize the final outcomes, we categorize the questions into three groups based on the pass@1 accuracy of the final prediction q_T,[3] corresponding to the intervals (0, 33%], (33%, 67%], and (67%, 100%). A higher pass@1 indicates a

---

[3] That is, the average accuracy across 64 CoT traces.

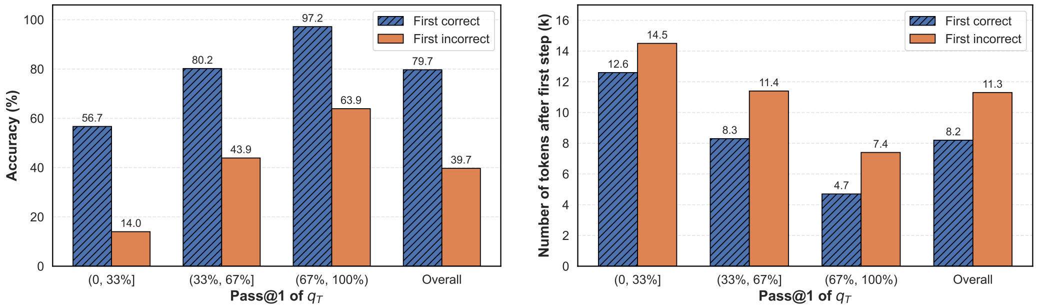


Figure 3: Accuracy and number of tokens on DS-R1-Qwen-7B. **Left:** The relationship between the accuracy of the final prediction (q_T) and the correctness of the prediction solely based on the first reasoning step (q_1) across different difficulty levels. If q_1 is incorrect, q_T is more likely incorrect. **Right:** The number of tokens used for the final prediction after the first reasoning step t_1, i.e., the number of tokens used for $[t_2, t_3, ..., t_T]$. Although q_1 is correct, the model still consumes a large amount of tokens for the following reasoning steps–overthinking.

simpler question. This grouping allows us to assess whether our observations hold consistently across varying levels of question difficulty.

Result. As shown in Figure 3 (Left), the commonly assumed self-correction capability of reasoning models appears to be overstated. When the first reasoning step is incorrect, the model's final prediction is also likely to be incorrect. On average, final prediction accuracy drops by 40% when the first reasoning step is incorrect, with the most substantial decrease (43%) occurring for difficult questions (0–33% range) and a notable decline (33%) even for easier questions (67–100% range). In addition, we also compute the Pearson correlation between the correctness of the first prediction p_1 and the final prediction p_T over all questions. The coefficient $r = 0.60$ and p-value $p = 0.0$ denote a moderately strong positive correlation. All these results underscore the pivotal role of the first reasoning step in steering the model toward a correct final answer, particularly in more complex instances where recovery from early mistakes is more challenging. Extending this analysis to DeepSeek and Qwen models of different sizes yields consistent trends: final accuracy remains substantially higher when the first step is correct, and the accuracy gap persists as model scale increases (Figure F.2).

Figure 3 (Right) further illustrates the model's tendency toward overthinking. Even when the first reasoning step is sufficient to arrive at the correct answer, the model continues to generate a substantial number of additional reasoning tokens—the same scale in length to those generated following an incorrect first step. Both findings are also observed for Qwen3-8B in Figure F.1, reinforcing this pattern across models.

3.3 MINOR PERTURBATION TO THE CORRECT FIRST STEP LEADS TO SIGNIFICANT LOSS

Building on our findings in §3.2, which demonstrate a positive correlation between the model's first and final predictions, we further investigate the significance of the first reasoning step by introducing minor perturbations. Specifically, we slightly alter an initially correct reasoning step and provide it as input to the model to assess whether it can recover from such errors.

Experimental setup. Unlike §3.2, where we analyze the correctness of the first reasoning step t_1, here we treat the final correct conclusion c_T—which satisfies $q_T = g(c_T) = a$—as the new first reasoning step, denoted t_1'. This choice ensures that the step contains all necessary reasoning for arriving at the correct answer, which cannot be guaranteed for t_1. As illustrated in Figure 3 (Left), an initially correct reasoning step can still lead to an incorrect final prediction. To construct t_1', we apply the following perturbations to c_T (see Appendix I for an example): (1) we remove not only the explicit answer formatting (e.g., `\boxed{a}`) but also any surrounding sentences that may directly disclose or repeat the final answer; (2) the resulting text from (1) is treated as the correct version of t_1' (serving as our baseline); (3) we generate an incorrect version by replacing the correct answer a in the remaining reasoning with $a \pm 1$ or $a \pm 10$.[4]

---

[4] The answer of AIME question is integer in the range of $[0, 999)$.

These perturbations are minimal, as they preserve the core reasoning structure while only altering the final prediction in the incorrect variant. We then combine the prompt p with the modified first reasoning step t'_1 and input it to the model as $\mathcal{M}(p\text{<think>}t'_1\text{Alternatively})$ to assess subsequent reasoning behavior.

Table 1: Perturbation experiments. Reported accuracy (%) with correct vs. incorrect first step. Even minor perturbations cause significant drops.

Model	**Correct (%)**	**Incorrect (%)**
DS-R1-Qwen-1.5B	95.4	64.4
DS-R1-Qwen-7B	94.8	28.5
DS-R1-Qwen-32B	100.0	85.8
Qwen3-1.7B	96.0	46.6
Qwen3-8B	71.4	37.0
Qwen3-30B-A3B	100.0	74.7
Qwen3-235B-A22B	100.0	78.7

Result. As shown in Table 1, we make two key observations: (1) Smaller models rarely reach 100% accuracy even when the first reasoning step is correct, suggesting that they may revise or deviate from their initial reasoning. In contrast, larger models (e.g., DS-R1-32B) consistently achieve 100% accuracy given a correct first step, indicating greater stability. (2) There is a substantial drop in accuracy when the first reasoning step is incorrect, highlighting that even minor errors early in the reasoning process can significantly affect the final prediction. These findings further indicate that the LLM's ability to self-correct has been overestimated.

In this section, we observe that the reasoning model is particularly vulnerable at the initial stage of the reasoning process; an error in the first step can propagate and substantially degrade the final prediction. *Can we develop a method to identify and retain more promising first reasoning steps while discarding suboptimal ones to enhance the overall generation efficiency?*

4 EARLY PRUNING WITH HINT FROM FIRST STEP

In this section, we propose an efficient and straightforward sampling method to identify a promising first reasoning step. By doing so, we can terminate the generation process early when a suboptimal first step is detected, thereby reducing unnecessary computational overhead.

4.1 PROBLEM DEFINITION

In contrast to the notation introduced in §3, we incorporate a random seed ϵ to introduce stochasticity into the sampling process. Specifically, a sampled trace is computed as $t \oplus c_T = \mathcal{M}(p, \epsilon)$. By varying the random seed ϵ^n, we obtain diverse generations, denoted as $t^n \oplus c_T^n = \mathcal{M}(p, \epsilon^n)$, where $t^n = [t_1^n, t_2^n, \ldots, t_T^n]$.[5]

A widely adopted technique for reasoning tasks is majority voting or self-consistency generation (Wang et al., 2022). To promote exploration of the reasoning space, models are typically sampled at high temperatures, resulting in diverse outputs. Majority voting then serves to aggregate these outputs by selecting the most frequent final prediction. Formally, majority voting over K samples is defined as:

$$q_{\text{maj@}K} = \text{mode}(\{q_T^n\}_{n=1}^K) \quad \text{where} \quad q_T^n = g(c_T^n)$$

However, for models generating long CoT traces, majority voting becomes computationally expensive, as it requires sampling N complete reasoning paths independently. In this section, we propose a more efficient approach that samples only M full traces, where $M < N$, while maintaining comparable majority voting performance to the one with N samplings.

4.2 METHODOLOGY

In §3, we demonstrated a strong positive correlation between the first reasoning step and the final prediction. This observation motivates a method that identifies the top M most promising first reasoning steps out of a total of N, and continues generation only for these selected M candidates, while discarding the remaining $(N - M)$.

Let a reasoning model generate N candidate first reasoning step $\{t_1^1, t_1^2, \ldots, t_1^N\}$ from a prompt p with different random seeds $\{\epsilon^n\}_{n=1}^N$. Each t_1^n is the first reasoning step of a full reasoning trajectory.

---

[5] In prior experiments, we sampled 64 CoT traces per question using 64 distinct random seeds $\{\epsilon^n\}_{n=1}^{64}$.

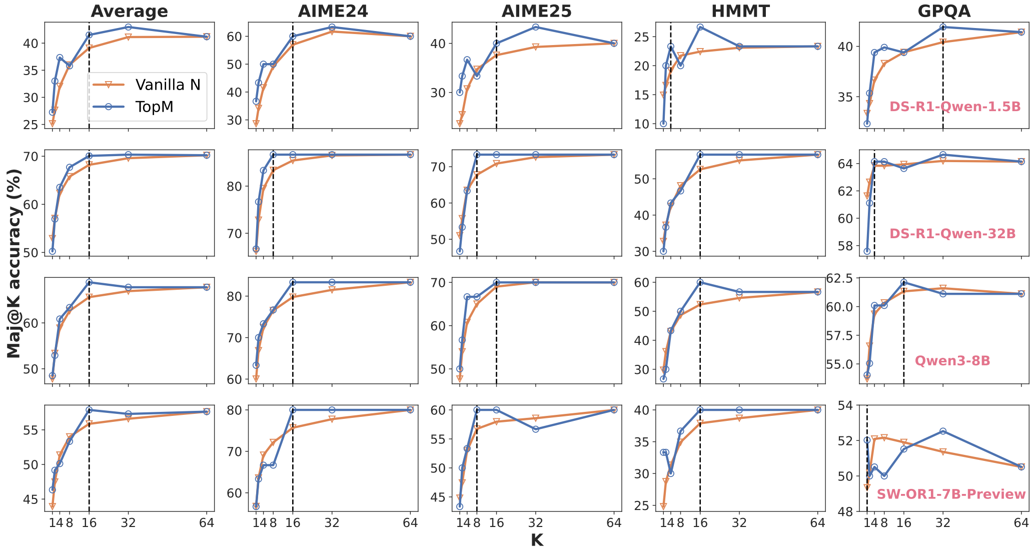


Figure 4: Majority voting accuracy with different number of samplings for four LLMs. The vertical dashed line denotes the smallest M whose accuracy is equal to or larger than the accuracy of $N = 64$.

We define a scoring function $r : t_1^n \to \mathbb{R}$ that estimates the promise of a first step, e.g., rating from a reward model. We then select the top M first steps based on their scores:

$$\mathcal{R}_{\text{top}} = \text{TopM}(\{r(t_1^n)\}_{n=1}^{N})$$

Only the selected M first steps $\{t_1^n \mid n \in \mathcal{R}_{\text{top}}\}$ are used for further multi-step generation. The remaining $(N - M)$ are discarded. Since the first step typically requires only a small number of tokens to generate, this selection approach efficiently reduces computation by avoiding full sampling for the less promising $(N - M)$ candidates.

4.3 EXPERIMENTS

Setup. We evaluate five families of reasoning models—DS-R1-Qwen (Guo et al., 2025), Qwen3 (Yang et al., 2025a), Skywork-OR1 (SW-OR1) (He et al., 2025), Magistral (Rastogi et al., 2025) and GPT-OSS (Agarwal et al., 2025)—on five challenging reasoning benchmarks spanning mathematics, science and programming: AIME24, AIME25 (MAA Committees, 2025), HMMT Feb 2025 (Balunović et al., 2025), the GPQA Diamond set (Rein et al., 2024) and LiveCodeBench[6] (Jain et al., 2024), consisting of 30, 30, 30, 198, and 175 problems, respectively. For decoding, we adopt each model's recommended temperature and top_p, with a maximum generation length of 32K tokens.

We consider values of N and M from the set $\{1, 2, 4, 8, 16, 32, 64\}$. For early pruning, we fix $N = 64$ and select the top M most promising first-step candidates using reward scores predicted by a process reward model (PRM), Qwen2.5-Math-PRM-7B (Zhang et al., 2025). When $M = 64$, the accuracy is exactly the same as the one for $N = 64$, since all candidates are chosen. For PRM, a "step" is defined as a segment of text ending with "`\n\n`", and each step receives an individual reward score. The trajectory t_1^n contains multiple such steps, and we use the score of its final step as the overall reward, $r(t_1^n)$. Notably, using a PRM to score t_1^n is really cheap, because t_1^n is short, and its computation is similar to generate one token with input $p \oplus t_1^n$.

Unlike the definition used in §3, where t_1^n terminates upon generating the keyword "Alternatively", we redefine t_1^n in this section to have a fixed token length, $\text{len}(t_1^n)$. The previous definition made it difficult to control the computational budget for generating first steps, as the trigger phrase might not appear or could occur late in the sequence. By fixing the token length, we achieve precise control over the generation budget when sampling N candidate first steps. By default, $\text{len}(t_1^n) = 512$.

---

[6] Latest release, containing problems from Jan–Apr 2025.

Table 2: Early pruning accuracy and efficiency. We select $M = 16$ first steps with the highest reward scores out of $N = 64$ candidate first steps. The number of tokens used for the 48 discarded first steps is also included for the early pruning method. Early pruning maintains the accuracy, even improves sometimes, while only requiring $< 30\%$ original inference budget.

Model	**Method**	**Maj@K accuracy (%) ↑**					**Avg. #Tokens ↓**
		AIME24	**AIME25**	**HMMT**	**GPQA**	**Avg.**	
DS-R1-Qwen-1.5B	$N = 64$	**60.0**	**40.0**	23.3	**41.4**	41.2	×1.00
	$N = 16$	57.0	37.6	22.4	39.4	39.1 (-2.1)	×0.25
	$M = 16$	**60.0**	**40.0**	**26.7**	39.4	**41.5** (+0.3)	×0.28
DS-R1-Qwen-32B	$N = 64$	**86.7**	**73.3**	**56.7**	**64.1**	**70.2**	×1.00
	$N = 16$	85.4	70.9	52.6	63.9	68.2 (-2.0)	×0.25
	$M = 16$	**86.7**	**73.3**	**56.7**	63.6	70.1 (-0.1)	×0.29
Qwen3-8B	$N = 64$	**83.3**	**70.0**	56.7	61.1	67.8	×1.00
	$N = 16$	79.8	69.0	52.3	61.3	65.6 (-2.2)	×0.25
	$M = 16$	**83.3**	**70.0**	**60.0**	**62.1**	**68.9** (+1.1)	×0.28
SW-OR1-7B	$N = 64$	**80.0**	**60.0**	**40.0**	50.5	57.6	×1.00
	$N = 16$	75.7	58.0	37.9	51.9	55.9 (-1.7)	×0.25
	$M = 16$	**80.0**	**60.0**	**40.0**	**52.5**	**57.9** (+0.3)	×0.29
Magistral-Small	$N = 64$	86.7	**83.3**	**76.7**	**70.2**	**79.2**	×1.00
	$N = 16$	**87.1**	82.6	71.1	69.2	77.5 (-1.7)	×0.25
	$M = 16$	86.7	**83.3**	73.3	**70.2**	78.4 (-0.8)	×0.28
GPT-OSS-20B	$N = 64$	**86.7**	**83.3**	**80.0**	**73.2**	**80.8**	×1.00
	$N = 16$	85.3	81.7	73.3	72.7	78.3 (-2.5)	×0.25
	$M = 16$	**86.7**	**83.3**	**80.0**	**73.2**	**80.8** (-0.0)	×0.27

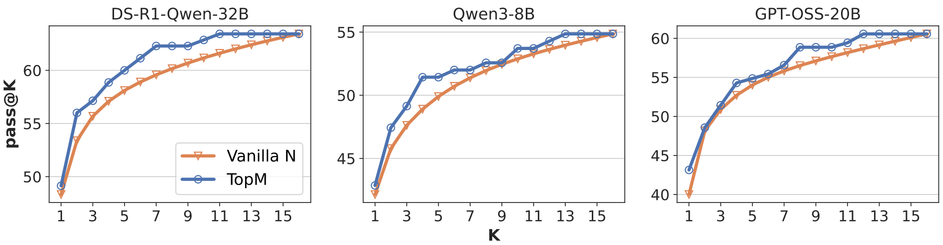


Figure 5: The pass rate on LiveCodeBench, where we set $N = 16$ for early pruning (TopM).

4.3.1 MAIN RESULTS

In Figure 4, we analyze the performance as M varies. We find that selecting the top 16 first reasoning steps from 64 candidates and continuing generation from them achieves accuracy on par with, or even exceeding, conventional sampling with $N = 64$. This trend is consistent across diverse LLMs and benchmarks. Interestingly, for certain cases—such as HMMT on DS-R1-Qwen-1.5B, and AIME24, AIME25, and GPQA on DS-R1-Qwen-32B—using as few as $M \leq 8$ suffices to match the performance obtained with all 64 samples.

Table 2 reports the detailed accuracy and token consumption of different methods. When using $M = 16$, early pruning consistently matches the accuracy of $N = 64$ across a range of LLMs and benchmarks, while substantially outperforming $N = 16$ under a comparable token budget. Notably, for Qwen3-8B, early pruning even yields a 1.1% improvement in average accuracy. Importantly, these gains come at less than 30% of the inference cost of majority voting with $N = 64$, underscoring the strong efficiency advantage of our method.

For the code generation benchmark, LiveCodeBench, where majority voting is not applicable, we present the pass rate in Figure 5. Early pruning consistently surpasses standard sampling given the same number of complete sampling.

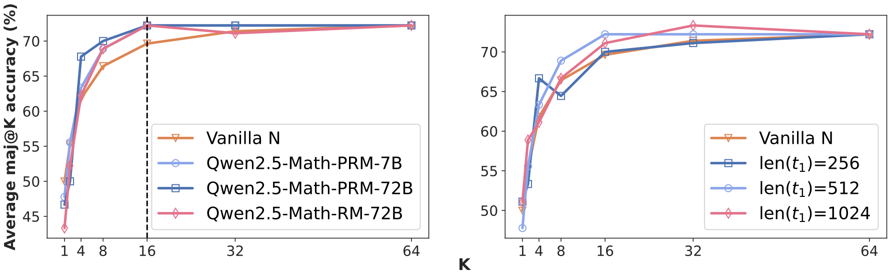


Figure 6: Average accuracy across AIME24, AIME25 and HMMT on DS-R1-Qwen-32B. **Left:** Comparison of reward signals derived from different reward models. The choice of reward model has minimal impact on overall performance. **Right:** Effect of varying the length of the first reasoning step. Using a very short first step, like 256 tokens, leads to suboptimal performance, likely because it provides insufficient reasoning context to effectively evaluate the quality of the step.

4.3.2 ABLATION STUDIES

Here we further validate our default settings.

Choice of reward model. In Figure 6 (Left), we evaluate two additional reward models: a larger PRM, Qwen2.5-Math-PRM-72B, and an outcome reward model, Qwen2.5-Math-RM-72B (Yang et al., 2024). The results indicate that the choice of reward model has minimal impact on performance. Notably, the smaller 7B PRM achieves comparable results, highlighting the efficiency of our approach.

Length of first step. In Figure 6 (Right), we examine the impact of varying the length of the first reasoning step. We observe that the shortest first step (i.e., $\text{len}(t_1^n) = 256$) leads to degraded performance. We hypothesize that shorter t_1^n sequences lack sufficient reasoning content, making them less informative for reliable reward model evaluation. Nevertheless, setting $\text{len}(t_1^n) \geq 512$ tokens yields consistently better performance than vanilla sampling.

Effect of first step split. Table 3 examines how defining the first step influences early pruning performance. The heuristic approach follows the method described in §3. Overall, both definitions yield comparable results. Nevertheless, we recommend using the token-count–based definition, as it provides a clearer way to manage the token budget across candidate first steps. Moreover, trigger phrases that signal step boundaries may vary across LLMs.

Table 3: Average maj@K for early pruning, with first step defined by length or phrase.

Model	len(t_1)=512	Heuristic
DS-R1-Qwen-1.5B	41.5	43.8
DS-R1-Qwen-32B	70.1	70.4
Qwen3-8B	68.9	68.2
SW-OR1-7B	57.9	57.7

Table 4: Time spent by early pruning.

Model	$N = 64$	$M = 16$
DS-R1-Qwen-1.5B	×1.00	×0.27
Qwen3-8B	×1.00	×0.37

Overhead from reward model. Relative to vanilla sampling, early pruning requires scoring the first step with a reward model, which introduces additional overhead. To ensure a fair comparison, we avoid using extra GPUs for deploying the reward model. Instead, our procedure is as follows: (1) load the reasoning model to generate candidate first steps and then offload it; (2) load the reward model on the same GPU to evaluate these steps and offload it; and (3) reload the reasoning model to continue generation from the selected first steps. The timing results are reported in Table 4. Notably, early pruning remains efficient both in terms of token usage and runtime, since evaluating the first step with the reward model is inexpensive—comparable to computing embeddings for the short first steps.

5 CONCLUSION

In this paper, we empirically demonstrate that the first reasoning step plays a critical role in determining the final outcome of a model's reasoning process. Errors introduced early can significantly degrade overall performance. Motivated by this observation, we propose an efficient sampling strategy that identifies and retains high-quality first reasoning steps, reducing the inference computation up to 70% across three model families. These findings suggest that improving or exploiting the very first step is a promising direction for building more accurate and efficient reasoning LLMs.

ETHICS STATEMENT

This work builds on publicly available datasets (AIME24, AIME25, HMMT Feb 2025, GPQA, and LiveCodeBench) and openly released models (e.g., DeepSeek, Qwen), following their respective licenses. We report exact model identifiers and dataset subsets to facilitate reproducibility. We acknowledge that improvements in efficiency may reduce computational cost barriers, which could enable misuse (e.g., large-scale automated disinformation). However, our focus is on rigorous evaluation of reasoning processes and improving efficiency for legitimate research purposes. We encourage responsible use of our findings and emphasize that the proposed methods should be applied in research and educational contexts, not for harmful applications.

REPRODUCIBILITY STATEMENT

Our code builds on widely used open-source libraries for training and evaluation of LLMs, including vLLM and Hugging Face Transformers. All experiments are conducted on datasets available through Hugging Face, and we adopt default prompt formats unless otherwise specified. We report the exact model names and versions used, including both open-source and closed-source models.

For datasets, we evaluate on AIME24, AIME25, HMMT Feb 2025, GPQA, and LiveCodeBench, specifying the subsets employed in our experiments. All datasets are publicly available. We will release our code (and configuration details) upon acceptance of this paper to further support reproducibility.

We encourage the community to design and release additional evaluation datasets that are explicitly intended for research use and distributed under open licenses.

REFERENCES

Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

Pranjal Aggarwal, Aman Madaan, Yiming Yang, et al. Let's sample step by step: Adaptive-consistency for efficient reasoning and coding with llms. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12375–12396, 2023.

Anthropic. Claude 3.7 sonnet system card. 2025. URL `https://api.semanticscholar.org/CorpusID:276612236`.

Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. Matharena: Evaluating llms on uncontaminated math competitions, February 2025. URL `https://matharena.ai/`.

Federico Barbero, Alvaro Arroyo, Xiangming Gu, Christos Perivolaropoulos, Michael Bronstein, Petar Veličković, and Razvan Pascanu. Why do llms attend to the first token? *arXiv preprint arXiv:2504.02732*, 2025.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

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

Xinyi Chen, Baohao Liao, Jirui Qi, Panagiotis Eustratiadis, Christof Monz, Arianna Bisazza, and Maarten de Rijke. The SIFo benchmark: Investigating the sequential instruction following ability of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 1691–1706. Association for Computational Linguistics, 2024b. URL `https://aclanthology.org/2024.findings-emnlp.92/`.

Cheng-Han Chiang and Hung-yi Lee. Over-reasoning and redundant calculation of large language models. In Yvette Graham and Matthew Purver (eds.), *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 161–169, St. Julian's, Malta, March 2024. Association for Computational Linguistics. URL `https://aclanthology.org/2024.eacl-short.15/`.

Thibault Formal, Benjamin Piwowarski, and Stéphane Clinchant. Splade: Sparse lexical and expansion model for first stage ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2288–2292, 2021.

Thibault Formal, Carlos Lassance, Benjamin Piwowarski, and Stéphane Clinchant. From distillation to hard negative sampling: Making sparse neural ir models more effective, 2022. URL `https://arxiv.org/abs/2205.04733`.

Yichao Fu, Junda Chen, Yonghao Zhuang, Zheyu Fu, Ion Stoica, and Hao Zhang. Reasoning without self-doubt: More efficient chain-of-thought through certainty probing. In *ICLR 2025 Workshop on Foundation Models in the Wild*, 2025a. URL `https://openreview.net/forum?id=wpK4IMJfdX`.

Yichao Fu, Xuewei Wang, Yuandong Tian, and Jiawei Zhao. Deep think with confidence. *arXiv preprint arXiv:2508.15260*, 2025b.

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

Jujie He, Jiacai Liu, Chris Yuhao Liu, Rui Yan, Chaojie Wang, Peng Cheng, Xiaoyu Zhang, Fuxiang Zhang, Jiacheng Xu, Wei Shen, Siyuan Li, Liang Zeng, Tianwen Wei, Cheng Cheng, Yang Liu, and Yahui Zhou. Skywork open reasoner series. `https://capricious-hydrogen-41c.notion.site/Skywork-Open-Reaonser-Series-1d0bc9ae823a80459b46c149e4f51680`, 2025. Notion Blog.

Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409*, 2017.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. Thinkprune: Pruning long chain-of-thought of llms via reinforcement learning. *arXiv preprint arXiv:2504.01296*, 2025.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Yiwei Li, Peiwen Yuan, Shaoxiong Feng, Boyuan Pan, Xinglin Wang, Bin Sun, Heda Wang, and Kan Li. Escape sky-high cost: Early-stopping self-consistency for multi-step reasoning. In *ICLR*, 2024.

Baohao Liao and Christof Monz. 3-in-1: 2d rotary adaptation for efficient finetuning, efficient batching and composability. *arXiv preprint arXiv:2409.00119*, 2024.

Baohao Liao, Hanze Dong, Yuhui Xu, Doyen Sahoo, Christof Monz, Junnan Li, and Caiming Xiong. Fractured chain-of-thought reasoning. *arXiv preprint arXiv:2505.12992*, 2025a.

Baohao Liao, Yuhui Xu, Hanze Dong, Junnan Li, Christof Monz, Silvio Savarese, Doyen Sahoo, and Caiming Xiong. Reward-guided speculative decoding for efficient LLM reasoning. *arXiv preprint arXiv:2501.19324*, 2025b.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024a. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Yijin Liu, Xianfeng Zeng, Chenze Shao, Fandong Meng, and Jie Zhou. Instruction position matters in sequence generation with large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 11652–11663, Bangkok, Thailand, August 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.693. URL https://aclanthology.org/2024.findings-acl.693/.

Haotian Luo, Li Shen, Haiying He, Yibo Wang, Shiwei Liu, Wei Li, Naiqiang Tan, Xiaochun Cao, and Dacheng Tao. O1-pruner: Length-harmonizing fine-tuning for o1-like reasoning pruning. *arXiv preprint arXiv:2501.12570*, 2025.

MAA Committees. AIME Problems and Solutions. https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions, 2025.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.

Pouya Pezeshkpour and Estevam Hruschka. Large language models sensitivity to the order of options in multiple-choice questions. *arXiv preprint arXiv:2308.11483*, 2023.

Guanghui Qin, Yukun Feng, and Benjamin Van Durme. The NLP task effectiveness of long-range transformers. In Andreas Vlachos and Isabelle Augenstein (eds.), *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 3774–3790. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.eacl-main.273. URL https://aclanthology.org/2023.eacl-main.273/.

Abhinav Rastogi, Albert Q Jiang, Andy Lo, Gabrielle Berrada, Guillaume Lample, Jason Rute, Joep Barmentlo, Karmesh Yadav, Kartik Khandelwal, Khyathi Raghavi Chandu, et al. Magistral. *arXiv preprint arXiv:2506.10910*, 2025.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2019. URL https://arxiv.org/abs/1908.10084.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. GPQA: A graduate-level Google-proof Q&A benchmark. In *First Conference on Language Modeling*, 2024.

Ruoqi Shen, Sébastien Bubeck, Ronen Eldan, Yin Tat Lee, Yuanzhi Li, and Yi Zhang. Positional description matters for transformers arithmetic. *arXiv preprint arXiv:2311.14737*, 2023.

Lin Shi, Chiyu Ma, Wenhua Liang, Weicheng Ma, and Soroush Vosoughi. Judging the judges: A systematic investigation of position bias in pairwise comparative assessments by llms. *arXiv preprint arXiv:2406.07791*, 2024.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

Qwen Team. Qwen3, April 2025. URL `https://qwenlm.github.io/blog/qwen3/`.

Guangya Wan, Yuqi Wu, Jie Chen, and Sheng Li. Reasoning aware self-consistency: Leveraging reasoning paths for efficient LLM sampling. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *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)*, pp. 3613–3635, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.184. URL `https://aclanthology.org/2025.naacl-long.184/`.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9440–9450. Association for Computational Linguistics, 2024. URL `https://aclanthology.org/2024.acl-long.511/`.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 2020. URL `https://arxiv.org/abs/2002.10957`.

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

Yue Wang, Qiuzhi Liu, Jiahao Xu, Tian Liang, Xingyu Chen, Zhiwei He, Linfeng Song, Dian Yu, Juntao Li, Zhuosheng Zhang, et al. Thoughts are all over the place: On the underthinking of o1-like llms. *arXiv preprint arXiv:2501.18585*, 2025a.

Zezhong Wang, Xingshan Zeng, Weiwen Liu, Yufei Wang, Liangyou Li, Yasheng Wang, Lifeng Shang, Xin Jiang, Qun Liu, and Kam-Fai Wong. Chain-of-probe: Examining the necessity and accuracy of CoT step-by-step. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 2586–2606. Association for Computational Linguistics, 2025b. ISBN 979-8-89176-195-7. URL `https://aclanthology.org/2025.findings-naacl.140/`.

Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, Haosheng Zou, Yongchao Deng, Shousheng Jia, and Xiangzheng Zhang. Light-r1: Curriculum sft, dpo and rl for long cot from scratch and beyond. *arXiv preprint arXiv:2503.10460*, 2025.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=NG7sS51zVF`.

Wei Xiong, Wenting Zhao, Weizhe Yuan, Olga Golovneva, Tong Zhang, Jason Weston, and Sainbayar Sukhbaatar. Stepwiser: Stepwise generative judges for wiser reasoning. *arXiv preprint arXiv:2508.19229*, 2025.

Mingfeng Xue, Dayiheng Liu, Wenqiang Lei, Xingzhang Ren, Baosong Yang, Jun Xie, Yidan Zhang, Dezhong Peng, and Jiancheng Lv. Dynamic voting for efficient reasoning in large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 3085–3104, 2023.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Zheng Lin, Li Cao, and Weiping Wang. Dynamic early exit in reasoning models. *arXiv preprint arXiv:2504.15895*, 2025b.

Yusen Zhang, Sarkar Snigdha Sarathi Das, and Rui Zhang. Verbosity neq veracity: Demystify verbosity compensation behavior of large language models. *arXiv preprint arXiv:2411.07858*, 2024a.

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

Zhenyu Zhang, Runjin Chen, Shiwei Liu, Zhewei Yao, Olatunji Ruwase, Beidi Chen, Xiaoxia Wu, and Zhangyang Wang. Found in the middle: How language models use long contexts better via plug-and-play positional encoding. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024b.

Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. Large language models are not robust multiple choice selectors. In *The Twelfth International Conference on Learning Representations*, 2024.

A LIMITATIONS

Our study has several limitations. First, we restrict our analysis to open-source long-CoT models. This excludes state-of-the-art closed-source systems such as GPT-4 and Claude, where reasoning traces are not accessible and controlled interventions are difficult. Focusing on open-source models ensures transparency and reproducibility, and the insights we uncover are in principle model-agnostic. Second, our experiments focus on mathematical, code generation, and scientific reasoning benchmarks. Although these datasets are limited in scope, they are representative of structured multi-step reasoning and provide a controlled setting for evaluating first-step effects. Importantly, our approach is flexible and could be extended to broader domains such as commonsense reasoning or creative writing with task-specific modifications.

B THE USE OF LARGE LANGUAGE MODELS

We used LLMs (e.g., GPT-5) to assist with segmenting reasoning traces into steps as part of our pipeline (see Section 3.1). We also used ChatGPT to generate plotting scripts and to make minor improvements in grammar and phrasing of the manuscript. No LLM was used for research ideation, experiment design, dataset creation, or for producing original scientific contributions. All conceptual and methodological contributions are from the authors.

C KEYWORDS FOR REASONING STEP SEGMENTATION

Good reasoning switching keywords should signal a clear switch in reasoning, while should not occur too frequently which often indicate only minor digressions, or too rare to be reliable delimiters. We collect a list of reasoning switching keywords from papers (Muennighoff et al., 2025; Hou et al., 2025). To assess which reasoning switch keywords could serve a similar role, we compute their average frequency per question in model outputs. After cosidering the frequencies and manually check the segment results, we found that *"alternatively"* strikes a balance, appearing often enough to capture step boundaries while still reflecting shifts in reasoning, for DS-R1-Qwen-7B and Qwen3-8B. We uses both *"alternatively"* and *"wait"* for Claude-3.7-Sonnet, GPT-OSS-20B and Magistral-Small.

Table C.1: Average frequency per question of discourse markers in model outputs.

	but	wait	alternatively	hmm	hold on	let me confirm	however
DS-R1-Qwen-7B	82	72	14	7	3	<1	10.1
Qwen3-8B	91	66	20	6	1	<1	10.4
Claude-3.7-Sonnet	28.5	22.6	0.2	6.1	0.7	0.0	0.9
Magistral-Small	47.1	54.8	3.0	0.0	0.9	0.2	1.4
GPT-OSS-20B	171.6	60.5	4.9	1.2	0.7	0.0	5.9

D MODEL IDENTIFIERS

For reproducibility, we provide the exact model identifiers used in our experiments. The models are: DeepSeek-R1-Distill-Qwen-7B (`deepseek-ai/DeepSeek-R1-Distill-Qwen-7B`), Qwen3-8B (`Qwen/Qwen3-8B`), Claude 3.7 (`claude-3-7-sonnet-20250219`, accessed via the Claude API), GPT-OSS-20B (`openai/gpt-oss-20b`), and Magistral-Small (`mistralai/Magistral-Small-2509`).

E ADDITIONAL SIMILARITY ANALYSES

We report the cosine similarity between the embedding of the i-th reasoning step t_i and the final conclusion c_T, interpolated using the maximum number of reasoning steps (Figure E.1).

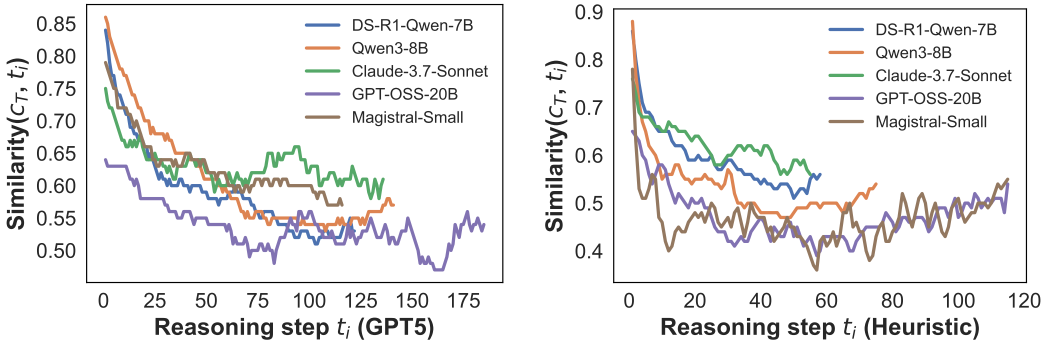


Figure E.1: Cosine similarity between the embeddings of the i-th reasoning step t_i and the final conclusion c_T, using the maximum number of reasoning steps for interpolation. The reasoning steps are segmented either by GPT-5 (left) or by heuristic rules (right).

In addition, we report SPLADE similarity (Formal et al., 2021), which provides a sparse lexical–semantic relevance signal by expanding tokens into weighted lexical features. We compute SPLADE similarity using the `naver/splade-cocondenser-ensembledistil` model (Formal et al., 2022), following standard practice. As shown in Figure E.2, the trends closely mirror those observed under cosine similarity. These results serve as complementary checks, confirming that our findings are not tied to dense embedding models and hold across both dense and sparse similarity measures.

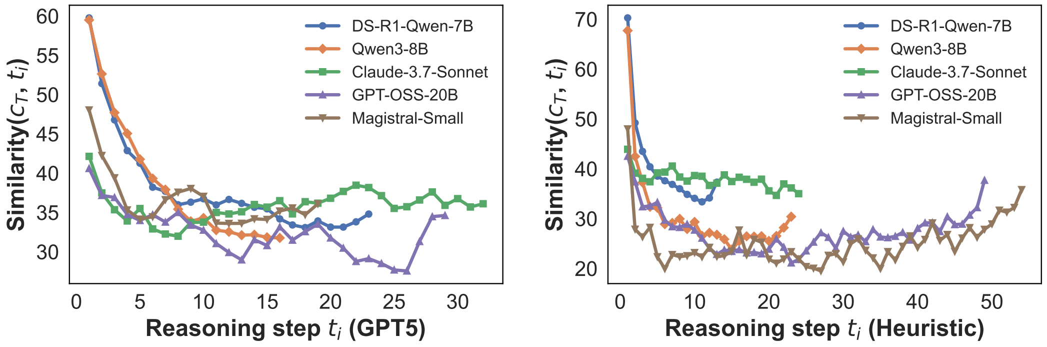


Figure E.2: SPLADE similarity between the representations of the i-th reasoning step t_i and the final conclusion c_T, using the maximum number of reasoning steps for interpolation. The reasoning steps are segmented either by GPT-5 (left) or by heuristic rules (right).

F CORRELATION BETWEEN FIRST REASONING STEP AND FINAL PREDICTION

In the main text, we reported results on DS-R1-Qwen-7B. Here we extend the analysis to Qwen3-8B (Figure F.1) and further include DeepSeek and Qwen3 models of different sizes (Figure F.2). These additional results verify that the correlation between the first reasoning step and the final prediction persists as model size scales.

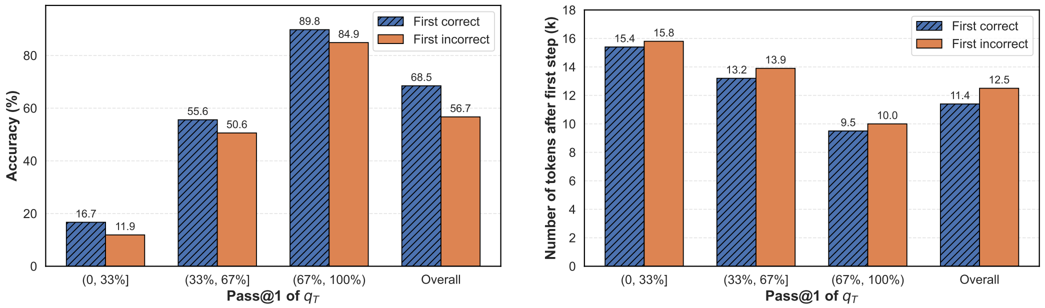


Figure F.1: Accuracy and number of tokens on Qwen3-8B. **Left:** The relationship between the accuracy of the final prediction (q_T) and the correctness of the prediction solely based on the first reasoning step (q_1) across different difficulty levels. If q_1 is incorrect, q_T is more likely incorrect. **Right:** The number of tokens used for the final prediction after the first reasoning step t_1, i.e., the number of tokens used for $[t_2, t_3, ..., t_T]$. Although q_1 is correct, the model still consumes a large amount of tokens for the following reasoning steps–overthinking.

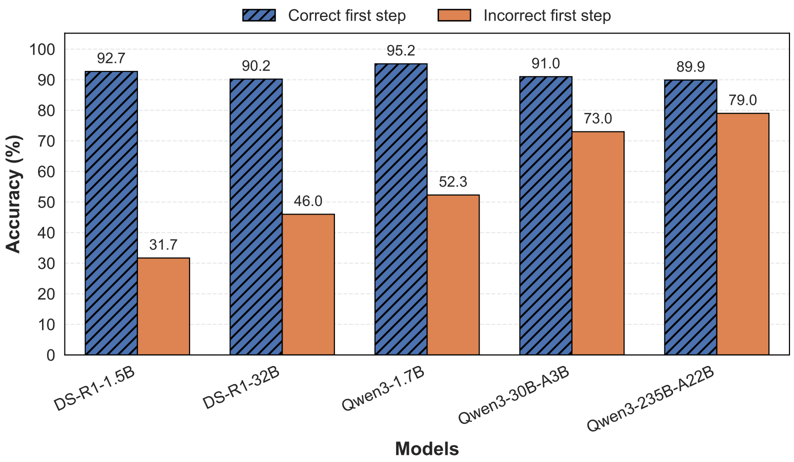


Figure F.2: More results from Qwen and DeepSeek models of different sizes. We report the relationship between the accuracy of the final prediction (q_T) and the correctness of the first reasoning step (q_1). If q_1 is incorrect, q_T is more likely to be incorrect.

G PROMPT EXAMPLES

The template for using the DeepSeek-R1 model is provided below. We apply the same structure to other open-source thinking models, with slight adjustments for model-specific input and output formats.

Prompt example for DeepSeek-R1:

```
<|begin_of_sentence|>Please reason step by step, and put your
final answer within \boxed{}.

<|User|>[Question]<|Assistant|><think>
```

H EXAMPLE OF A INCORRECT FIRST STEP

Figure H.1 illustrates a case where the initial reasoning step is incorrect.

Example of incorrect first step

Question: Four unit squares form a 2×2 grid. Each of the 12 unit line segments forming the sides of the squares is colored either red or blue in such a way that each unit square has 2 red sides and 2 blue sides. One example is shown below (red is solid, blue is dashed). Find the number of such colorings.
[asy] size(4cm); defaultpen(linewidth(1.2)); draw((0, 0) – (2, 0) – (2, 1)); draw((0, 1) – (1, 1) – (1, 2) – (2,2)); draw((0, 0) – (0, 1), dotted); draw((1, 0) – (1, 1) – (2, 1) – (2, 2), dotted); draw((0, 1) – (0, 2) – (1, 2), dotted); [/asy]
First reason: Alright, so I have this problem about coloring the sides of a 2x2 grid of squares. Each square has four sides, and each of these sides is colored either red or blue. The condition is that each individual square must have exactly two red sides and two blue sides. The question is asking for the number of such colorings.
First, let me visualize the grid. It's a 2x2 grid, so there are four squares arranged in a larger square. Each square has four sides, and each side is shared with another square except for the outermost sides. But in the problem, it says there are 12 unit line segments forming the sides of the squares. Wait, let me check that: a 2x2 grid has 3 horizontal lines and 3 vertical lines, each of length 2 units. So, each horizontal line is divided into two segments, right? So, the total number of horizontal segments is 3 rows times 2 segments per row, which is 6, and similarly, the vertical segments are also 3 columns times 2 segments per column, which is another 6. So, 6 + 6 = 12 segments in total. Okay, that makes sense.
Each of these 12 segments can be colored either red or blue. However, the constraint is that each of the four squares must have exactly two red sides and two blue sides. So, even though each segment can be colored independently, we have these local constraints for each square.
I think a good way to approach this problem is by considering each square and the possible colorings for its sides, then seeing how they can fit together without violating the constraints. However, since the coloring of one square's side affects the adjacent squares, this could get complicated.
Maybe I can model this as a graph where each segment is a node, and the squares impose constraints on the nodes (i.e., exactly two red and two blue edges around each square). But I'm not too familiar with graph theory, so maybe that's not the best approach.
Mistakes in first reasoning step: Mistake 1 incorrectly assumes that in a 2×2 grid, each corner square has two shared and two unshared sides, without properly accounting for edge distribution. Mistake 2 introduces unnecessary abstraction by treating segments as "nodes," which obscures the natural structure: the segments themselves are the edges whose colours we choose, and each square simply places a rule on its four incident edges. Working directly with those four constraints is far cleaner than introducing an extra level of abstraction.

Figure H.1: Example of incorrect first step.

I EXAMPLES FOR MINOR PERTURBATION

Original final conclusion

To solve the problem, we are given triangle ABC with side lengths $AB = 5$, $BC = 9$, and $AC = 10$.
[...]
Step 4: Compute AP. Since P lies on line segment AD, and D lies outside the circle, the point P lies between A and D. Therefore:

$$AP = DA - DP = \frac{325}{22} - \frac{2025}{286}.$$

Find a common denominator:

$$\frac{325}{22} = \frac{4225}{286} \Rightarrow AP = \frac{4225 - 2025}{286} = \frac{2200}{286} = \frac{100}{13}.$$

Final Step: Compute $m + n$.
The reduced form of AP is $\frac{100}{13}$, so:

$$m = 100, \quad n = 13 \Rightarrow m + n = \texttt{\textbackslash boxed\{113\}}.$$

Final Answer: `\boxed{113}`

New correct first reason

To solve the problem, we are given triangle ABC with side lengths $AB = 5$, $BC = 9$, and $AC = 10$.
[...]
Step 4: Compute AP. Since P lies on line segment AD, and D lies outside the circle, the point P lies between A and D. Therefore:

$$AP = DA - DP = \frac{325}{22} - \frac{2025}{286}.$$

Find a common denominator:

$$\frac{325}{22} = \frac{4225}{286} \Rightarrow AP = \frac{4225 - 2025}{286} = \frac{2200}{286} = \frac{100}{13}.$$

Final Step: Compute $m + n$.
The reduced form of AP is $\frac{100}{13}$, so:

$$m = 100, \quad n = 13 \Rightarrow m + n = \texttt{113}.$$

New incorrect first reason

To solve the problem, we are given triangle ABC with side lengths $AB = 5$, $BC = 9$, and $AC = 10$.
[...]
Step 4: Compute AP. Since P lies on line segment AD, and D lies outside the circle, the point P lies between A and D. Therefore:

$$AP = DA - DP = \frac{325}{22} - \frac{2025}{286}.$$

Find a common denominator:

$$\frac{325}{22} = \frac{4225}{286} \Rightarrow AP = \frac{4225 - 2025}{286} = \frac{2200}{286} = \frac{100}{13}.$$

Final Step: Compute $m + n$.
The reduced form of AP is $\frac{100}{13}$, so:

$$m = 100, \quad n = 13 \Rightarrow m + n = \texttt{103}.$$

Figure I.1: Illustration of a targeted perturbation to create correct and incorrect first reasoning steps, as discussed in §3.3. The modified context is highlighted in orange.