FoE: Forest of Errors Makes the First Solution the Best in Large Reasoning Models

Kehan Jiang[1*] **Haonan Dong**[2*] **Zhaolu Kang**[1]
Zhengzhou Zhu[1] **Guojie Song**[2†]
[1]School of Software and Microelectronics, Peking University
[2]State Key Laboratory of General Artificial Intelligence,
School of Intelligence Science and Technology, Peking University
[*]Equal contribution [†]Corresponding author
jiangkh5521@gmail.com, gjsong@pku.edu.cn

Abstract

Recent Large Reasoning Models (LRMs) like DeepSeek-R1 have demonstrated remarkable success in complex reasoning tasks, exhibiting human-like patterns in exploring multiple alternative solutions. Upon closer inspection, however, we uncover a surprising phenomenon: **`The First is The Best`**, where alternative solutions are not merely suboptimal but potentially detrimental. This observation *challenges widely accepted test-time scaling laws*, leading us to hypothesize that ***errors within the reasoning path scale concurrently with test time***. Through comprehensive empirical analysis, we characterize errors as a forest-structured Forest of Errors (**`FoE`**) and conclude that ***FoE makes the First the Best***, which is underpinned by rigorous theoretical analysis. Leveraging these insights, we propose **`RED`**, a self-guided efficient reasoning framework comprising two components: I) *Refining First*, which suppresses **`FoE`** growth in the first solution; and II) *Discarding Subs*, which prunes subsequent **`FoE`** via dual-consistency. Extensive experiments across five benchmarks and six backbone models demonstrate that **`RED`** outperforms eight competitive baselines, achieving performance gains of up to 19.0% while reducing token consumption by 37.7% $\sim$ 70.4%. Moreover, comparative experiments on **`FoE`** metrics shed light on how **`RED`** achieves effectiveness.

1 Introduction

Reasoning capability stands as the cornerstone of human intelligence (Johnson-Laird, 2010). Recently, LLMs have achieved significant advancements in reasoning, demonstrating immense potential across mathematical (Suzgun et al., 2023), scientific (Lewkowycz et al., 2022), and coding tasks (Chen et al., 2021). This progress is largely attributed to the evolution of the Chain-of-Thought (CoT) (Wei et al., 2022) research line, which specifically decomposes complex tasks into step-by-step reasoning processes. Furthermore, the advent of DeepSeek-R1 (DeepSeek-AI et al., 2025) marks a paradigm shift for Large Reasoning Models (LRMs). Powered by Reinforcement Learning (RL)-based training frameworks (Yue et al., 2025; Wang et al., 2025a; Yu et al., 2025), R1-like models tend to generate more extensive responses while exhibiting an "aha-moment", characterized by self-verification, reflection, and the exploration of alternative methodologies during the reasoning process (DeepSeek-AI et al., 2025).

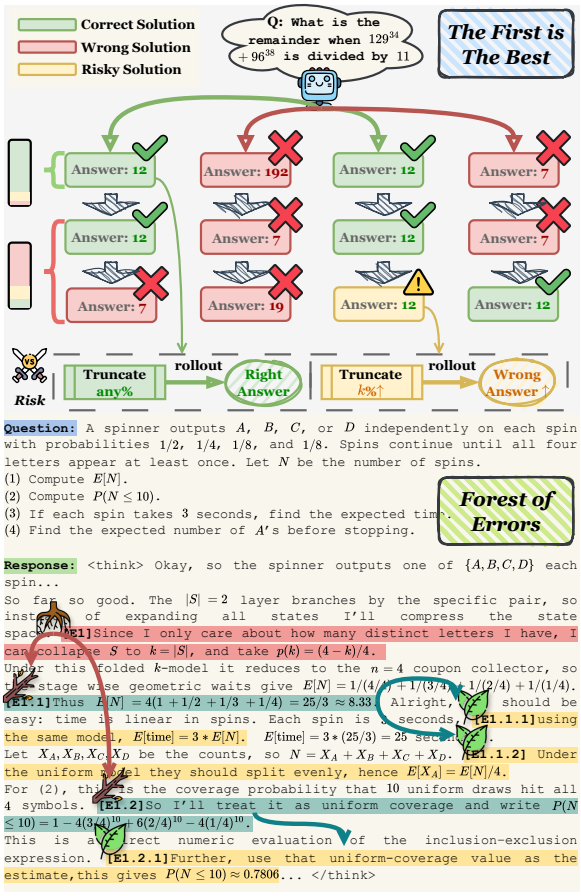


Figure 1: (***Upper***) **`The First is The Best`**. (***Lower***) Forest of Errors.

Phenomenon ❶. Despite the success of R1-like models in reasoning tasks achieved through RL-driven self-evolution (OpenAI et al., 2024; Team,

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 24591–24629
July 2-7, 2026

2025), we observe that their frequent exploration of multiple potential solutions results in excessively long CoT, leading to a substantial waste of computational resources (Chen et al., 2024). This observation prompts a critical question: *Is the exploration of multiple solutions truly necessary?* Upon closer inspection of the solutions generated during the reasoning process, we uncover a surprising phenomenon: `The First is The Best`. We denote the first solution generated by LRM as `First` and the subsequent solutions as `Subs`. Specifically, as illustrated in Figure 1(***Upper***) and Table 1, we find that: ♣ `First` is optimal in up to 93.7% of cases; ♦ `Subs` fails to rectify an erroneous `First` with a 75.4% $\sim$ 82.8% probability; ♥ more concerningly, there is a huge probability (up to 21.2%) that `Subs` misleads a potentially correct `First` towards an incorrect answer; and ♠ even when both `First` and `Subs` are correct, `Subs` harbors a substantial latent risk of error. To some extent, it *challenges the widely accepted test-time scaling law* (Snell et al., 2024) and motivates us to hypothesize that ***as test-time scales up, errors within the reasoning path may scale up concurrently***.

Phenomenon ❷. To rigorously validate this hypothesis, we conduct an in-depth analysis of errors emerging during the reasoning process. As depicted in Figure 1 (***Lower***), we observe that errors propagate diffusely, stemming from multiple root causes and ultimately manifesting as a forest-like structure, which we term the Forest of Errors (`FoE`). Through qualitative and quantitative analyses of `FoE`, we find that: ♣ There exists a strong dependencies between parent and child nodes within `FoE`, with root error nodes playing a pivotal role in the overall growth of the error structure; ♦ The scale of `FoE` in `First` is significantly smaller than that in `Subs`; ♥ The generation of error nodes is closely correlated with entropy and entropy variance, with `Subs` exhibiting a higher propensity for generating error nodes compared to `First`; and ♠ The self-reflection mechanisms of LRMs appear ineffective in pruning `FoE`. Extensive empirical experiments yield a total of 5 observations (**Obs.**), consistently showing that errors accumulate with reasoning length, validating that ***FoE makes the First the Best***.

Practical Method. Building upon the systematic empirical analysis above, we further provide a rigorous mathematical analysis and proof of ***FoE makes the First the Best*** through the lens of probabilistic branching process theory. Furthermore, synthesizing insights from Phenomena ❶ and ❷, we propose `RED` (`Refine First and Discard Subs`), an efficient reasoning method based on self-guidance. Specifically, `RED` comprises two components: **I)** ***Refining `First`***, inspired by the pivotal role of root nodes in `FoE` and the strong correlation between error generation and entropy statistics (entropy and its variance), we employ an entropy-based intervention mechanism at positions prone to root errors within `First`, thus ensuring a superior one. **II)** ***Discarding `Subs`***, given our finding that `Subs` not only fails to improve performance but also carries the risk of misleading the model, we implement a dual-consistency-based early stopping strategy. This prevents inferior subsequent solutions from negatively impacting the final answer. Our contributions are summarized as follows:

- ♥ ***Phenomenon Discovery.*** We identify two critical phenomena: `The First is The Best` and `FoE`. These findings reveal a pivotal insight: as test-time scales up, errors within the reasoning path scale up concurrently. This leads to the conclusion that FoE renders the first the best.
- ♥ ***Insightful Analysis.*** Through qualitative and quantitative experiments alongside theoretical derivation, we derive five insightful observations for future reasoning models. Furthermore, we established a `FoE`-based probabilistic framework to theoretically validate the optimality of the `First` (detailed in Appendix K).
- ♥ ***Practical Method.*** Building upon these empirical foundations, we propose `RED`, a self-guided efficient reasoning method designed to optimize `First` while pruning `Subs`. Extensive experiments across four datasets and six backbone models demonstrate that `RED` outperforms seven competitive baselines, delivering up to 19.0% accuracy gains and 37.7% $\sim$ 70.4% token reduction. Furthermore, evaluations on `FoE` metrics confirm that `RED` effectively eliminates `FoE`.

2 `The First is The Best`

In this section, we investigate and identify the `The First is The Best` phenomenon across multiple datasets and models, as shown in Table 1. We perform a statistical analysis of the correctness relationship between `First` and `Subs`.

Obs.❶ When the `First` is incorrect, an average of 75.4%-82.8% of `Subs` persist in being incorrect. This indicates that `Subs` struggle to rectify an

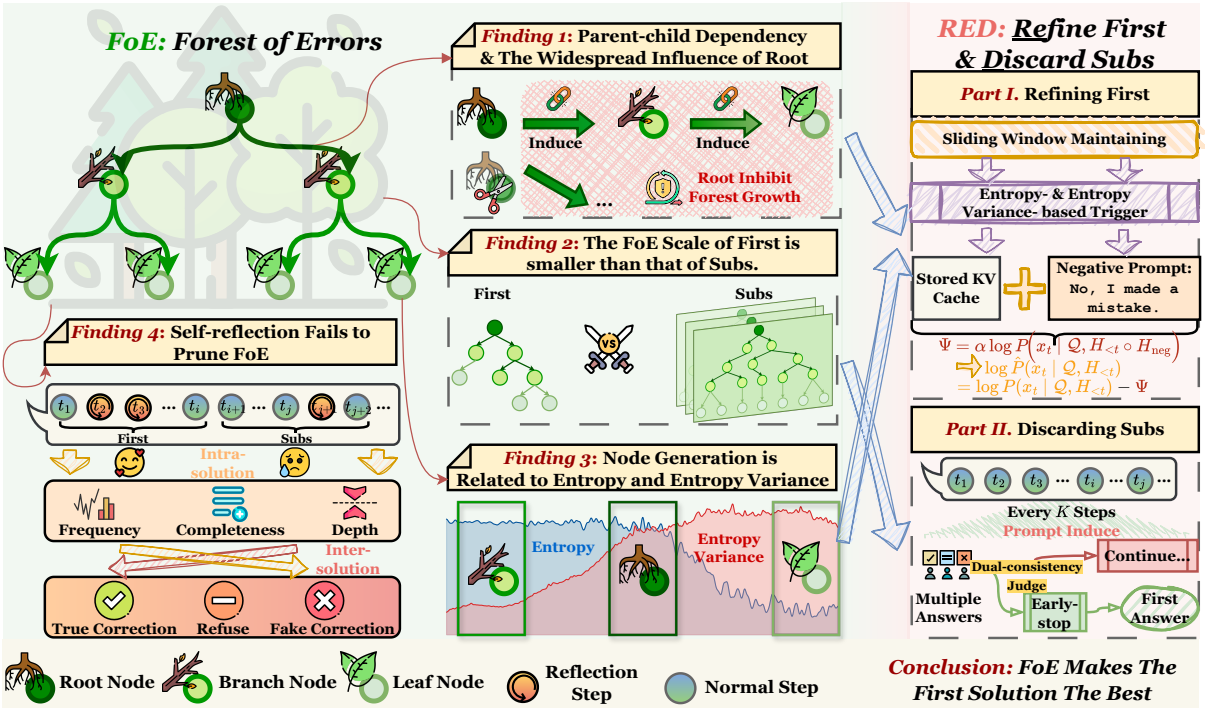


Figure 2: (***Left***) Key observations within the FoE. (***Right***) Our proposed RED framework.

Dataset	Model	✓ → ✗	✗ → ✗	✗ → ✓
AIME25	Qwen-distilled-7b	**18.3**	77.4	4.3
	Qwen-distilled-32b	**16.7**	76.9	6.4
	Qwen3-8b	**15.5**	79.1	5.4
	Qwen3-32b	**13.0**	79.9	7.1
	Llama-distilled-70b	**18.8**	75.4	5.8
GPQA-Diamond	Qwen-distilled-7b	**16.0**	81.1	2.9
	Qwen-distilled-32b	**15.3**	81.4	3.3
	Qwen3-8b	**14.4**	82.7	2.9
	Qwen3-32b	**14.1**	82.8	3.1
	Llama-distilled-70b	**21.2**	77.9	0.9

Table 1: Distribution of the Influence of Subs on First, excluding(✓ → ✓) (%). Notably, 93.7% of cases favor First, defined as instances where it is either uniquely correct or more robust than Subs (▶ Appendix E).

incorrect First, failing to provide effective cross-verification while resulting in a waste of computational resources.

Obs.❷ The Subs fail to rectify an incorrect first one, with a maximum success rate of merely 2.0%-7.1%; in many dataset-model combinations, successful cases are entirely absent.

Obs.❸ More alarmingly, Subs may mislead the model from a correct first one to an incorrect answer. This probability reaches up to 18.8%, significantly exceeding the rate of correcting an incorrect first one, implying that the potential risks of generating multiple solutions far outweigh their benefits.

Obs.❹ Furthermore, even in cases where both First and Subs appear correct, Subs retains a latent risk of inducing errors. To investigate this stability, we save the KV cache at regular intervals during generation. Upon completion, we rollback all solutions by a range of steps and perform extensive random sampling to regenerate the final answers. We observe that the probability of First yielding errors after sampling from the interruption point is merely 3.7% of that of Subs. Detailed experimental results are provided in Appendix E.

3 FoE

Through the case analysis (Figure 1), we observe that errors within the reasoning process manifest as a forest-like structure, which we term FoE. To investigate this further, we conduct an in-depth analysis of FoE to validate our hypothesis that as test-time scales up, errors within the reasoning path may scale up concurrently, thereby explaining the optimality of The First is The Best. To this end, we proceed as follows: **(i)** we formalize the modeling of FoE and analyze the dependencies between parent and child nodes (▶ Section 3.1); **(ii)** we introduce evaluation metrics specific to FoE and assess both First and Subs (▶ Section 3.2); **(iii)** we investigate the genesis of error nodes from an entropy perspective (▶ Section 3.3); and **(iv)** we examine whether the self-reflection capabilities of LRMs can prune FoE (▶ Section 3.4). Ultimately, these empirical analyses consistently support the conclusion that the FoE renders the First the Best.

3.1 FoE Initialization.

Forest Modeling. To facilitate a quantitative analysis of the FoE, we propose a formal modeling

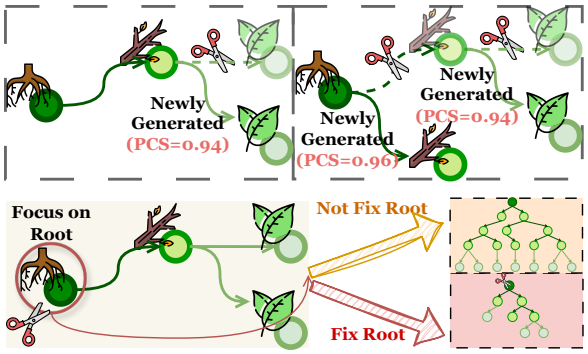


Figure 3: Manual correction on distinct error node types. (***Upper***) Impact of rectifying individual **G**randchild, **C**hild, and **R**oot nodes. (***Lower***) Consequences of delayed root correction in a formed tree, demonstrating substantial mitigation of subsequent node proliferation.

approach. Specifically, we assume that all errors within the reasoning trace have been identified using the o1-based annotation method (Yang et al., 2025c), which leverages powerful closed-source models for robust error detection. We first organize all errors into a chronological sequence, denoted as $[e_1, e_2, \ldots, e_n]$. For a given error e_j, to identify its parent node e_i, we define a parent-child association score, Score(e_i, e_j), which quantifies the likelihood of e_i inducing e_j. To implement this, we establish a $1 \sim 5$ scoring scale based on expert-designed criteria. Leveraging advanced LLMs via few-shot prompting, we evaluate candidate nodes e_k sequentially, ordered by their proximity to e_j (from nearest to farthest). Using a pre-defined threshold τ, if Score$(e_k, e_j) \geq \tau$, e_k is identified as the parent of e_j. If no candidate exceeds the threshold, e_j is designated as a root, instantiating a new Tree of Errors (ToE) within the FoE. The process then advances to the next unprocessed node and iterates. Further Modeling pipeline details are provided in Appendix F.2.

Obs.❶ Merely rectifying child nodes results in the parent continuously spawning new offspring, whereas correcting the root node significantly decelerates the error node generation rate. To investigate dependencies, we employ an iterative leaf-to-root correction strategy on grandchild (G), child (C), and root (R) nodes. As shown in Figure 3 (***Upper***), correcting only descendants (e.g., G2 or C1) fails to stop error propagation due to unaddressed ancestors. Conversely, correcting the root node even after it has spawned children effectively mitigates the rate of subsequent error generation (***Lower***). Further details are in Appendix F.3.

3.2 Evaluation Metrics of `FoE`

To quantitatively assess the scale and growth of `FoE`, we design the static and dynamic metrics.

Dataset	Static Metrics			Dynamic Metrics
	FS	N/T	D/T	Repro
AIME25	6.9/**8.1**↑1.2	7.1/**8.4**↑1.3	4.9/**5.7**↑0.8	0.084/**0.126**↑50.0%
MATH500	3.9/**5.6**↑1.7	5.6/**7.7**↑2.1	3.2/**4.4**↑1.2	0.047/**0.102**↑117.0%
GSM8K	1.1/**2.1**↑1.0	2.1/**3.7**↑1.6	1.5/**2.0**↑0.5	0.009/**0.027**↑200.0%
GPQA	4.2/**6.1**↑1.9	5.1/**7.6**↑2.5	3.4/**4.9**↑1.5	0.052/**0.107**↑105.8%

Table 2: Average statistics of error trees across datasets (First Solution / Average Subsequent Solutions).

Static Metrics. We employ following static metrics commonly used for forest structures: **(i) forest size(FS)**, the number of trees within `FoE`; **(ii) average nodes per tree(N/T)**, the average error count within a single tree; **(iii) average depth per tree(D/T)**, the number of layers from the root to the deepest descendant.

Dynamic Metrics. To characterize the dynamic evolutionary process of the forest in greater detail, we design an additional dynamic metric: **the Average Error Node Reproduction Rate (Repro)**. Let V denote the set of all error nodes within a given solution, where $|V|$ represents the total number of such nodes. For an arbitrary error node $v \in V$, let $k(v)$ denote the number of direct child error nodes generated by v, and let $L_t(v)$ represent the lifespan (i.e., the layer depth) of v. The average error node reproduction rate can be expressed as:

$$\bar{r} = \frac{1}{|V|} \sum_{v \in V} \frac{k(v)}{\max\big(L_t(v),\, 1\big)} \tag{1}$$

Evaluation. To evaluate the static and dynamic metrics of the `FoE`, we employ Qwen3-8B (thinking) on AIME25, MATH500, GSM8K, and GPQA datasets. The results are presented in Table 2.

Obs.❷ The `FoE` in `First` exhibits a significantly smaller scale and a slower average error node reproduction rate compared to that in `Subs`. In terms of *static* metrics, the forest size in `First` is substantially smaller than in `Subs` (6.9 vs. 8.1). Specifically, the forest depth in `Subs` exceeds that in `First` by approximately 16.3%. This implies that root errors in `Subs` propagate over longer durations and exert a broader impact. Additionally, we observe a higher count of average error nodes per tree within `Subs` (7.1 vs. 8.4), indicating that errors accumulate increasingly as the reasoning process progresses. Regarding *dynamic* metrics, we find that the average error node reproduction rate in `First` is 33.3% lower than in `Subs`. This suggests that error nodes in `First` generate offspring at a slower pace, further implying greater controllability over error propagation within `First`.

3.3 The Reason for Node Generation

Following the above analysis, a natural question arises: *How are error nodes within `FoE` generated?*

Insufficiency of individual metrics. Drawing on prior work linking entropy to uncertainty (Farquhar et al., 2024), we analyze node generation via entropy and its variance. However, high entropy may stem from benign alternatives rather than errors, while high variance can simply reflect valid reasoning transitions. Consequently, neither signal alone suffices; instead, it is their joint behavior that effectively characterizes the emergence of errors, particularly structural root nodes.

Empirical Substantiate. We conduct our experiments on a subset of Bespoke-Stratos-17k (Labs, 2025). Specifically, we select the mathematics and science tasks to construct a subset, denoted as BS-17k-subset. We locate each error node at its first occurrence $t(e)$ and extract the corresponding entropy features $(h_{t(e)}, v_{t(e)})$ using a window length of $L{=}15$, then partition the events into four percentile-based regions: low-low, high-h only, high-v only, and high-high. For each region, we compute the node-trigger rate (NTR), root-trigger rate (RTR), and average node depth (AND).

Obs.❸ Higher simultaneous levels of entropy and entropy variance correlate with a greater likelihood of root node generation; notably, the probability of root node emergence in `First` is lower than in `Subs`. Results in Figure 4 show that high-h alone mainly increases the frequency of errors but yields mostly shallow nodes, while high-v alone shifts errors to higher levels without maximizing root-node occurrence; in contrast, the high–high region exhibits the highest RTR, with `First` consistently achieving a lower RTR than `Subs` under the same condition. Moreover, this trend is robust to the window choice and remains stable when sweeping $L \in [10, 20]$. Further detailed analysis is provided in Appendix H.

3.4 LRMs trys to clear `FoE` through Reflection

One might argue that despite the continuous reproduction of error nodes, models trained via RL possess intrinsic self-reflection capabilities, enabling them to self-prune error nodes. But is this truly the case? Unfortunately, we find that the portion of the FoE cleared through self-reflection is extremely limited. To demonstrate this, we examine two distinct levels: *intra*- and *inter*-solution.

Intra-solution. At the intra-solution level, we examine the reflection behaviors inherent to each solution, focusing on three key axes: **(i) frequency**, denoting the number of reflective instances; **(ii) completeness**, which assesses whether the reflection is structurally complete, incomplete reflections often involve initiating a reflective thought only to prematurely resume reasoning; and **(iii) depth**, which gauges the profundity of error detection, shallow reflections typically identify superficial errors while failing to uncover underlying issues. The formal definitions and measurement methods for all axes are detailed in Appendix G.

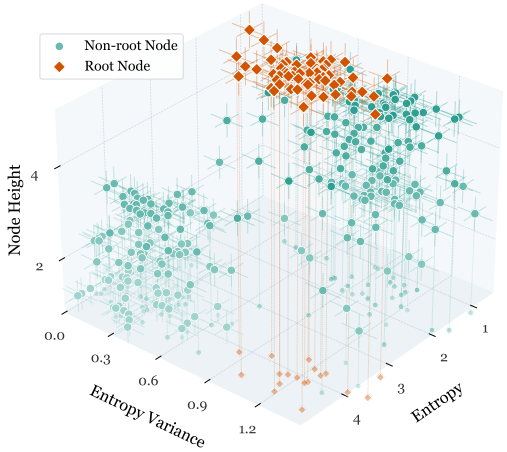


Figure 4: Distribution of various node types with respect to entropy and entropy variance. Experiments were conducted using the Qwen3-8B model on BS-17K-subset.

Obs.❹ `Subs` exhibits inferior reflection capabilities and a diminished capacity for error correction compared to `First`. Figure 6 presents large-scale experimental results across multiple backbones on BS-17k-subset. We observe a significant downward trend in the frequency, completeness, and depth of reflection within the `Subs` across models of nearly all sizes. For instance, with Qwen-8B-thinking, comparing the first to the last solution reveals a 62.5% reduction in reflection frequency, and a substantial decrease in both completeness (68.2%) and depth (82.1%). This indicates that `Subs` possesses weaker reflective and error-correction capabilities. Consequently, this suggests that if we aim to leverage self-reflection for error correction in LRMs, we should prioritize incentivizing self-reflection within `First`.

Inter-solution. At the inter-solution level, we investigate whether retained error nodes exert cross-solution influence when self-reflection fails. To demonstrate this, we facilitate model reflection by manually injecting prompts that signal errors within the context. This yields three distinct correction categories: **(i) True Correction**, where the model successfully identifies and rectifies the error; **(ii) Refusal to Correct**, involving either explicit denial or passive retention of the error; and **(iii) Fake Correction**, the most insidious type, charac-

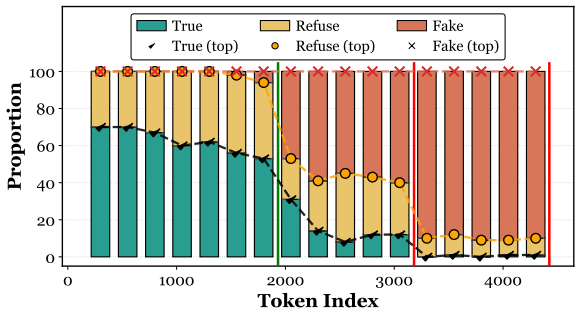


Figure 5: Average distribution of correction types (True, Refuse, Fake) on BS-17k-subset + Qwen3-8B. We manually inject error-signaling prompts to probe early-stage errors ($< 20\%$). The green and red vertical lines mark the completion of `First` and `Subs`, respectively.

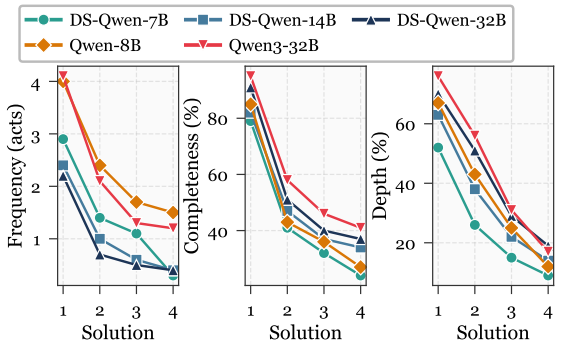


Figure 6: Evaluation of intra-solution reflection metrics.

terized by superficial textual edits while subsequent reasoning persists on the original erroneous logic.

Obs.❺ In `Subs`, fake correction and refusal to correct displace true correction as the dominant behaviors compared to `First`. As shown in Figure 5, we observe that in `First`, true correction dominates with a 67.1% share, while fake correction and refusal to correct constitute a minority (32.4% and 0.5%, respectively). Conversely, the scenario is nearly inverted for `Subs`, fake correction and refusal to correct surge to alarmingly high proportions of 64.2% and 31.1%. In the final solution, the behavior is essentially fake correction.

4 Method

In this section, we **(i)** outline the motivation behind our method design, which is grounded in the in-depth analysis of `FoE` and the extensive observations discussed previously (▶ Section 4.1); and **(ii)** elaborate on our proposed `RED`, a self-guided efficient reasoning method (▶ Section 4.2 & 4.3).

4.1 Motivation

Mot.❶ Refining `First`. Our analysis in Section 3.3 reveals that `First` still exhibits instances of simultaneous high entropy and high entropy variance, leading to the genesis of root error nodes, especially in complex tasks. Motivated by the insight from Section 3.1 that eliminating the root can suppress the growth of the entire `FoE`, we implement an intervention at segments characterized by high entropy and variance within `First`.

Mot.❷ Discarding `Subs`. Drawing from our analysis of `FoE` and the `The First is The Best` phenomenon, we realize that the potential risks introduced by `Subs` far outweigh their possible benefits. Furthermore, generating `Subs` results in a substantial waste of computational resources. These factors inspire us to discard `Subs` entirely.

4.2 Refining `First`.

Building on the finding in Section 3.3 that the co-occurrence of high entropy and high entropy variance precipitates error generation, we intervene at these critical junctures. Specifically, we maintain a sliding window of length L and monitor two key statistics: (1) the entropy variance of tokens within the window; and (2) the average of the maximum Top-K entropy values within the window. We trigger the intervention mechanism when the variance of the current window exceeds the threshold T and the entropy of the next token surpasses the recorded average. Drawing inspiration from classifier-free guidance (CFG)(Ho and Salimans, 2022), we append a concise negative prompt (e.g., "`No, I made a mistake.`") after the stored KV cache to generate a negative sampling branch. We then extract the logits of the next generated token, which represent the distribution steering the model toward an erroneous trajectory. Finally, we implement the intervention by subtracting these negative logits:

$$\begin{aligned}\log \hat{P}(x_t \mid \mathcal{Q}, H_{<t}) = \log P(x_t \mid \mathcal{Q}, H_{<t}) \\ -\alpha \log P\big(x_t \mid \mathcal{Q},\ H_{<t} \circ H_{\text{neg}}\big),\end{aligned} \tag{2}$$

where α is a tunable scaling coefficient, $\mathcal{Q}$ denotes the input question, and H represents the context. $P(\cdot)$ and $\hat{P}(\cdot)$ denote the pre- and post-intervention prediction probabilities, respectively.

4.3 Discarding `Subs`

Our method leverages two empirical observations under high-temperature sampling: (1) ***Convergence***: induced answers evolve from early-stage diversity to a single dominant mode as reasoning progresses; (2) ***Robustness***: unlike brittle intermediate states, stabilized induced answers remain consistent across different probe prompts. Further details are provided in Appendix I.

Dual-consistency Mechanism. Based on this, we trigger a periodic probe every K steps. We pause generation and concurrently inject M distinct prompts to induce interim answers (each

Model	Method	AIME24		AIME25		MATH500		GSM8K		GPQA-Diamond	
		Pass@1↑	Token↓	Pass@1↑	Token↓	Pass@1↑	Token↓	Pass@1↑	Token↓	Pass@1↑	Token↓
Qwen3-8B-thinking	Vanilla	70.0	11125	61.1	12490	95.9	4486	94.1	1573	58.8	6638
	DEER	71.1	7903	58.9	9271	95.6	2889	94.3	711	56.9	**2926**
	DAST	67.8	5964	60.0	**6055**	96.0	2158	94.1	483	58.4	3007
	Think or Not	68.9	5387	62.2	6674	96.3	2408	94.2	599	59.1	3678
	AlphaOne	73.3	8343	**63.3**	8607	96.7	4311	94.5	1095	59.8	6002
	RL + LP	71.1	6986	58.9	7639	96.6	2692	94.1	823	59.4	3139
	GRPO	72.2	10931	62.2	11098	96.7	4225	94.4	1373	59.1	7472
	S-GRPO	73.3	6771	61.1	7331	96.5	2576	94.6	775	59.6	3046
	RED	**75.6**	**5209**	**63.3**	6203	**97.0**	**2029**	**95.0**	**465**	**60.1**	3309
	Δ	↑5.6	↓53.1%	↑2.2	↓50.3%	↑1.1	↓54.8%	↑0.9	↓70.4%	↑1.3	↓50.2%
Qwen3-32B-thinking	Vanilla	77.8	10677	68.9	11589	96.8	4318	94.3	1435	65.3	5475
	DEER	74.4	7894	66.7	8417	96.4	2733	94.1	792	64.8	2247
	Think or Not	76.7	6173	65.6	6772	97.1	2179	94.5	641	64.5	**1713**
	AlphaOne	78.9	8007	71.1	8569	97.8	3170	94.4	1090	**66.8**	5591
	DAST	77.8	5981	68.9	6504	96.8	2417	94.2	609	65.7	2918
	RL + LP	76.7	6238	66.7	6854	97.2	2701	**94.7**	822	66.0	4362
	GRPO	78.9	11934	70.0	12487	97.1	4641	94.5	1558	66.2	6151
	S-GRPO	77.8	6040	71.1	6389	97.3	2566	94.6	809	66.5	4151
	RED	**80.0**	**5793**	**72.2**	**5908**	**97.9**	**1995**	94.6	**443**	**66.8**	2477
	Δ	↑2.2	↓45.7%	↑3.3	↓49.0%	↑1.1	↓53.8%	↑0.3	↓69.1%	↑1.5	↓54.8%
DpSk-R1-Distill-Qwen-7B	Vanilla	54.4	10438	43.3	11454	91.8	2887	92.4	442	49.2	8016
	DEER	53.3	7197	42.2	8261	92.4	1494	89.7	297	47.3	4423
	Think or Not	52.2	**4341**	38.9	**4760**	92.0	**1103**	92.9	**264**	47.0	3390
	AlphaOne	55.6	8224	44.4	8921	92.5	3791	93.4	459	50.5	8591
	DAST	55.6	7258	41.1	7904	91.6	1330	91.8	301	48.8	3635
	RL + LP	52.2	5693	43.3	6255	93.4	1322	92.5	291	50.3	3209
	GRPO	56.7	11673	45.6	12006	93.9	2873	92.1	275	49.8	8890
	S-GRPO	54.4	5094	44.4	5794	93.2	1204	93.7	297	49.5	**3107**
	RED	**57.8**	**4293**	**47.8**	5690	**94.1**	1187	**94.1**	271	**51.2**	4109
	Δ	↑3.4	↓58.9%	↑4.5	↓50.3%	↑2.3	↓58.9%	↑1.7	↓38.7%	↑2.0	↓48.7%
DpSk-R1-Distill-Qwen-32B	Vanilla	70.0	7873	58.9	8906	93.3	2337	93.9	438	60.8	6027
	DEER	68.9	6461	57.8	8008	91.8	1697	92.5	290	58.9	4611
	Think or Not	67.8	3993	58.9	**4711**	92.9	1410	93.3	247	60.4	3706
	AlphaOne	72.2	8210	61.1	8994	94.4	3037	94.1	433	61.3	6771
	DAST	70.0	5802	54.4	6647	93.5	1421	93.9	266	60.3	4048
	RL + LP	71.1	5492	62.2	6124	93.1	1379	**94.7**	239	60.4	3596
	GRPO	73.3	8389	62.2	8990	93.4	3012	94.4	420	60.4	7123
	S-GRPO	70.0	4906	60.0	5334	94.2	1556	**94.7**	269	61.3	**3119**
	RED	**74.4**	**3898**	**63.3**	5018	**95.2**	**1347**	94.6	**209**	**61.8**	3497
	Δ	↑4.4	↓50.5%	↑4.4	↓43.7%	↑1.9	↓42.4%	↑0.7	↓52.3%	↑1.0	↓42.0%
DeepSeek-R1-Distill-Llama-8B	Vanilla	45.6	10798.9	28.9	11548.2	86.2	3635	92.3	606	46.3	8341
	DEER	46.7	8001	31.1	8936	86.5	2171	89.6	394	45.5	4152
	Think or Not	44.4	6761	30.0	7158	87.4	**1954**	92.5	257	46.8	3729
	AlphaOne	**47.8**	8339	**34.4**	9005	89.1	3804	93.1	598	47.6	8569
	DAST	45.6	8246	32.2	8438	87.0	2458	91.9	388	46.1	4410
	RL + LP	45.6	5333	30.0	5897	89.4	2290	92.3	446	45.3	**3299**
	GRPO	44.4	11312	30.0	11987	89.6	3309	92.9	571	46.6	8783
	S-GRPO	45.6	4809	31.1	5426	89.1	2195	**93.2**	432	47.0	3624
	RED	**47.8**	**4507**	**34.4**	**5039**	**90.1**	2009	93.1	283	**47.8**	3593
	Δ	↑2.2	↓58.3%	↑5.5	↓56.4%	↑3.9	↓44.7%	↑0.8	↓53.3%	↑1.5	↓56.9%
DeepSeek-R1-Distill-Llama-70B	Vanilla	68.9	7766	47.8	8909	94.1	2433	94.0	432	64.5	5881
	DEER	70.0	6829	48.9	7478	92.3	1817	93.3	277	63.1	4502
	Think or Not	71.1	4005	46.7	4414	94.4	1360	93.7	241	64.6	3544
	AlphaOne	72.2	7873	50.0	8689	95.3	2009	94.2	427	65.3	4322
	DAST	67.8	5115	45.6	5932	94.2	1563	93.9	**239**	63.5	4026
	RL + LP	66.7	5304	45.6	5967	93.8	1149	94.4	255	65.0	4101
	GRPO	72.2	8109	**51.1**	8715	94.5	3167	94.7	473	64.8	6274
	S-GRPO	70.0	5002	48.9	5896	95.1	1252	94.5	249	65.2	4001
	RED	**73.3**	**3974**	50.0	**4303**	**96.2**	**932**	**94.9**	269	**66.2**	3563
	Δ	↑4.4	↓48.8%	↑2.2	↓51.7%	↑2.1	↓61.7%	↑0.9	↓37.7%	↑1.7	↓39.4%

Table 3: Main results. Best results are highlighted in **bold**, with runners-up underlined.

prompt with N parallel samples). To facilitate reliable extraction, we employ vLLM's guided decoding to constrain generation to a predefined set of valid answer tokens (e.g., option letters, numerical digits, or symbols). We trigger an early exit only if a *dual-consistency* condition is met: (i) **Internal**

Consistency, where the dominant answer within each prompt template must appear with a frequency $\geq P\%$; and (ii) **Cross-Prompt Agreement**, where the dominant answers across all M templates must be identical, ensuring the emerging answer is a robust solution rather than a prompt-sensitive artifact.

5 Experiments

5.1 Experimental Setup

Backbones. We conduct experiments using representative open-source LRMs with diverse architectures from different families. **I)** ***Qwen family***, Qwen3-thinking series (8B & 32B) (Yang et al., 2025a), DeepSeek-R1-Distill-Qwen series (7B & 32B); **II)** ***Llama family***, DeepSeek-R1-Distill-Llama series (8B & 70B).

Baselines. We compare `RED` against a comprehensive set of baselines categorized into three groups: **I) Vanilla Model**, the original backbone LRM; **II) Training-free methods**, including DEER (Yang et al., 2025b), Think or Not (Yong et al., 2025), and AlphaOne (Zhang et al., 2025b); **III) RL-based strategies**, including DAST (Shen et al., 2025), RL + Length Penalty (Arora and Zanette, 2025), GRPO (DeepSeek-AI et al., 2025), and S-GRPO (Dai et al., 2025). Detailed settings of baselines are provided in Appendix D.

Benchmarks. We conduct extensive evaluations of `RED` on five benchmarks spanning two complex reasoning domains: **I) Mathematical Reasoning**, including GSM8K (Cobbe et al., 2021), MATH500 (Lightman et al., 2023), AIME 2024, and AIME 2025; and **II) Scientific Reasoning**, specifically GPQA-Diamond (Rein et al., 2023).

Implementation details. We use a sampling temperature of 0.6 and top-p of 0.95, reporting Pass@1 averaged over three runs. For refining `First`, we set the sliding-window length L=15, threshold T=2.4, and top-K=3. For discarding `Subs`, we probe every 2 steps with M=4 distinct prompt templates, each with N=12 parallel samples and a consistency rate of 60%.

5.2 Main Results

Obs.❶ `RED` strikes the best balance between performance and efficiency across almost all scales. Compared to eight baselines on five benchmarks (Table 3), `RED` generally improves performance by $0.3 \sim 5.6$ ($3.2\% \sim 19.0\%$) over the Vanilla model while slashing token consumption by $37.7\% \sim 70.4\%$. Notably, on DeepSeek-R1-Distill-Llama-8B+AIME25, `RED` outperforms all baselines with a 5.5 score increase and 56.4% token reduction.

Model	Method	AIME25				
		Pass@1↑	FS↓	N/T↓	D/T↓	Repro↓
Qwen3-32B-thinking	Vanilla	68.9	6.8	7.4	5.3	0.081
	DEER	66.7	6.3	7.2	5.4	0.083
	RL + LP	66.7	5.9	6.9	4.7	0.071
	S-GRRO	71.1	6.1	7.0	5.1	0.074
	RED	**72.2**	**3.1**	**4.2**	**3.1**	**0.026**
	Δ	↑+3.3	↓54.4%	↓43.2%	↓41.5%	↓67.9%
R1-Qwen-32B	Vanilla	58.9	7.0	7.8	5.8	0.095
	DEER	57.8	6.5	7.6	5.9	0.097
	RL + LP	62.2	6.1	7.3	5.2	0.085
	S-GRRO	60.0	6.3	7.4	5.6	0.088
	RED	**63.3**	**3.2**	**4.6**	**3.6**	**0.040**
	Δ	↑+4.4	↓54.3%	↓41.0%	↓37.9%	↓57.9%
R1-Llama-70B	Vanilla	47.8	8.6	8.3	6.4	0.125
	DEER	48.9	7.9	7.8	6.5	0.128
	RL + LP	45.6	7.6	7.5	5.8	0.110
	S-GRRO	48.9	7.8	7.6	6.1	0.115
	RED	**50.0**	**3.9**	**4.7**	**3.7**	**0.040**
	Δ	↑+2.2	↓54.7%	↓43.4%	↓42.2%	↓68.0%

Table 4: Results on AIME25 with FoE-related metrics. Additional results are provided in Appendix B.1.

It even surpasses the strong S-GRPO baseline by 3.3 in score and 7.1% in efficiency. Furthermore, scalability tests on the DeepSeek-R1-Distill-Llama series reveal that while DAST's performance gain degrades from $+3.3$ (8B) to -2.2 (70B) relative to Vanilla, `RED` remains robust, achieving gains of $+5.5$ and $+2.2$, respectively.

Obs.❷ `RED` achieves substantial pruning of the `FoE`. Table 4 shows that our approach reduces all `FoE` metrics by $41.0\% \sim 68.0\%$, setting a new standard against baselines. Scrutinizing the results reveals a sharp contrast: even the competitive S-GRPO struggles with `FoE` mitigation. On DeepSeek-R1-Distill-Qwen-32B, S-GRPO lowers the static D/T metric by a negligible 3.4% and the dynamic Repro metric by 7.4%. In comparison, `RED` dramatically reduces by 37.9% and 57.9%, respectively. By simultaneously shrinking forest size and inhibiting node reproduction, `RED` validates the effectiveness of our proposed mechanism.

5.3 Framework Analysis

Ablation studies, sensitivity analysis, and Cons@k are detailed in Appendices B.3, C, and B.2.

6 Conclusion

In this work, we uncover the counter-intuitive `The First is The Best` in LRMs, significantly challenging the test-time scaling laws. Through a rigorous analysis, we attribute it to the Forest of Errors (`FoE`), revealing that reasoning errors scale concurrently with test time, making the first the best. Motivated by these, we introduce `RED`, a framework that synergizes *Refining First* to inhibit error growth and *Discarding Subs* to eliminate redundant, error-prone computations. We believe our findings offer valuable insights for further research.

Limitations

A potential limitation of RED is the introduction of additional latency due to extra operations within the decoding process. However, we evaluated this additional latency through a rigorous latency stress test (► Appendix J). Specifically, we isolated the operational overhead by disabling the early-exit mechanism (i.e., continuing generation even if exit criteria are satisfied) while fully maintaining the periodic injection of probe prompts and conducting inference intervention based on entropy and entropy variance. This worst-case profiling reveals that the additional latency overhead averages only 4.6%. Since this additional latency is minimal, the time saved by the early-exit mechanism significantly outweighs it. As a result, RED achieves a net speedup compared to the baseline.

Acknowledgement

This work is supported by the State Key Laboratory of General Artificial Intelligence; and the National Natural Science Foundation of China (Grant No. 62276006); and the Humanities and Social Sciences Research Planning Fund Project of the Ministry of Education: "Research on Metacognitive Diagnosis Theory and Technology Driven by Multimodal Learning Data" (23YJA880091).

References

Pranjal Aggarwal and Sean Welleck. 2025. L1: controlling how long A reasoning model thinks with reinforcement learning. *CoRR*, abs/2503.04697.

Shengnan An, Xunliang Cai, Xuezhi Cao, Xiaoyu Li, Yehao Lin, Junlin Liu, Xinxuan Lv, Dan Ma, Xuanlin Wang, Ziwen Wang, and Shuang Zhou. 2025. Amobench: Large language models still struggle in high school math competitions. *CoRR*, abs/2510.26768.

Daman Arora and Andrea Zanette. 2025. Training language models to reason efficiently. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Simon A. Aytes, Jinheon Baek, and Sung Ju Hwang. 2025. Sketch-of-thought: Efficient LLM reasoning with adaptive cognitive-inspired sketching. *CoRR*, abs/2503.05179.

Yupeng Chang, Yi Chang, and Yuan Wu. 2026. BA-loRA: Bias-alleviating low-rank adaptation to mitigate catastrophic inheritance in large language models. In *The Fourteenth International Conference on Learning Representations*.

Yupeng Chang, Chenlu Guo, Yi Chang, and Yuan Wu. 2025. Lora-mgpo: Mitigating double descent in low-rank adaptation via momentum-guided perturbation optimization. In *Findings of the Association for Computational Linguistics: EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 648–659. Association for Computational Linguistics.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. 2024. A survey on evaluation of large language models. *ACM Trans. Intell. Syst. Technol.*, 15(3):39:1–39:45.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. Evaluating large language models trained on code. *CoRR*, abs/2107.03374.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2024. Do NOT think that much for 2+3=? on the overthinking of o1-like llms. *CoRR*, abs/2412.21187.

Zhiwei Chen, Yupeng Hu, Zhiheng Fu, Zixu Li, Jiale Huang, Qinlei Huang, and Yinwei Wei. 2026. INTENT: invariance and discrimination-aware noise mitigation for robust composed image retrieval. In *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pages 20463–20471. AAAI Press.

Zhiwei Chen, Yupeng Hu, Zixu Li, Zhiheng Fu, Xuemeng Song, and Liqiang Nie. 2025. OFFSET: segmentation-based focus shift revision for composed image retrieval. In *Proceedings of the 33rd ACM International Conference on Multimedia, MM 2025, Dublin, Ireland, October 27-31, 2025*, pages 6113–6122. ACM.

Jeffrey Cheng and Benjamin Van Durme. 2024. Compressed chain of thought: Efficient reasoning through dense representations. *CoRR*, abs/2412.13171.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Mz Dai, Chenxu Yang, and Qingyi Si. 2025. S-GRPO: Early exit via reinforcement learning in reasoning models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 181 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Haonan Dong, Wenhao Zhu, Guojie Song, and Liang Wang. 2025. AuroRA: Breaking low-rank bottleneck of loRA with nonlinear mapping. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. 2024. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630.

Yichao Fu, Junda Chen, Yonghao Zhuang, Zheyu Fu, Ion Stoica, and Hao Zhang. 2025. Reasoning without self-doubt: More efficient chain-of-thought through certainty probing. In *ICLR 2025 Workshop on Foundation Models in the Wild*.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. 2025. Token-budget-aware LLM reasoning. In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 24842–24855. Association for Computational Linguistics.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2024. Training large language models to reason in a continuous latent space. *CoRR*, abs/2412.06769.

Yanji He, Yuxin Jiang, Yiwen Wu, Bo Huang, Jiaheng Wei, and Wei Wang. 2026. Idea: An interpretable and editable decision-making framework for llms via verbal-to-numeric calibration. *Preprint*, arXiv:2604.12573.

Jonathan Ho and Tim Salimans. 2022. Classifier-free diffusion guidance. *Preprint*, arXiv:2207.12598.

Yiyang Jiang, Guangwu Qian, Jiaxin Wu, Qi Huang, Qing Li, Yongkang Wu, and Xiao-Yong Wei. 2026. Self-paced learning for images of antinuclear antibodies. *IEEE Trans. Medical Imaging*, 45(4):1661–1672.

Yiyang Jiang, Wengyu Zhang, Xulu Zhang, Xiaoyong Wei, Chang Wen Chen, and Qing Li. 2024. Prior knowledge integration via LLM encoding and pseudo event regulation for video moment retrieval. In *Proceedings of the 32nd ACM International Conference on Multimedia, MM 2024, Melbourne, VIC, Australia, 28 October 2024 - 1 November 2024*, pages 7249–7258. ACM.

Philip N Johnson-Laird. 2010. Mental models and human reasoning. *Proceedings of the National Academy of Sciences*, 107(43):18243–18250.

Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. 2025. C3ot: Generating shorter chain-of-thought without compromising effectiveness. In *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 24312–24320. AAAI Press.

Bespoke Labs. 2025. Bespoke-stratos: The unreasonable effectiveness of reasoning distillation. https://www.bespokelabs.ai/blog/bespoke-stratos-the-unreasonable-effectiveness-of-reasoning-distillation. Accessed: 2025-01-22.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay V. Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. Solving quantitative reasoning problems with language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Junxian Li, Di Zhang, Xunzhi Wang, Zeying Hao, Jingdi Lei, Qian Tan, Cai Zhou, Wei Liu, Yaotian Yang, Xinrui Xiong, Weiyun Wang, Zhe Chen, Wenhai Wang, Wei Li, Mao Su, Shufei Zhang, Wanli Ouyang, Yuqiang Li, and Dongzhan Zhou. 2025a. Chemvlm: Exploring the power of multimodal large language models in chemistry area. In *Thirty-Ninth AAAI Conference on Artificial Intelligence, Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence, Fifteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2025, Philadelphia, PA, USA, February 25 - March 4, 2025*, pages 415–423. AAAI Press.

Mengdi Li, Jiaye Lin, Xufeng Zhao, Wenhao Lu, Peilin Zhao, Stefan Wermter, and Di Wang. 2025b. Curriculum-rlaif: Curriculum alignment with reinforcement learning from AI feedback. *CoRR*, abs/2505.20075.

Songze Li, Xiaoke Guo, Tianqi Liu, Biao Yi, Zhaoyan Gong, Zhiqiang Liu, Huajun Chen, and Wen Zhang. 2026a. What's missing in screen-to-action? towards a ui-in-the-loop paradigm for multimodal gui reasoning. *Preprint*, arXiv:2604.06995.

Zixu Li, Yupeng Hu, Zhiwei Chen, Qinlei Huang, Guozhi Qiu, Zhiheng Fu, and Meng Liu. 2026b. Retrack: Evidence-driven dual-stream directional anchor calibration network for composed video retrieval. In *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pages 23373–23381. AAAI Press.

Zixu Li, Yupeng Hu, Zhiwei Chen, Shiqi Zhang, Qinlei Huang, Zhiheng Fu, and Yinwei Wei. 2026c. HABIT: chrono-synergia robust progressive learning framework for composed image retrieval. In *Fortieth AAAI*

Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026, pages 6762–6770. AAAI Press.

Guosheng Liang, Longguang Zhong, Ziyi Yang, and Xiaojun Quan. 2025. Thinkswitcher: When to think hard, when to think fast. *CoRR*, abs/2505.14183.

Baohao Liao, Yuhui Xu, Hanze Dong, Junnan Li, Christof Monz, Silvio Savarese, Doyen Sahoo, and Caiming Xiong. 2025. Reward-guided speculative decoding for efficient LLM reasoning. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*. OpenReview.net.

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. *Preprint*, arXiv:2305.20050.

Jiaye Lin, Yifu Guo, Yuzhen Han, Sen Hu, Ziyi Ni, Licheng Wang, Mingguang Chen, Hongzhang Liu, Ronghao Chen, Yangfan He, Daxin Jiang, Binxing Jiao, Chen Hu, and Huacan Wang. 2025. Se-agent: Self-evolution trajectory optimization in multi-step reasoning with llm-based agents. *CoRR*, abs/2508.02085.

Junlin Liu, Shengnan An, Shuang Zhou, Dan Ma, Shixiong Luo, Ying Xie, Yuan Zhang, Wenling Yuan, Yifan Zhou, Xiaoyu Li, Ziwen Wang, Xuezhi Cao, and Xunliang Cai. 2026. General365: Benchmarking general reasoning in large language models across diverse and challenging tasks. *Preprint*, arXiv:2604.11778.

Peiyang Liu, Sen Wang, Xi Wang, Wei Ye, and Shikun Zhang. 2021. Quadrupletbert: An efficient model for embedding-based large-scale retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 3734–3739. Association for Computational Linguistics.

Peiyang Liu, Xi Wang, Ziqiang Cui, and Wei Ye. 2025a. Queries are not alone: Clustering text embeddings for video search. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2025, Padua, Italy, July 13-18, 2025*, pages 874–883. ACM.

Peiyang Liu, Jinyu Yang, Lin Wang, Sen Wang, Yunlai Hao, and Huihui Bai. 2023. Retrieval-based unsupervised noisy label detection on text data. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM 2023, Birmingham, United Kingdom, October 21-25, 2023*, pages 4099–4104. ACM.

Tengxiao Liu, Qipeng Guo, Xiangkun Hu, Cheng Jiayang, Yue Zhang, Xipeng Qiu, and Zheng Zhang. 2024a. Can language models learn to skip steps? In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Zheng Liu, Hao Liang, Xijie Huang, Wentao Xiong, Qinhan Yu, Linzhuang Sun, Chong Chen, Conghui He, Bin Cui, and Wentao Zhang. 2024b. Synthvlm: High-efficiency and high-quality synthetic data for vision language models. *CoRR*, abs/2407.20756.

Zheng Liu, Mengjie Liu, Siwei Wen, Mengzhang Cai, Bin Cui, Conghui He, and Wentao Zhang. 2025b. From uniform to heterogeneous: Tailoring policy optimization to every token's nature. *CoRR*, abs/2509.16591.

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

Ruotian Ma, Peisong Wang, Cheng Liu, Xingyan Liu, Jiaqi Chen, Bang Zhang, Xin Zhou, Nan Du, and Jia Li. 2025a. S^2r: Teaching llms to self-verify and self-correct via reinforcement learning. *CoRR*, abs/2502.12853.

Xinyin Ma, Guangnian Wan, Runpeng Yu, Gongfan Fang, and Xinchao Wang. 2025b. Cot-valve: Length-compressible chain-of-thought tuning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 6025–6035. Association for Computational Linguistics.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. Self-refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*.

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

Tergel Munkhbat, Namgyu Ho, Seo Hyun Kim, Yongjin Yang, Yujin Kim, and Se-Young Yun. 2025. Self-training elicits concise reasoning in large language models. In *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 25127–25152. Association for Computational Linguistics.

Shuaiyi Nie, Siyu Ding, Wenyuan Zhang, Linhao Yu, Tianmeng Yang, Yao Chen, Tingwen Liu, Weichong

Yin, Yu Sun, and Hua Wu. 2026. ATTNPO: attention-guided process supervision for efficient reasoning. *CoRR*, abs/2602.09953.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M. Waleed Kadous, and Ion Stoica. 2024. Routellm: Learning to route llms with preference data. *CoRR*, abs/2406.18665.

OpenAI, :, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, and 244 others. 2024. Openai o1 system card. *Preprint*, arXiv:2412.16720.

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. *Preprint*, arXiv:2311.12022.

Yi Shen, Jian Zhang, Jieyun Huang, Shuming Shi, Wenjing Zhang, Jiangze Yan, Ning Wang, Kai Wang, Zhaoxiang Liu, and Shiguo Lian. 2025. DAST: Difficulty-adaptive slow-thinking for large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 2322–2331, Suzhou (China). Association for Computational Linguistics.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *CoRR*, abs/2408.03314.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, Hanjie Chen, and Xia Hu. 2025. Stop overthinking: A survey on efficient reasoning for large language models. *Preprint*, arXiv:2503.16419.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. 2023. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 13003–13051. Association for Computational Linguistics.

Gemini Team. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *CoRR*, abs/2507.06261.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Fengxiang Tang, Flood Sung, Guangda Wei, Guokun Lai, and 75 others. 2025. Kimi k1.5: Scaling reinforcement learning with llms. *CoRR*, abs/2501.12599.

Jiayu Wang, Yifei Ming, Zixuan Ke, Caiming Xiong, Shafiq Joty, Aws Albarghouthi, and Frederic Sala. 2025a. Beyond accuracy: Dissecting mathematical reasoning for llms under reinforcement learning. *CoRR*, abs/2506.04723.

Tongxi Wang. 2026. Fbs: Modeling native parallel reading inside a transformer. *Preprint*, arXiv:2601.21708.

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*.

Yiming Wang, Pei Zhang, Siyuan Huang, Baosong Yang, Zhuosheng Zhang, Fei Huang, and Rui Wang. 2025b. Sampling-efficient test-time scaling: Self-estimating the best-of-n sampling in early decoding. *CoRR*, abs/2503.01422.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Zhenhe Wu, Jian Yang, Jiaheng Liu, Xianjie Wu, Changzai Pan, Jie Zhang, Yu Zhao, Shuangyong Song, Yongxiang Li, and Zhoujun Li. 2025. Table-r1: Region-based reinforcement learning for table understanding. *CoRR*, abs/2505.12415.

Heming Xia, Yongqi Li, Chak Tou Leong, Wenjie Wang, and Wenjie Li. 2025. Tokenskip: Controllable chain-of-thought compression in llms. *CoRR*, abs/2502.12067.

Xi Xiao, Chenrui Ma, Yunbei Zhang, Chen Liu, Zhuxuanzi Wang, Yanshu Li, Lin Zhao, Guosheng Hu, Tianyang Wang, and Hao Xu. 2026a. Not all directions matter: Toward structured and task-aware low-rank adaptation. *CoRR*, abs/2603.14228.

Xi Xiao, Yunbei Zhang, Xingjian Li, Tianyang Wang, Xiao Wang, Yuxiang Wei, Jihun Hamm, and Min Xu. 2025. Visual instance-aware prompt tuning. In *Proceedings of the 33rd ACM International Conference on Multimedia, MM 2025, Dublin, Ireland, October 27-31, 2025*, pages 2880–2889. ACM.

Xi Xiao, Yunbei Zhang, Lin Zhao, Yiyang Liu, Xiaoying Liao, Zheda Mai, Xingjian Li, Xiao Wang, Hao

Xu, Jihun Hamm, Xue Lin, Min Xu, Qifan Wang, Tianyang Wang, and Cheng Han. 2026b. Prompt-based adaptation in large-scale vision models: A survey. *Trans. Mach. Learn. Res.*, 2026.

Can Xie, Ruotong Pan, Xiangyu Wu, Yunfei Zhang, Jiayi Fu, Tingting Gao, and Guorui Zhou. 2025. Unlocking exploration in RLVR: uncertainty-aware advantage shaping for deeper reasoning. *CoRR*, abs/2510.10649.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. 2025a. Chain of draft: Thinking faster by writing less. *CoRR*, abs/2502.18600.

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. 2025b. Softcot: Soft chain-of-thought for efficient reasoning with llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 23336–23351. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

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

Zhaohui Yang, Chenghua He, Xiaowen Shi, Shihong Deng, Linjing Li, Qiyue Yin, and Daxin Jiang. 2025c. Beyond the first error: Process reward models for reflective mathematical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 4711–4728, Suzhou, China. Association for Computational Linguistics.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. 2025. LIMO: less is more for reasoning. *CoRR*, abs/2502.03387.

Xixian Yong, Xiao Zhou, Yingying Zhang, Jinlin Li, Yefeng Zheng, and Xian Wu. 2025. Think or not? exploring thinking efficiency in large reasoning models via an information-theoretic lens. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, YuYue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, and 17 others. 2025. DAPO: An open-source LLM reinforcement learning system at scale. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Haohan Yuan, Sukhwa Hong, and Haopeng Zhang. 2026. Strucsum: Graph-structured reasoning for long document extractive summarization with llms. In *Findings of the Association for Computational Linguistics: EACL 2026, Rabat, Morocco, March 24-29, 2026*, Findings of ACL, pages 3708–3721. Association for Computational Linguistics.

Haohan Yuan and Haopeng Zhang. 2025a. Domainsum: A hierarchical benchmark for fine-grained domain shift in abstractive text summarization. In *Findings of the Association for Computational Linguistics: NAACL 2025, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, Findings of ACL, pages 2219–2231. Association for Computational Linguistics.

Haohan Yuan and Haopeng Zhang. 2025b. Understanding LLM reasoning for abstractive summarization. *CoRR*, abs/2512.03503.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. 2025. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Jintian Zhang, Yuqi Zhu, Mengshu Sun, Yujie Luo, Shuofei Qiao, Lun Du, Da Zheng, Huajun Chen, and Ningyu Zhang. 2025a. Lightthinker: Thinking step-by-step compression. *CoRR*, abs/2502.15589.

Junyu Zhang, Runpei Dong, Han Wang, Xuying Ning, Haoran Geng, Peihao Li, Xialin He, Yutong Bai, Jitendra Malik, Saurabh Gupta, and Huan Zhang. 2025b. AlphaOne: Reasoning models thinking slow and fast at test time. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 11340–11365, Suzhou, China. Association for Computational Linguistics.

Xiaoyun Zhang, Jingqing Ruan, Xing Ma, Yawen Zhu, Haodong Zhao, Hao Li, Jiansong Chen, Ke Zeng, and Xunliang Cai. 2025c. When to continue thinking: Adaptive thinking mode switching for efficient reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 5808–5828. Association for Computational Linguistics.

Ziqi Zhao, Zhaochun Ren, Jiahong Zou, Liu Yang, Zhiwei Xu, Xuri Ge, Zhumin Chen, Xinyu Ma, Daiting Shi, Shuaiqiang Wang, Dawei Yin, and Xin Xin. 2026. Reinforced efficient reasoning via semantically diverse exploration. *CoRR*, abs/2601.05053.

Keyang Zhong, Junlin Xie, Hefeng Wu, Haofeng Li, and Guanbin Li. 2026. Collaborative multi-agent scripts generation for enhancing imperfect-information reasoning in murder mystery games. *Preprint*, arXiv:2604.11741.

Yixiao Zhou, Ziyu Zhao, Dongzhou Cheng, Zhiliang Wu, Jie Gui, Yi Yang, Fei Wu, Yu Cheng, and Hehe Fan. 2025. Dropping experts, recombining neurons: Retraining-free pruning for sparse mixture-of-experts llms. In *Findings of the Association for Computational Linguistics: EMNLP 2025, Suzhou, China,*

November 4-9, 2025, pages 15169–15186. Association for Computational Linguistics.

A Related Work

Efficient LLM Reasoning. According to the efficient-reasoning survey of Sui et al. (2025), existing approaches can be organized into four families: (1) Model-based Efficient Reasoning, (2) Reasoning Output-based Efficient Reasoning, (3) Input Prompts-based Efficient Reasoning, and (4) Efficient Data and Models. ❶ **Model-based** methods internalize conciseness by training: a major thread augments RL with explicit length-aware objectives to curb overlong chains while preserving accuracy (e.g., O1-Pruner (Luo et al., 2025), Kimi k1.5 (Team et al., 2025), L1 (Aggarwal and Welleck, 2025), DAST (Shen et al., 2025), while another line performs SFT (Dong et al., 2025) on variable-length CoT data, either compressing long traces post hoc (e.g., C3oT (Kang et al., 2025), TokenSkip (Xia et al., 2025), ASRR (Zhang et al., 2025c)) or eliciting shorter reasoning during generation (e.g., Learn-to-Skip (Liu et al., 2024a), Self-Training (Munkhbat et al., 2025), CoT-Valve (Ma et al., 2025b)). ❷ **Reasoning Output-based** methods reduce explicit tokens by altering the reasoning form at inference: latent reasoning compresses intermediate steps into hidden/continuous representations (e.g., Coconut (Hao et al., 2024), CCOT (Cheng and Durme, 2024), SoftCoT (Xu et al., 2025b)), and dynamic inference allocates computation adaptively via reward-, confidence-, or consistency-guided control (e.g., RSD (Liao et al., 2025), Dynasor (Fu et al., 2025), ST-BoN (Wang et al., 2025b), LightThinker (Zhang et al., 2025a)). ❸ **Input Prompts-based** approaches improve efficiency without changing core weights by (i) directly prompting for brevity or budget adherence (e.g., TokenBudget (Han et al., 2025), Chain-of-Draft (Xu et al., 2025a)) and (ii) routing queries to different reasoning modes/models based on prompt attributes such as difficulty (e.g., Sketch-of-Thought (Aytes et al., 2025), RouteLLM (Ong et al., 2024), ThinkSwitcher (Liang et al., 2025)). ❹ Finally, **Efficient Data and Models** target deployment-oriented efficiency through data-efficient reasoning supervision (e.g., LIMO (Ye et al., 2025), s1 (Muennighoff et al., 2025), S2R (Ma et al., 2025a)). Moreover, several studies have emerged aiming to improve LRMs reasoning via explainability-based enhancements (He et al., 2026; Zhong et al., 2026; Xie et al., 2025; Li et al., 2026a; Zhou et al., 2025; Nie et al., 2026; Wang, 2026). In contrast to these works, we first identify two critical phenomena: `The First is The Best` and `FoE`. Following a series of in-depth analyses, we introduce `RED`, a method grounded in these insights and underpinned by extensive empirical observations.

B Additional Experimental Results

B.1 Additional Results of FoE-related Metrics

Mathematical Task. On MATH500 (Table 6), `RED` continues to substantially prune the `FoE` across all three backbones, while maintaining (and slightly improving) accuracy: Pass@1 increases by $+1.1 \sim +2.1$ compared to Vanilla, and all `FoE` metrics (FS, N/T, D/T, Repro) are reduced by $37.1\% \sim 68.0\%$. Importantly, the advantage is not merely from "shorter" traces, but from directly suppressing both the *static* forest structure and the *dynamic* reproduction process. This is evident when contrasting against the strong RL baseline S-GRPO: on DeepSeek-R1-Distill-Qwen-32B, S-GRPO only yields a negligible D/T reduction of 2.9% and a Repro reduction of 7.0%, whereas `RED` reduces D/T and Repro by 37.1% and 57.9%, respectively. The same pattern holds for Qwen3-32B-Thinking and R1-Distill-Llama-70B, indicating that competitive baselines may reach similar Pass@1 but still struggle to mitigate the underlying `FoE` growth.

Scientific Task. On GPQA-Diamond (Table 7), the same phenomenon persists under a clear domain shift from math to scientific reasoning: `RED` consistently improves Pass@1 by $+1.0 \sim +1.7$ and simultaneously reduces all `FoE` metrics by $38.6\% \sim 68.1\%$ relative to Vanilla. Again, S-GRPO exhibits limited `FoE` mitigation—on DeepSeek-R1-Distill-Qwen-32B, it reduces D/T by only 4.5% and Repro by 7.0%, whereas `RED` achieves much larger reductions of 38.6% and 57.7%, respectively. Together with the AIME25 evidence (Table 5), these consistent gains across datasets strengthen the core conclusion in the main text: naively extending test-time exploration does not reliably "fix" earlier mistakes, but instead tends to expand a `FoE` through both wider/deeper error structures and faster error reproduction. By simultaneously shrinking the forest (FS, N/T, D/T) and inhibiting node reproduction (Repro), `RED` offers a direct mechanism-level explanation for why the First solution often remains the best under test-time scaling.

Model	Method	AIME25				
		Pass@1↑	FS↓	N/T↓	D/T↓	Repro↓
Qwen3-32B-Thinking	Vanilla	68.9	6.8	7.4	5.3	0.081
	DEER	66.7	6.3	7.2	5.4	0.083
	RL + LP	66.7	5.9	6.9	4.7	0.071
	S-GRRO	71.1	6.1	7.0	5.1	0.074
	RED	**72.2**	**3.1**	**4.2**	**3.1**	**0.026**
	Δ	↑+3.3	↓54.4%	↓43.2%	↓41.5%	↓67.9%
R1-Qwen-32B	Vanilla	58.9	7.0	7.8	5.8	0.095
	DEER	57.8	6.5	7.6	5.9	0.097
	RL + LP	62.2	6.1	7.3	5.2	0.085
	S-GRRO	60.0	6.3	7.4	5.6	0.088
	RED	**63.3**	**3.2**	**4.6**	**3.6**	**0.040**
	Δ	↑+4.4	↓54.3%	↓41.0%	↓37.9%	↓57.9%
R1-Llama-70B	Vanilla	47.8	8.6	8.3	6.4	0.125
	DEER	48.9	7.9	7.8	6.5	0.128
	RL + LP	45.6	7.6	7.5	5.8	0.110
	S-GRRO	48.9	7.8	7.6	6.1	0.115
	RED	**50.0**	**3.9**	**4.7**	**3.7**	**0.040**
	Δ	↑+2.2	↓54.7%	↓43.4%	↓42.2%	↓68.0%

Table 5: Results on AIME25 with FoE-related metrics.

Model	Method	MATH500				
		Pass@1	FS↓	N/T↓	D/T↓	Repro↓
Qwen3-32B-Thinking	Vanilla	96.8	4.1	4.4	3.2	0.049
	DEER	96.4	3.8	4.3	3.2	0.050
	RL + LP	97.2	3.5	4.1	2.8	0.043
	S-GRRO	97.3	3.7	4.2	3.1	0.044
	RED	**97.9**	**1.9**	**2.5**	**1.9**	**0.016**
	Δ	↑+1.1	↓54.4%	↓43.2%	↓41.5%	↓67.9%
R1-Qwen-32B	Vanilla	93.3	4.2	4.7	3.5	0.057
	DEER	91.8	3.9	4.6	3.5	0.058
	RL + LP	93.1	3.7	4.4	3.1	0.051
	S-GRRO	94.2	3.8	4.4	3.4	0.053
	RED	**95.2**	**1.9**	**2.8**	**2.2**	**0.024**
	Δ	↑+1.9	↓54.3%	↓41.0%	↓37.9%	↓57.9%
R1-Llama-70B	Vanilla	94.1	5.2	5.0	3.8	0.075
	DEER	92.3	4.7	4.7	3.9	0.077
	RL + LP	93.8	4.6	4.5	3.5	0.066
	S-GRRO	95.1	4.7	4.6	3.7	0.069
	RED	**96.2**	**2.3**	**2.8**	**2.2**	**0.024**
	Δ	↑+2.1	↓54.7%	↓43.4%	↓42.2%	↓68.0%

Table 6: Results on MATH500 with FoE-related metrics.

B.2 Self-Consistency Experiment

Figure 7a and 7b illustrate that **RED** achieves superior Cons@k performance across nearly all k values compared to Vanilla SC, AlphaOne, and S-GRPO. On the challenging AIME25 benchmark with the DeepSeek-R1-Distill-Llama-70B backbone, **RED** demonstrates substantial improvements, outperforming the vanilla baseline by a remarkable margin of **10.0%** at $k = 64$ ($56.7\% \rightarrow 66.7\%$). Notably, **RED** exhibits exceptional sample efficiency. As shown in Figure 7b, on the Llama-70B model, our method at $k = 8$ achieves performance (56.7%) comparable to the vanilla method at $k = 64$, indicating an $\sim 8\times$ reduction in computational cost to reach the same accuracy. Similarly, on the DeepSeek-R1-Distill-Qwen-32B

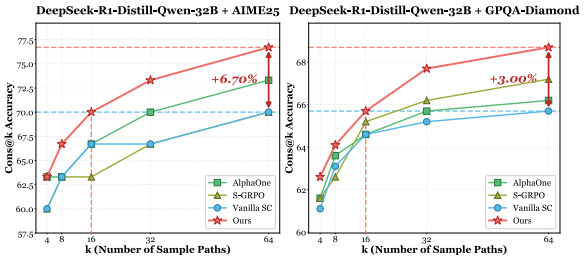
Model	Method	GPQA-Diamond				
		Pass@1↑	FS↓	N/T↓	D/T↓	Repro↓
Qwen3-32B-Thinking	Vanilla	65.3	5.1	5.6	4.0	0.061
	DEER	64.8	4.7	5.4	4.1	0.062
	RL + LP	66.0	4.4	5.2	3.5	0.053
	S-GRRO	66.5	4.6	5.3	3.8	0.056
	RED	**66.8**	**2.3**	**3.2**	**2.3**	**0.020**
	Δ	↑+1.5	↓54.4%	↓43.2%	↓41.5%	↓67.9%
R1-Qwen-32B	Vanilla	60.8	5.3	5.9	4.4	0.071
	DEER	58.9	4.9	5.7	4.4	0.073
	RL + LP	60.4	4.6	5.5	3.9	0.064
	S-GRRO	61.3	4.7	5.6	4.2	0.066
	RED	**61.8**	**2.4**	**3.5**	**2.7**	**0.030**
	Δ	↑+1.0	↓54.3%	↓41.0%	↓37.9%	↓57.9%
R1-Llama-70B	Vanilla	64.5	6.5	6.2	4.8	0.094
	DEER	63.1	5.9	5.9	4.9	0.096
	RL + LP	65.0	5.7	5.6	4.4	0.083
	S-GRRO	65.2	5.9	5.7	4.6	0.086
	RED	**66.2**	**2.9**	**3.5**	**2.8**	**0.030**
	Δ	↑+1.7	↓54.7%	↓43.4%	↓42.2%	↓68.0%

Table 7: Results on GPQA-Diamond with FoE-related metrics.

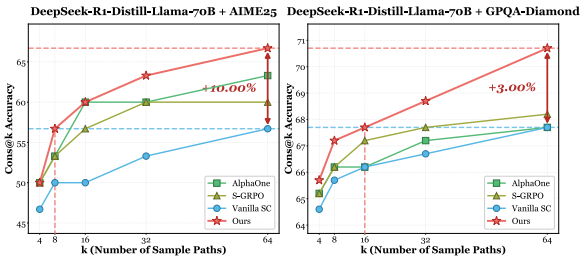
model, **RED** consistently surpasses the baselines. On GPQA-Diamond, **RED** at $k = 16$ matches the performance of the vanilla baseline at $k = 64$ (65.7%), effectively accelerating inference by $4\times$. At $k = 64$, **RED** achieves consistent gains ranging from 3.0% to 10.0% across all datasets and models, underscoring its robustness in scaling test-time computation.

B.3 Ablation Study

Table 8 validates the necessity of combining *Refining First* and *Discarding Subs*. ❶ Removing *Discarding Subs* consistently hurts Pass@1 compared to RED, even though *Refining First* is still applied. This is because subsequent solutions can contaminate the improved first trajectory: later rollouts may reuse or amplify wrong artifacts and eventually override an otherwise high-quality first solution. In addition, keeping subs naturally increases the generation length, leading to substantially more tokens. ❷ Removing *Refining First* while still *Discarding Subs* yields slightly higher Pass@1 than the vanilla setting on most benchmarks. This supports our hypothesis that a major failure mode comes from the interference introduced by subs; early stopping reduces such interference and stabilizes the final decision. However, this variant remains below RED because it lacks targeted interventions on error-prone positions in the first solution, so the model may require more steps to reach a stable and correct answer, which also results in higher token usage than RED. Overall, RED achieves the best trade-off: *Refining First* improves the quality of the



(a) Test-time scalability on Qwen model



(b) Test-time scalability on Llama model

Figure 7: Test-time scalability under self-consistency.

first solution by reducing root errors, and *Discarding Subs* prevents later solutions from perturbing that improved trajectory, jointly improving Pass@1 while reducing tokens.

C Hyperparameter Settings

C.1 Prompt Templates for Dual-Consistency Probing.

In Section 4.3, we periodically probe the current hidden state by appending short prompts that request only the final answer. Since the induced answer is typically fragile in early reasoning, even minor surface-form changes may elicit different answers when the model is not yet confident. We therefore use M=4 stylistically diverse yet semantically equivalent probe templates and require cross-prompt agreement before triggering an early exit. To reduce prompt length to enable fully batched probing, we length-normalize all templates to have the same token count.

- **Template A (imperative / QA-style).** *Stop now immediately and output the final answer only. Use one token. Format: Final Answer: <answer>. No explanation, no extras.*
- **Template B (role / evaluator-style).** *Switch to answer-mode. Output exactly one token inside \boxed { }. Do not justify. Do not add words, symbols, or spaces.*
- **Template C (schema / machine-readable).** *Structured reply: answer=<token>. Output only the value token after '='. No braces, quotes, labels, or commentary. One token strictly now.*
- **Template D (conversational / best-guess).** *Quick check: what's your final answer? Reply with one token only. No reasoning, no repeats, no punctuation. Just answer now.*

Rationale for Prompt Selection. The four templates are intentionally short but differ sharply in surface form and pragmatics while preserving the same semantic request ("return only the final answer token"). They span distinct instruction styles: (A) a strict imperative with an explicit `Final Answer:` anchor; (B) a mode-switch framing with a Latex Style wrapper cue (`\boxed`) to alter formatting habits; (C) a schema like key–value constraint that changes punctuation and discourse structure; and (D) an informal question that tests robustness under a different conversational intent ("quick check / best guess"). These differences induce diverse decoding priors (register, anchors, and syntax), so requiring cross-prompt agreement makes the trigger a strong indicator that the latent answer has stabilized, rather than being a prompt-sensitive artifact of an early, brittle internal state.

C.2 Hyperparameters in Refining `First`.

We set the sliding-window length to $L = 15$ and define the trigger condition as the entropy falling within the local Top-K ($K = 3$) For the entropy-variance threshold T, we perform a sensitivity study on Qwen3-32B-thinking over MATH500, reporting Pass@1 and average generated Token count. We fix all other hyperparameters, including the full Discarding **`Subs`** configuration, and vary only T. As shown in Table 9, values around $2.0 \sim 2.5$ behave similarly, while T=2.4 yields the best joint outcome (highest Pass@1 and lowest tokens), which we adopt as default.

C.3 Hyperparameters in Discarding `Subs`.

We probe every K=2 steps, use M=4 prompt templates, and draw N=12 parallel samples per template (fixed throughout). We ablate only the intra-prompt-template internal consistency threshold P while keeping all other settings fixed (including Refine **`First`**). As shown in Table 10, P=0.6 yields the highest Pass@1 (97.9) and is notably more token-efficient than the stricter P=0.7; lower thresholds (P=0.4, 0.5) save some tokens but incur a large accuracy drop. We therefore use P=0.6 by default.

Model	Method	AIME24		AIME25		MATH500		GSM8K		GPQA-Diamond	
		Pass@1 ↑	Token ↓	Pass@1 ↑	Token ↓	Pass@1 ↑	Token ↓	Pass@1 ↑	Token ↓	Pass@1 ↑	Token ↓
DeepSeek-R1	***w/o* All (Vanilla)**	76.7	11638	67.8	12625	97.2	4716	95.1	1325	70.9	7544
	***w/o* Refining First**	76.9	6739	67.9	7067	97.2	2769	95.0	519	71.0	4179
	***w/o* Discarding Subs**	77.9	9831	69.3	10469	97.6	3887	94.8	947	71.8	5947
	RED (ours)	78.9	6148	70.0	6404	98.1	2408	95.1	447	72.4	3704
Qwen3-8B Thinking	***w/o* All (Vanilla)**	70.0	11125	61.1	12490	95.9	4486	94.1	1573	58.8	6638
	***w/o* Refining First**	70.1	6907	61.3	6815	96.0	2529	93.9	567	58.8	3639
	***w/o* Discarding Subs**	73.8	9127	62.5	10612	96.5	3611	94.4	1019	59.6	5693
	RED (ours)	75.6	5583	63.3	6203	97.0	2301	95.0	465	60.1	3309
Qwen3-32B Thinking	***w/o* All (Vanilla)**	77.8	10677	68.9	11589	96.8	4318	94.3	1435	65.3	5475
	***w/o* Refining First**	78.0	6354	69.0	6729	97.0	2361	94.2	517	65.4	2971
	***w/o* Discarding Subs**	79.1	9017	70.1	9617	97.4	3317	94.5	981	66.3	3749
	RED (ours)	80.0	5793	72.2	5908	97.9	1995	94.6	443	66.8	2477
Dpsk-R1-Distill Qwen-7B	***w/o* All (Vanilla)**	54.4	10438	43.3	11454	91.8	2887	92.4	442	49.2	8016
	***w/o* Refining First**	54.5	4511	43.2	6797	92.0	1359	92.5	289	49.0	4527
	***w/o* Discarding Subs**	55.3	8679	44.8	8589	93.0	1979	93.6	391	50.4	5941
	RED (ours)	57.8	4293	47.8	5690	94.1	1187	94.1	271	51.2	4109
Dpsk-R1-Distill Qwen-32B	***w/o* All (Vanilla)**	70.0	7873	58.9	8906	93.3	2337	93.9	438	60.8	6027
	***w/o* Refining First**	70.2	4327	59.0	5741	93.2	1549	94.0	257	60.8	4091
	***w/o* Discarding Subs**	73.7	6317	60.8	7511	94.2	1962	94.2	339	61.1	5311
	RED (ours)	74.4	3898	63.3	5018	95.2	1347	94.6	209	61.8	3497
Dpsk-R1-Distill LLaMA-8B	***w/o* All (Vanilla)**	45.6	10799	28.9	11548	86.2	3635	92.3	606	46.3	8341
	***w/o* Refining First**	45.5	4879	29.1	5487	86.3	2267	92.5	336	46.1	4727
	***w/o* Discarding Subs**	46.3	8841	30.1	9587	87.6	3076	92.7	493	47.2	6661
	RED (ours)	47.8	4507	34.4	5039	90.1	2009	93.1	283	47.8	3593
Dpsk-R1-Distill LLaMA-70B	***w/o* All (Vanilla)**	68.9	7766	47.8	8909	94.1	2433	94.0	432	64.5	5881
	***w/o* Refining First**	69.0	4491	48.0	5397	94.0	1219	94.1	303	64.7	4077
	***w/o* Discarding Subs**	71.4	5583	49.2	6607	95.6	1969	94.6	381	65.0	4663
	RED (ours)	73.3	3974	50.0	4303	96.2	932	94.9	269	66.2	3563

Table 8: Ablation study.

Model	Setting	MATH500	
		Pass@1↑	Token↓
Qwen3-32B-thinking	T=2.0	97.6	2058
	T=2.1	97.7	2042
	T=2.2	97.6	2027
	T=2.3	97.8	2011
	T=2.4	**97.9**	**1995**
	T=2.5	97.5	2008

Table 9: Sensitivity of the entropy-variance threshold T for Refine `First` on MATH500 (Qwen3-32B-thinking). Best results are highlighted in **bold**.

Model	Setting	MATH500	
		Pass@1↑	Token↓
Qwen3-32B-thinking	P=0.4	87.1	**1741**
	P=0.5	87.4	1877
	P=0.6	**97.9**	1995
	P=0.7	97.4	2434

Table 10: Sensitivity of the internal consistency rate P for Discarding `Subs` on MATH500 (Qwen3-32B-thinking). Best results are highlighted in **bold**.

D Baselines: Further Details

D.1 Introduction of Baselines

To comprehensively evaluate the effectiveness of `RED`, we compare it against a diverse set of baselines. These methods are categorized into three distinct groups: the vanilla backbone model, training-free intervention methods, and reinforcement learning (RL) based strategies.

I) Vanilla Model. We utilize the original backbone Large Reasoning Model (LRM) as the fundamental baseline. It generates Chain-of-Thought (CoT) reasoning and answers using standard decoding without any additional training or dynamic intervention mechanisms.

II) Training-free Methods. These approaches introduce inference-time heuristics to improve efficiency without updating model parameters:

- **DEER** (Yang et al., 2025b) is a dynamic early-exit mechanism that terminates generation based on the geometric mean of token probabilities within a tentative answer.
- **Think or Not** (Yong et al., 2025) employs an entropy-based adaptive stopping strategy,

halting reasoning when the generation entropy exceeds a specific threshold.

- **AlphaOne** (Zhang et al., 2025b) introduces a dynamic switching mechanism using a Bernoulli process to transition between "slow thinking" (reasoning) and "fast thinking" (answering) modes at test time.

III) RL-based Strategies. These methods employ reinforcement learning to explicitly optimize the trade-off between reasoning capability and efficiency:

- **DAST** (Shen et al., 2025) (Difficulty-Adaptive Slow-Thinking) utilizes a difficulty-aware token budget and reward optimization to adaptively adjust reasoning length based on problem complexity.

- **RL + Length Penalty** (Arora and Zanette, 2025) modifies the reward function to penalize correct answers that deviate from the average length, encouraging concise reasoning.

- **GRPO** (DeepSeek-AI et al., 2025) (DeepSeek-R1) applies Group Relative Policy Optimization to incentivize reasoning capabilities by comparing a group of candidate outputs and updating the policy based on normalized advantages.

- **S-GRPO** (Dai et al., 2025) extends the GRPO framework with a Serial Group Decay Reward strategy, assigning decaying rewards to different reasoning steps to train the model in identifying optimal early-exit points.

D.2 Baseline Settings

For all baselines, we strictly adhere to the settings detailed in their respective original papers. To ensure fair comparison, we keep the same inference hyperparameters across methods (our vanilla decoding): sampling with temperature T=0.6 and top-p=0.95, max-token=16000, and we report Pass@1 averaged over three repeated runs. To ensure optimal performance, we utilize the official chat templates corresponding to each model family (e.g., applying the specific formats for Qwen and Llama, respectively). For RL-based baselines, training is conducted on a cluster of NVIDIA A100 GPUs with bf16 mixed precision, DeepSpeed ZeRO-3 sharding, gradient checkpointing, and identical reward extraction / answer parsing logic across methods (An et al., 2025; Liu et al., 2026).

E Experimental Results of Rollback in Subs.

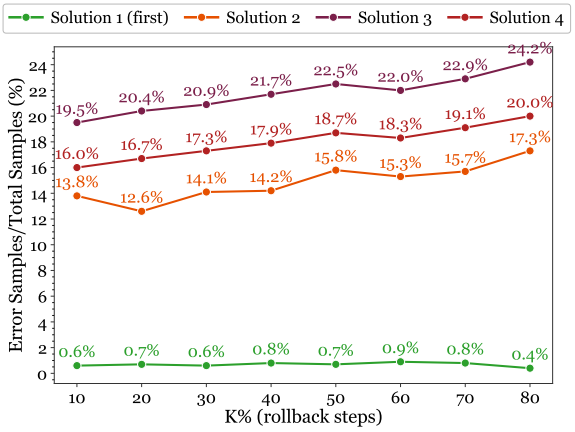


Figure 8: Average sampling error rate (%, lower is better) under rollback sampling. For each solution variant, we truncate an originally correct trajectory at rollback ratio $K \in \{10, 20, \ldots, 80\}\%$, restore the KV cache at the truncation point, and re-sample the remaining continuation $N = 100$ times; Curves report error rates averaged over problems. FIRST remains highly robust (error $< 1\%$ across all K), whereas SUBS exhibit substantially higher and largely K invariant error rates, revealing persistent latent instability even when the original final answers are correct.

Observations and Motivations. Through case study we find that subsequent solutions (`Subs`) can still contain noticeably unreliable or even incorrect intermediate steps, even when their *final* answers appear correct. Therefore, a one-shot correct outcome does not necessarily imply a stable reasoning trajectory. This raises a latent risk: if the generation process is perturbed (e.g., by interrupting and resampling the continuation), these "apparently correct" `Subs` may drift to an incorrect final answer more easily than the first solution.

Rollback-sampling Experiment. To quantify the robustness of each solution's reasoning process, we design a *rollback sampling* experiment on BS-17k-subset using Qwen3-8. The core idea is to truncate a solution that is correct in the original run, and then repeatedly re-sample its continuation from the same intermediate state, measuring how often the final answer becomes incorrect. We evaluate the first four solutions for each problem, denoted as Solution 1–Solution 4, where Solution 1

corresponds to `First` and Solution 2–Solution 4 are `Subs`. For metrics involving Solution k, the average is computed over the subset of problems for which at least k solutions were successfully sampled. The procedure is as follows.

Instance Selection. We collect problems for which the model can produce a correct final answer in a single pass, and for each solution variant under evaluation, we only keep instances where that solution is correct in the original run. This isolates stability under correctness.

Rollback Point. For a solution s with reasoning length T_s, we choose a rollback ratio $K \in \{10, 20, \ldots, 80\}$ and truncate the trajectory at

$$t = \left\lfloor \frac{K}{100} T_s \right\rfloor . \tag{3}$$

We then roll back the generation state to this truncation point and keep the prefix up to step t. In practice, we store the KV cache during decoding and restore it at the rollback point, so that we can resume sampling from the identical intermediate state without recomputing the prefix.

Re-sampling and Error-rate Estimation. Starting from the restored intermediate state, we independently sample the continuation $N = 100$ times until completion, producing 100 final answers. We define the sampling error rate of solution s at rollback ratio K as

$$\widehat{e}_s(K) = \frac{1}{N} \sum_{n=1}^{N} \mathbf{1}\left[\hat{y}_{s,K}^{(n)} \neq y^\star\right], \tag{4}$$

where $y^\star$ is the ground-truth answer and $\hat{y}_{s,K}^{(n)}$ is the final answer from the n-th re-sampled continuation. Averaging over all retained problems yields $\bar{e}_s(K)$.

Experimental Results. Figure 8 plots $\bar{e}_s(K)$ against K. `First` is extremely robust: across all rollback ratios, the error rate is consistently below 1% (mean $\approx 0.69\%$, range $0.4\% \sim 0.9\%$). This indicates that the intermediate states along `First` reliably constrain the continuation toward the correct answer, and re-sampling rarely deviates from the correct trajectory. In contrast, `Subs` exhibit pronounced fragility. Solution 2 has a mean error rate of about 14.85% (range $12.6\% \sim 17.3\%$), Solution 3 reaches about 21.76% (range $19.5\% \sim 24.2\%$), and Solution 4 stays around 18.0% (range $16.0\% \sim 20.0\%$). Importantly, these error rates remain consistently high across rollback points: sweeping K from 10% to 80% changes the error probabilities only mildly. This suggests that the risk is not concentrated in a small set of "critical" steps near the end; instead, later solutions follow comparatively risky reasoning paths throughout the trajectory. If we aggregate Solutions 2–4 as `Subs`, the mean error rate is $\approx 18.20\%$. Compared with `First` (mean $\approx 0.69\%$), the relative error probability under interruption-and-resampling is

$$\frac{\bar{e}_{\texttt{First}}}{\bar{e}_{\texttt{Subs}}} \approx \frac{0.69}{18.20} \approx 3.8\%, \tag{5}$$

which is consistent with the main-text claim that `First` incurs only a small fraction of the rollback-induced error probability of `Subs`.

Conclusion Aligned with Observation Rollback sampling directly validates the "latent risk" in `Subs`: even when `Subs` happens to be correct in a single run, its intermediate states provide weaker "lock-in" toward the correct answer, so re-sampling from an interruption point can readily slip to an incorrect final answer. In future work, we plan to explore the application of our method to multimodal reasoning (Jiang et al., 2026, 2024; Xiao et al., 2025, 2026a,b; Li et al., 2026b,c; Chen et al., 2025, 2026; Liu et al., 2024b, 2025b; Li et al., 2025a; Zhao et al., 2026).

F FoE Initialization: Further Details

F.1 Human-LLM Agreement of Few-Shot PCS Judging

Purpose. In Section 3.1, we initialize the Forest of Errors (FoE) by defining a parent-child association score $\mathrm{Score}(e_i, e_j)$ on a 1-5 scale, and using an advanced LLM with few-shot prompting to score candidate parents for each child error in a near-to-far order, accepting the first candidate whose score exceeds a threshold τ. Since this few-shot judging step directly determines FoE edges, we conduct a human-LLM agreement study to validate that the LLM judge aligns with expert causal judgments.

Task and inputs. Formally, we define the input space for our experiments such that each evaluation instance is structured as a triple $(\mathrm{CONTEXT\_UP\_TO\_CHILD}, e_i, e_j)$:

(i) the full reasoning prefix up to and including the child error node e_j, (ii) an earlier candidate parent error node e_i, and (iii) the target child error node e_j. Both humans and the LLM judge output a

PCS (Parent-Children Score) in $[1.0, 5.0]$ with one decimal place, following the same rubric of *direct artifact reuse and directness*.

Data. We sample instances from the same benchmark suite used throughout the paper (AIME25, MATH500, GSM8K, and GPQA-Diamond). To reflect the near-to-far evaluation regime in Section 3.1, we stratify sampling by temporal distance between e_i and e_j: (1) **Near** pairs, where e_i is among the 1-3 nearest preceding error nodes of e_j; and (2) **Far** pairs, where e_i occurs at least 6 error nodes before e_j. Our final evaluation set contains $N = 500$ pairs, balanced across math/science domains and near/far strata.

Human annotation protocol. We recruit four expert annotators, covering both mathematical and scientific reasoning backgrounds. Annotators independently assign PCS scores using a shared guideline sheet (the same definitions/anchors used by the few-shot prompt). We use the median of four scores as the human reference. To quantify human consistency, we report Krippendorff's α (ordinal) across the four annotators. We also compute a leave-one-out (LOO) human agreement at the decision threshold $\tau = 4.0$ (each annotator vs. the majority of the other three), to contextualize the LLM agreement relative to typical human variance.

LLM judging. We run the PCS few-shot prompt (Appendix A) with a frontier LLM under deterministic decoding (temperature 0). The judge sees exactly the same $(\texttt{CONTEXT\_UP\_TO\_CHILD}, e_i, e_j)$ tuple as the human experts, and outputs only the numeric PCS score.

Metrics. We report: (i) **score-level correspondence** via Spearman correlation ρ and mean absolute error (MAE) against the median human score; (ii) **ordinal agreement** via quadratic weighted kappa (QWK) after mapping PCS to the nearest integer in $\{1, 2, 3, 4, 5\}$; and (iii) **decision agreement** at $\tau = 4.0$, treating PCS≥ 4.0 as a positive edge (accuracy and F1).

Results. Table 11 summarizes the agreement performance. **Separately, to validate the consistency of the human reference,** we measured that human experts achieved a Krippendorff's α of 0.78, establishing a reliable consensus baseline for the PCS rubric. The LLM judge aligns strongly with this baseline, yielding a Spearman's $\rho = 0.88$ and a low MAE of 0.28. Crucially for FoE construction, at the $\tau = 4.0$ threshold, the LLM reaches **94% accuracy and 0.92 F1**, matching (and slightly exceeding) the **93% LOO human agreement**. As shown in Figure 9, the error distribution is heavily skewed toward zero, confirming that few-shot prompting effectively emulates expert reasoning in identifying artifact reuse.

Dataset	#Pairs	ρ	QWK	MAE	Acc@τ	F1@τ
AIME25	150	0.90	0.84	0.24	0.94	0.93
MATH500	150	0.87	0.82	0.27	0.93	0.92
GSM8K	100	0.85	0.79	0.30	0.92	0.91
GPQA-Diamond	100	0.84	0.80	0.32	0.92	0.90
Overall	500	0.88	0.82	0.28	0.94	0.92

Table 11: Human-LLM agreement for PCS few-shot judging. Spearman's ρ and MAE are computed on raw one-decimal PCS scores. QWK is computed after rounding PCS to the nearest integer in $\{1, 2, 3, 4, 5\}$. Acc@τ and F1@τ treat PCS≥ 4.0 as a positive edge.

Order	Candidate	Short description	Human PCS	LLM PCS
Nearest	e_{j-1}	Wrong binding reused by child (T=13)	4.9	5.0
2nd	e_{j-2}	Ancestor value error (total = 13)	3.6	3.7
Far	e_{j-6}	Same segment, no artifact reuse	2.1	2.0

Table 12: Single-case example of near-to-far candidate scoring for one child error node. The LLM and humans agree on the first candidate reaching τ=4.0, which becomes the parent.

Single-case sanity check (near-to-far parent selection). Table 12 validates this near-to-far selection: both humans and the LLM assign PCS$\geq \tau$ solely to the closest candidate containing the reused artifact, yielding identical parent assignment.

Conclusion. The human-LLM agreement results suggest that the proposed PCS few-shot judging protocol produces evaluations that are largely consistent with expert human judgments. Across both mathematical and scientific benchmarks, the LLM judge exhibits agreement levels comparable to human inter-annotator consistency, at both the continuous score level and the thresholded decision level used for FoE edge construction (τ=4.0). These findings indicate that the few-shot prompt captures key signals of direct error induction via artifact reuse, rather than relying solely on surface proximity or topical overlap. While the LLM judge is not intended to replace human expertise in all settings, the observed agreement provides empirical support for its use as a practical and scalable component in FoE initialization within our framework.

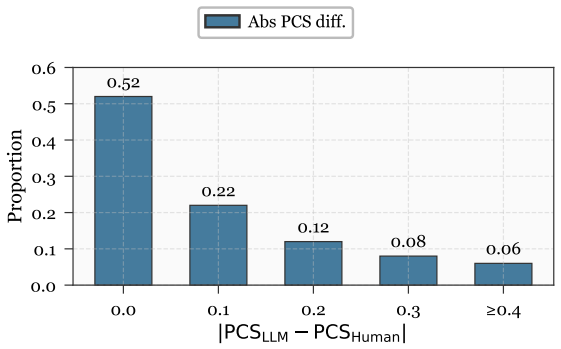


Figure 9: Distribution of absolute PCS disagreement between the LLM judge and the median human score.

F.2 Forest Modeling: Complete Details

Modeling details. Given a set of identified error nodes within a reasoning trace (e.g., via the o1-based annotation method (Yang et al., 2025c; Lin et al., 2025; Li et al., 2025b; Wu et al., 2025; Liu et al., 2021, 2025a, 2023; Yuan et al., 2026; Yuan and Zhang, 2025a,b; Chang et al., 2024, 2026, 2025)), we denote the chronologically ordered sequence as $E = [e_1, e_2, \ldots, e_n]$. To reconstruct the causal structure, we evaluate the likelihood that an earlier error e_i ($i < j$) directly induces a subsequent error e_j. We define a parent–child association score s_{ij}, quantified via the few-shot prompting protocol detailed in Appendix L:

$$s_{ij} \triangleq \textsc{Score}(e_i, e_j). \tag{6}$$

Original scores are assessed on a scale of $[1, 5]$. For computational consistency, we normalize these values to $[0.2, 1]$. We designate a predecessor as a valid candidate parent only if the score meets a strict threshold τ. As specified in our design, we set $\tau = 0.8$ (corresponding to a raw score of 4.0). Accordingly, the set of valid candidate parents $\mathcal{P}_j$ for node e_j is defined as:

$$\mathcal{P}_j \triangleq \{\, i \in [1, j-1] : s_{ij} \geq \tau \,\}. \tag{7}$$

To ensure a forest structure where each node has at most one parent, we select the temporally closest candidate (i.e., the largest index) from $\mathcal{P}_j$. The parent index $\pi(j)$ is formally determined by:

$$\pi(j) \triangleq \begin{cases} \max \mathcal{P}_j, & \mathcal{P}_j \neq \emptyset, \\ 0, & \mathcal{P}_j = \emptyset. \end{cases} \tag{8}$$

Here, $\pi(j) = 0$ indicates that e_j has no antecedent satisfying the threshold condition. In such cases, e_j is treated as a root node, instantiating a new *Tree of Errors* (ToE) within the *Forest of Errors* (**FoE**). Otherwise, a directed edge $e_{\pi(j)} \to e_j$ is added to the ToE containing $e_{\pi(j)}$. By iterating j from 1 to n, this procedure constructs the complete forest, as summarized in Algorithm 1.

F.3 Empirical Findings on Error Node Fixing

This subsection provides controlled intervention evidence to support the observations in Section 3.1: *merely rectifying child nodes leaves the (uncleared) ancestor error active, which continues spawning new offspring, whereas correcting the root node substantially decelerates subsequent error-node generation.*

Common setup. We operate on the constructed Forest of Errors (FoE) in a given *Sub* solution and use the same parent–child scoring function as in §3.1. For convenience, we report a normalized score $\textsc{nPCS} \triangleq \text{PCS}/5 \in [0.2, 1]$ so that the edge threshold in the main paper ($\text{PCS} \geq 4.0$) corresponds to $\textsc{nPCS} \geq \tau$ with $\tau = 0.8$.

Online correction of descendants induces sibling regeneration. We test whether correcting a non-root node can stop error propagation when its ancestor remains uncorrected. Given a formed local ToE $R \to C_1 \to G_1$ (root/child/leaf), once the leaf error G_1 appears, we *immediately correct* G_1 (e.g., replace the erroneous intermediate artifact with the correct one or insert a corrective instruction), while keeping C_1 and R intact. We then let the model continue for a fixed post-fix window and detect newly generated errors. If any newly generated error e satisfies $\textsc{nPCS}(C_1, e) \geq \tau$, we regard it as a regenerated sibling under the same parent (i.e., C_1 keeps spawning offspring despite the leaf fix). We further repeat the same procedure at the next stage: correct C_1 (while keeping R intact) and test whether R spawns a new child C_2 such that $\textsc{nPCS}(R, C_2) \geq \tau$.

To quantify the prevalence of this phenomenon beyond individual cases, we define the **Error Spawning Rate** as:

$$\textsc{Spawn@}\Delta \triangleq \mathbb{E}_{u \sim \mathcal{I}} \left[\mathbf{1}_{\mathcal{A}(u)} \right], \tag{9}$$

where $\mathcal{A}(u)$ denotes the event that there exists at least one new error node $e \in E_{\Delta}^{\text{post}}(u)$ such that $\textsc{nPCS}(\text{Anc}(u), e) \geq \tau$. Here, $\mathcal{I}$ is the set of intervention points (each involving the correction of

Algorithm 1 Constructing the `FoE` by linking each error to its nearest inducing predecessor.

Require: Chronological error list $E = [e_1, \ldots, e_n]$; threshold τ; PCS scorer $\textsc{Score}(\cdot, \cdot) \in [1, 5]$.
Ensure: Forest $\mathcal{F}$ (a set of ToEs) and parent map $\pi : \{1, \ldots, n\} \to \{0, \ldots, n\}$.

$\mathcal{F} \leftarrow \emptyset$; $m \leftarrow 0$ ▷ m is the number of ToEs in $\mathcal{F}$
$\textsc{TreeID}[1..n]$ ▷ maps each node to its ToE index
for $j \leftarrow 1$ **to** n **do**
  $p \leftarrow 0$ ▷ 0 denotes "no parent" (root)
  for $i \leftarrow j - 1$ **down to** 1 **do** ▷ nearest-to-farthest
    $s \leftarrow \textsc{Score}(e_i, e_j)$
    if $s \geq \tau$ **then**
      $p \leftarrow i$; **break**
    end if
  end for
  if $p = 0$ **then**
    $m \leftarrow m + 1$
    Initialize a new ToE $\mathcal{T}_m$ with root e_j
    $\mathcal{F} \leftarrow \mathcal{F} \cup \{\mathcal{T}_m\}$; $\textsc{TreeID}[j] \leftarrow m$
  else
    Add edge $e_p \to e_j$ to ToE $\mathcal{T}_{\textsc{TreeID}[p]}$
    $\textsc{TreeID}[j] \leftarrow \textsc{TreeID}[p]$
  end if
  $\pi(j) \leftarrow p$
end for
return $\mathcal{F}, \pi$

a non-root node u while leaving its nearest uncorrected ancestor $\mathrm{Anc}(u)$ intact), and $E_{\Delta}^{\text{post}}(u)$ represents the set of newly generated error nodes within the subsequent Δ decoding steps following the correction. Intuitively, $\textsc{Spawn}@\Delta$ measures the probability that an *uncleared* ancestor remains active and generates at least one new erroneous child after only its descendant has been corrected.

Our empirical results on the BS-17k-subset yield a $\textsc{Spawn}@15$ value of 0.842, indicating that in over 84% of cases, rectifying a descendant without addressing its root cause fails to halt the error cascade. This high spawning rate confirms that error nodes are structurally coupled with their ancestors rather than isolated events. The uncorrected node persisting in the context acts as a persistent "error factory," continuously driving the model toward new erroneous states.

Correcting the root of a formed tree suppresses subsequent growth. We next test whether root correction can decelerate node generation even when the tree has already formed. During normal decoding, when we identify a small formed error tree, we duplicate the KV cache at that moment and branch into two continuations: (i) **Fix-Root** branch: correct the root error immediately at the branching point; (ii) **No-Fix** branch: keep the root unchanged. We then let both branches continue decoding forward until a stop token is reached and remodeling the resulting Tree in each branch.

Metric	Fix-Root	No-Fix
$\lvert V \rvert$	**0.42**×	1.00×
$\bar{R}$	0.35×	1.00×
D	0.51×	1.00×

Table 13: Quantitative impact of root-node intervention on the `FoE` structure. All metrics are reported as relative ratios to the **No-Fix** baseline (1.00×), averaged over the BS-17k-subset. **Metric definitions:** $\lvert V \rvert$ denotes the total number of error nodes per tree; $\bar{R}$ is the error reproduction rate; and D represents the average tree depth.

Takeaway and mechanism-level explanation. Tables 13 and online correction experiments jointly validate observations in Section 3.1 from two complementary angles. The first experiment shows that correcting a descendant node only patches a *surface artifact* while the ancestor error state remains in-context, making it easy for subsequent steps to reinstantiate new children that are still causally

attributable to the same ancestor. The second experiment isolates the ancestor effect via KV-cache branching: once the root is corrected, the downstream decoding state is steered away from the erroneous causal source, substantially suppressing the growth of the error tree. This supports our core claim that the *root node dominates the effective reproduction process* in FoE, and hence root correction is the decisive operation for decelerating error-node generation in FoE.

G Formal Definitions and Measurement Methods for the Three Dimensions of Reflection

We study *intra-solution reflection*, i.e., self-check behaviors that occur within a single generated solution trajectory, and define how to measure it along three dimensions: *frequency*, *completeness*, and *depth*. In many test-time improvement frameworks, LLMs/LRMs generate self-feedback, roll back and revise earlier steps, or write reflective text to guide subsequent reasoning (Shinn et al., 2023).

G.1 Preliminaries: reflection instances and notation.

Let $p \in \mathcal{P}$ denote a problem. In one run, the model may produce multiple solutions; we use $s \in \{1, \dots, S_p\}$ to denote the s-th solution and denote its full text as $T_{p,s}$. We define a *reflection instance* as a *maximal contiguous span* in $T_{p,s}$ where the model explicitly performs self-verification or self-doubt (e.g., questioning assumptions, checking derivations, spotting potential mistakes, or proposing a revision plan). This definition operationalizes reflection as explicit self-feedback text produced during generation (Madaan et al., 2023). Using a strong closed-source model as the judge, since reflection spans are sparse and marked by salient cues, a single-pass judgment suffices to reliably identify them. Each $T_{p,s}$ is segmented into an alternating sequence of *normal-reasoning spans* and *reflection spans*. We define the set of reflection instances (equivalently, reflection start points) in solution (p, s) as $\mathcal{I}_{p,s}$, and denote its size by $N_{p,s}$:

$$N_{p,s} := |\mathcal{I}_{p,s}|. \tag{10}$$

For any $i \in \mathcal{I}_{p,s}$, let $X_{p,s,i}$ denote the breakpoint prefix (the full prefix up to, but excluding, the i-th reflection span), and let $Y_{p,s,i}$ denote the corresponding reflection span in the original run. Equivalently, with the operators $\mathrm{Pref}(\cdot)$ and $\mathrm{Span}(\cdot)$ defined by the above segmentation,

$$\begin{aligned} X_{p,s,i} &:= \mathrm{Pref}(T_{p,s}, i), \\ Y_{p,s,i} &:= \mathrm{Span}(T_{p,s}, i). \end{aligned} \tag{11}$$

Intuitively, $X_{p,s,i}$ is the *breakpoint prefix* and $Y_{p,s,i}$ is the model's *spontaneous* reflection at that breakpoint.

G.2 Reflection frequency (FRQ).

Definition and meaning. For solution (p, s), reflection frequency is the number of reflection instances:

$$\mathrm{FRQ}_{p,s} := N_{p,s}. \tag{12}$$

For a fixed solution index s, we aggregate over problems that contain the s-th solution:

$$\begin{aligned} \mathcal{P}_s &:= \{p \in \mathcal{P} : s \le S_p\}, \\ \mathrm{FRQ}_s &:= \frac{1}{|\mathcal{P}_s|} \sum_{p \in \mathcal{P}_s} \mathrm{FRQ}_{p,s}. \end{aligned} \tag{13}$$

FRQ_s measures how often the model *initiates* reflection within solution s.

G.3 Reflection completeness (COM).

Definition and meaning. Completeness measures whether a spontaneous reflection instance executes a *structurally complete self-check episode* at the breakpoint: it should cover the key corrective actions implied by the model's own best spontaneous continuation, rather than starting to reflect but prematurely switching back to forward reasoning.

Measurement method and formalization Measurement method and formalization (same-prefix, minimally-intervened spontaneous upper bound). Fix a weak continuation instruction γ (e.g., "Please continue your reasoning; you may check and correct if needed.", without explicitly demanding reflection). For each breakpoint prefix $X_{p,s,i}$, we sample M continuations from the *evaluated model* under the same prefix:

$$\tilde{Y}^{(m)}_{p,s,i} \sim \pi(\cdot \mid X_{p,s,i} \circ \gamma),\ m = 1, \dots, M. \tag{14}$$

This multi-sample selection follows the self-consistency intuition of exploring diverse continuations and selecting the best candidate under a fixed prefix (Wang et al., 2022). A strong judge model selects the most complete candidate:

$$\tilde{Y}^{(\mathrm{best})}_{p,s,i} \in \{\tilde{Y}^{(m)}_{p,s,i}\}_{m=1}^{M}. \tag{15}$$

From $\tilde{Y}^{(\text{best})}_{p,s,i}$, the judge extracts an *unordered* checklist of atomic, executable, correction-relevant actions, denoted by $\mathcal{C}_{p,s,i}$ with size $K_{p,s,i}$:

$$\begin{aligned} \mathcal{C}_{p,s,i} &:= \{c_k\}_{k=1}^{K_{p,s,i}}, \\ K_{p,s,i} &:= |\mathcal{C}_{p,s,i}|. \end{aligned} \tag{16}$$

We write $Y_{p,s,i} \succeq c_k$ if the judge determines that $Y_{p,s,i}$ explicitly states or semantically entails the corrective action c_k. Completeness is defined as checklist coverage:

$$\mathrm{COM}_{p,s,i} := \frac{1}{K_{p,s,i}} \sum_{k=1}^{K_{p,s,i}} \mathbf{1}[\, Y_{p,s,i} \succeq c_k \,]. \tag{17}$$

We report the average completeness over *all reflection instances* in solution index s:

$$\begin{aligned} \mathcal{P}_s^{\text{ref}} &:= \{p \in \mathcal{P}_s : N_{p,s} > 0\}, \\ \mathrm{COM}_s &:= \frac{\sum_{p \in \mathcal{P}_s^{\text{ref}}} \sum_{i \in \mathcal{I}_{p,s}} \mathrm{COM}_{p,s,i}}{\sum_{p \in \mathcal{P}_s^{\text{ref}}} N_{p,s}}. \end{aligned} \tag{18}$$

Thus, COM_s captures the average *structural completeness* of spontaneous reflection instances within solution s.

G.4 Reflection depth (DEP).

Definition and meaning. Definition and meaning. Depth measures whether reflection progresses from identifying *surface-level symptoms* to uncovering the *true underlying cause* and articulating a *correct correction route*. This aligns with the view that reliable reasoning benefits from step-level verification and process-aware evaluation of correction behaviors.

Measurement method and formalization. Measurement method and formalization (external oracle aligned to the correct correction path). For each breakpoint prefix $X_{p,s,i}$, we query an external, stronger closed-source model as an oracle to output a *minimal ordered* correction path that leads to a correct conceptual fix:

$$\begin{aligned} \mathcal{R}^\star_{p,s,i} &= (r^\star_1, r^\star_2, \ldots, r^\star_{D^\star_{p,s,i}}), \\ D^\star_{p,s,i} &:= |\mathcal{R}^\star_{p,s,i}|. \end{aligned} \tag{19}$$

A judge determines how many prefix steps of the oracle path are completed by the model's spontaneous reflection $Y_{p,s,i}$. We write $Y_{p,s,i} \trianglerighteq (r^\star_1, \ldots, r^\star_j)$ if the judge determines that $Y_{p,s,i}$ completes the first j steps of $\mathcal{R}^\star_{p,s,i}$ in order. Then the reached depth is

$$\begin{aligned} &\mathcal{D}_{p,s,i} := \{0, 1, \ldots, D^\star_{p,s,i}\}, \\ &\mathcal{J}_{p,s,i} := \{\, j \in \mathcal{D}_{p,s,i} \mid Y_{p,s,i} \trianglerighteq (r^\star_1, \ldots, r^\star_j) \,\}, \\ &D_{p,s,i} := \max \mathcal{J}_{p,s,i}. \end{aligned} \tag{20}$$

Depth is defined as the reached-step ratio (reported as a percentage when needed):

$$\mathrm{DEP}_{p,s,i} := \frac{D_{p,s,i}}{D^\star_{p,s,i}}. \tag{21}$$

We aggregate depth over *all reflection instances* in solution index s:

$$\mathrm{DEP}_s := \frac{\sum_{p \in \mathcal{P}_s^{\text{ref}}} \sum_{i \in \mathcal{I}_{p,s}} \mathrm{DEP}_{p,s,i}}{\sum_{p \in \mathcal{P}_s^{\text{ref}}} N_{p,s}}. \tag{22}$$

Thus, DEP_s measures how deeply spontaneous reflections in solution s advance along a correct correction trajectory.

H The Reason for Node Generation: Further Analysis

This appendix complements Section 3.3 with (i) a precise definition of the entropy features, (ii) filled quantitative summaries corresponding to Fig. 4, (iii) robustness to the window length, and (iv) a statistically grounded testing protocol. All experiments are conducted on BS-17k-subset using Qwen3-8B, following the setup in Section 3.3.

Entropy features and quadrant partition. At decoding step t, the model outputs a token distribution $p_t(\cdot)$. We define token entropy

$$H_t \;=\; -\sum_{x \in \mathcal{V}} p_t(x) \log p_t(x), \tag{23}$$

and compute window statistics with $\mathcal{W}_t = \{\max(1, t-L+1), \ldots, t\}$:

$$\begin{aligned} h_t &= \frac{1}{|\mathcal{W}_t|} \sum_{i \in \mathcal{W}_t} H_i, \\ v_t &= \frac{1}{|\mathcal{W}_t| - 1} \sum_{i \in \mathcal{W}_t} (H_i - h_t)^2. \end{aligned} \tag{24}$$

We use L=15 unless stated otherwise, and sweep $L \in [10, 20]$ for robustness. We split steps into four percentile-based regions using the 75th-percentile thresholds of h_t and v_t (computed *within* each setting, `First/Subs`): LL (low/low), HL (high-h only), LH (high-v only), and HH (high/high).

Trigger metrics. We report node-trigger rate (NTR), root-trigger rate (RTR), and average node depth (AND) as defined in Section 3.3.

Region (h/v)	First			Subs		
	NTR	RTR	AND	NTR	RTR	AND
LL (low/low)	0.082	0.014	1.12	0.115	0.028	1.25
HL (high-h only)	0.245	0.063	1.48	0.312	0.094	1.62
LH (high-v only)	0.198	0.052	2.85	0.224	0.071	3.14
HH (high/high)	**0.312**	**0.187**	**2.14**	**0.386**	**0.264**	**2.41**

Table 14: Node generation statistics across entropy quadrants (default L=15). NTR, RTR, and AND metrics are reported for both `First` and `Subs` settings.

Feature Set	AUC (First) ↑	AUC (Subs) ↑
h only	0.642	0.658
v only	0.615	0.631
$h + v$	0.724	0.742
$h + v + (h \times v)$	**0.816**	**0.835**

Table 15: Predictive ablation for root-node triggers using logistic regression. AUC scores quantify the predictive power of entropy (h) and variance (v) features. The interaction term ($h \times v$) provides a substantial gain.

Filled numeric summaries (counterpart of Fig. 4). Table 14 substantiates the Observation that high entropy alone (HL) primarily increases error frequency (higher NTR) while producing mostly shallow nodes (low AND). High variance alone (LH) shifts errors to higher structural levels (largest AND) but does not maximize root-node generation. In contrast, the high-high region (HH) exhibits the peak RTR in both settings (`First`: 0.187; `Subs`: 0.264), confirming that *simultaneously* elevated uncertainty and volatility is the most indicative regime for root-node emergence. Moreover, under identical regions, `First` consistently yields lower RTR than `Subs` (e.g., HH: 0.187 vs 0.264), supporting the main-text claim that `First` is less likely to spawn structural root errors.

Predictive ablation: why a single signal is insufficient. We fit logistic predictors for whether a decoding step triggers a *root* node using different feature sets. Table 15 shows that neither h nor v alone is sufficient. Combining them improves AUC, while adding the interaction term ($h \times v$) yields the best performance, confirming that *jointly* high entropy and high variance is the strongest predictor of root-node triggers.

Robustness to the entropy window length. Sweeping $L \in [10, 20]$ preserves the conclusion (Table 16): the HH region consistently yields the highest RTR, and the gap between `First` and `Subs` remains stable.

Window L	RTR_{HH} (First) ↑	RTR_{HH} (Subs) ↑
10	0.174	0.251
12	0.182	0.258
15 (Default)	**0.187**	**0.264**
18	0.185	0.261
20	0.179	0.255

Table 16: Robustness to window length L. Across all $L \in [10, 20]$, the HH region consistently yields the highest RTR, with `First` maintaining a lower trigger rate than `Subs` under identical conditions.

Comparison (RTR)	First		Subs	
	OR ↑	Holm-p ↓	OR ↑	Holm-p ↓
HH vs. LL	16.20	$< 10^{-12}$	12.45	$< 10^{-12}$
HH vs. HL	3.42	2.4×10^{-7}	3.46	1.1×10^{-8}
HH vs. LH	4.19	8.6×10^{-9}	4.69	3.2×10^{-10}
`First` vs. `Subs`	**OR = 0.64** (Holm-$p = 0.0042$)			

Table 17: Effect sizes and statistical significance for root-node generation. Odds ratios (OR) quantify the relative increase in root-trigger probability, with p-values Holm-Bonferroni corrected. **Note**: The cross-setting OR (0.64) is calculated for `First` relative to `Subs` as the reference.

Statistical significance. To test whether HH is statistically distinct in producing root nodes, construct 2×2 contingency tables over decoding steps and apply Fisher's exact test (two-sided) for HH vs. each region $r \in \{\text{LL,HL,LH}\}$. Correct for multiple comparisons with Holm (or Bonferroni) correction and report corrected p-values. Since p-values require raw step counts, we additionally provide sample-size-free effect sizes via odds ratios computed from observed RTRs:

$$\text{OR}(p, p') = \frac{p/(1-p)}{p'/(1-p')}. \tag{25}$$

Table 17 lists the resulting odds ratios; fill Holm-corrected p-values from your count-based tests.

Statistical Analysis. As shown in Table 17, the high–high (HH) region is a statistically distinct precursor for root-node emergence.

(i) Internal Comparison: Compared to the baseline LL region, the HH region exhibits an exceptionally high odds ratio (OR > 12, $p < 10^{-12}$), indicating that the co-occurrence of high entropy and high variance increases the odds of a root-node trigger by over an order of magnitude. Even when compared to regions where only one metric is high (HL or LH), the HH region maintains a significant lead (OR ≈ 3.4–4.7, $p < 10^{-6}$), confirming that neither h_t nor v_t alone is a sufficient predictor.

Variant	ESC (%)	WESR (%)	Crisk@Agree (%)
A	73.20	0.88	1.20

Table 18: Baseline A.

(ii) Cross-Setting Comparison: Within the HH region, `First` demonstrates a significantly lower likelihood of triggering a root node than `Subs` (OR $= 0.64$, $p < 0.01$). This substantiates that the `First` setting is structurally more robust even when the model enters a high-uncertainty state, whereas `Subs` is more prone to cascading structural failures under identical entropy dynamics.

I Discarding `Subs`: Details and Analysis

Goal and setting. This subsection studies when it is safe to discard `Subs` and stop decoding early. We conduct the analysis on Qwen3-8B-Thinking using the BS-17k-subset. Our probing-based early stop design is decomposed into three ingredients. Component (A) checks *convergence* by repeatedly sampling answers under one fixed probe prompt, and verifying if the modal answer's proportion exceeds a consistency threshold P. Component (B) checks *robustness* by enforcing agreement across multiple diverse probe prompts. Component (C) reduces sampling variance by drawing multiple parallel samples per prompt. Below, we analyze each component in isolation, visualize typical success and failure trajectories, and then connect them through a unified reverse ablation in Table 21.

Periodic probing protocol. During generation, we pause every K tokens at checkpoint positions $t \in \{K, 2K, \dots\}$, reuse the current KV cache, append a short probe prompt that asks for the final answer only, and decode short answers with stochastic sampling. Let M denote the number of probe prompts and N denote the number of parallel samples per prompt. For any checkpoint t and prompt index m, we obtain samples $\{A_t^{(m,n)}\}_{n=1}^{N}$ and form the empirical answer histogram. Throughout this subsection, a *trigger* means we stop at some checkpoint and output the induced answer at that checkpoint.

Baseline A: internal convergence is necessary but not sufficient. Baseline A implements the simplest trigger: choose one probe prompt, draw N samples at each checkpoint, and trigger once the empirical modal proportion exceeds a fixed threshold P. Figure 10 illustrates the intended behavior:

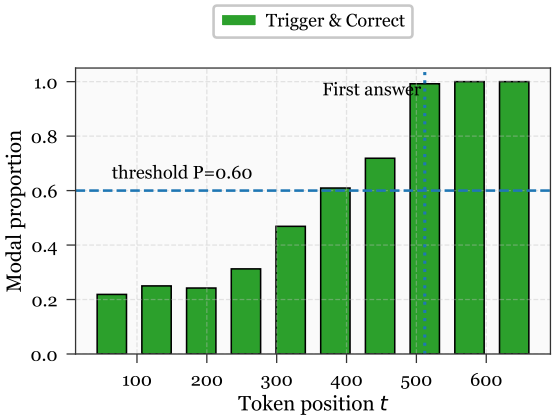


Figure 10: **Convergence (A, success).** For a fixed probe prompt, the modal answer ratio (mode count / N) increases over checkpoints and exceeds the threshold P before the *First* answer becomes explicit; bars are green since the mode is correct.

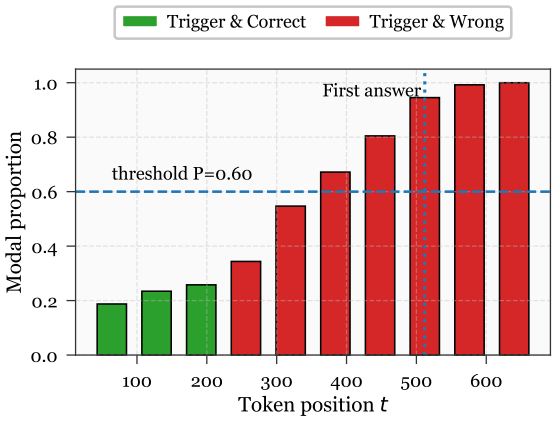


Figure 11: **Convergence (A, failure).** A single prompt probe can wrongly converge: the mode switches from the correct answer (green) to a stable wrong answer (red) while still surpassing P.

as the hidden state accumulates enough information, repeated probing becomes stable, and the dominant induced answer emerges *before* the model explicitly prints the first final answer. This justifies the core premise that early stopping can be driven by a *latent* answer signal rather than by waiting for explicit answer text. However, Figure 11 reveals a critical failure mode. The induced answer distribution can become sharply peaked around an *incorrect* option, leading to a high modal ratio that still passes P. Empirically, this manifests as an early green regime that later flips into a stable red mode. Table 18 quantifies the aggregate behavior: Baseline A triggers on 73.20% of instances (ESC), but still yields 0.88% wrong early stops (WESR), cor-

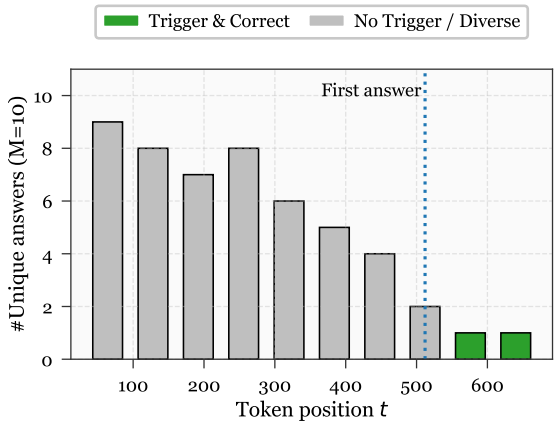


Figure 12: **Robustness (B, success).** With M diverse probe prompts and one sample per prompt (N=1), the number of unique induced answers decreases over time and eventually reaches 1 (green), indicating cross-prompt agreement.

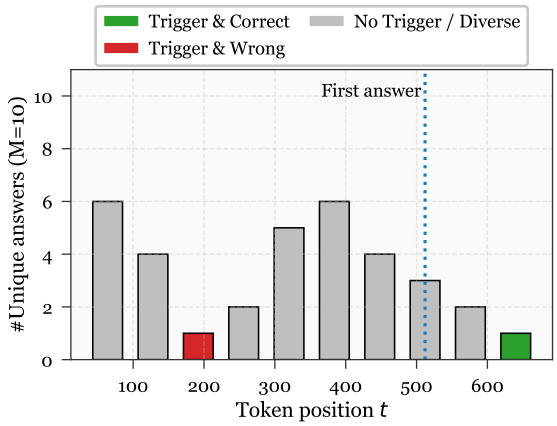


Figure 13: **Robustness (B, failure).** Cross prompt agreement with N=1 can be spurious: unique answers briefly drop to 1 on a wrong answer (red) due to high variance single draws.

responding to a conditional risk of 1.20% among triggered cases (Crisk@Agree). Therefore, convergence alone cannot guarantee safety, because it measures *stability* but not *correctness*.

Baseline B: cross prompt robustness reduces sensitivity, but single draws are noisy. Baseline B replaces repeated sampling under one prompt with a diversity check across prompts. At each checkpoint, we issue M semantically equivalent probe prompts, sample one answer from each prompt (N=1), and trigger if all prompts agree, that is, if the number of unique induced answers equals one. Figure 12 shows why this can work: as generation proceeds, prompt-induced variability shrinks,

Variant	ESC (%)	WESR (%)	Crisk@Agree (%)
B	70.50	0.92	1.30

Table 19: Baseline B.

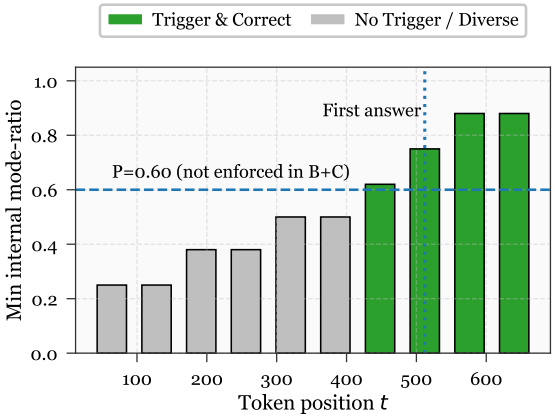


Figure 14: **Robustness+Parallelism (B+C, success).** Adding per-prompt parallel samples stabilizes prompt-wise modes; once modes align, the trigger is correct (green), and the minimum internal mode ratio rises.

and agreement eventually indicates a shared latent answer. The weakness is that N=1 makes each prompt level estimate extremely high variance. As shown in Figure 13, the unique answer count can transiently drop to one purely due to chance alignment, and the aligned answer may be wrong. Table 19 reflects this tradeoff: compared to A, B slightly reduces coverage (ESC) but does not reduce error risk, since spurious agreement events still occur. This motivates adding per-prompt parallelism to stabilize prompt-wise modes.

Baseline B+C: per prompt parallelism improves robustness, but still needs consistency control. Baseline B+C keeps the multi-prompt agreement idea of B, but adds component C by sampling N>1 answers per prompt and taking the prompt-wise mode. This converts the high-variance single draw in B into a lower variance estimate of each prompt-induced answer. Figure 14 shows that once prompt-wise modes align, the agreement is correct and the minimum internal modal ratio tends to rise. Nevertheless, agreement alone is still insufficient if prompt-wise modes align while the per-prompt internal modal ratios remain small. Figure 15 illustrates an early wrong agreement event that happens when internal ratios are far below a reasonable consistency threshold, even though prompt-wise modes coincide. Table 20 confirms the benefit and

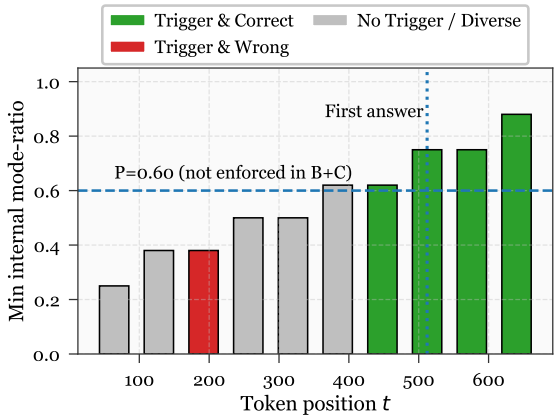


Figure 15: **Robustness+Parallelism (B+C, failure).** Without enforcing an internal threshold, prompt-wise modes may align too early when internal mode ratios are still low, producing a wrong trigger (red).

Variant	ESC (%)	WESR (%)	Crisk@Agree (%)
B+C	76.80	0.46	0.60

Table 20: Baseline B+C.

limitation: B+C increases trigger coverage, and roughly halves the wrong early stop rate relative to A and B, but nonzero risks remain without an explicit consistency gate.

Unified reverse ablation and metric definitions. To connect these observations, we evaluate all variants under a unified setting and report three early stop metrics. Let the evaluation set contain D problems. For problem i, let $S_i \in \{0,1\}$ indicate whether the method triggers early, and let $\mathbb{I}_i \in \{0,1\}$ indicate whether the triggered answer is wrong. We define

$$\begin{aligned} \text{ESC} &:= 100 \cdot \frac{1}{D}\sum_{i=1}^{D} S_i, \\ \text{WESR} &:= 100 \cdot \frac{1}{D}\sum_{i=1}^{D} S_i\,\mathbb{I}_i, \\ \text{Crisk@Agree} &:= 100 \cdot \frac{\sum_{i=1}^{D} S_i\,\mathbb{I}_i}{\sum_{i=1}^{D} S_i + \epsilon}, \end{aligned} \tag{26}$$

where ϵ is a tiny constant used only to avoid division by zero. By construction, Crisk@Agree is the conditional error probability given that a trigger happened, and it is consistent with the identity

$$\text{WESR} \approx \frac{\text{ESC} \cdot \text{Crisk@Agree}}{100}. \tag{27}$$

Variant	ESC (%)	WESR (%)	crisk@agr (%)
A/IC	73.20	0.88	1.20
B/CP1	70.50	0.92	1.30
B+C/CPN	76.80	0.46	0.60
A+B+C/DC	62.10	0.12	0.19
RED (ours)	66.70	0.04	0.06

Table 21: Reverse ablation results under a unified evaluation setting.

Abbreviations. IC: internal consistency (single probe prompt, N parallel samples, threshold P); CP1: cross prompt agreement with M prompts and one sample per prompt (N=1); CPN: cross prompt agreement with M prompts and N samples per prompt (no internal threshold); DC: dual consistency (IC + CPN); RED: DC + Refine First.

Takeaways from Table 21. Several consistent conclusions emerge. First, per-prompt parallelism (B+C) substantially reduces risk compared to A and B, confirming that most failures in B are due to single-draw variance. Second, adding the internal consistency gate on top of multi-prompt agreement (A+B+C, that is, DC) sharply reduces wrong early stops, but also reduces coverage because the joint trigger condition is stricter. Third, RED increases coverage relative to DC while further reducing risk, suggesting that refining the reasoning state improves the quality of the latent answer signal and makes convergence and agreement happen earlier and more reliably.

Theoretical analysis. We now provide a probabilistic account of why convergence, robustness, and parallelism interact as observed.

Answer distribution under probing. Fix a checkpoint t and probe prompt m. Let $\mathcal{A}$ be the set of possible extracted answers and let $\pi_t^{(m)}$ be the induced categorical distribution over $\mathcal{A}$ under the current hidden state. Denote the correct answer by $a^\star$ and define

$$\begin{aligned} p_t^{(m)} &:= \pi_t^{(m)}(a^\star), \\ q_{t,a}^{(m)} &:= \pi_t^{(m)}(a) \quad \text{for } a \neq a^\star. \end{aligned} \tag{28}$$

Sampling N parallel answers gives counts $C_{t,a}^{(m)}$ with $\sum_{a\in\mathcal{A}} C_{t,a}^{(m)} = N$ and empirical proportions

$$\widehat{\pi}_t^{(m)}(a) := \frac{C_{t,a}^{(m)}}{N}. \tag{29}$$

Let the empirical mode be defined as:

$$\widehat{a}_t^{(m)} := \arg\max_{a\in\mathcal{A}} \widehat{\pi}_t^{(m)}(a), \tag{30}$$

and let the corresponding modal ratio be

$$R_t^{(m)} := \max_{a \in \mathcal{A}} \widehat{\pi}_t^{(m)}(a) = \widehat{\pi}_t^{(m)}\left(\widehat{a}_t^{(m)}\right). \quad (31)$$

Baseline A as a hypothesis test. Baseline A triggers at the first checkpoint where $R_t^{(1)} \geq P$ under a single prompt. A wrong trigger at time t occurs when $\widehat{a}_t^{(1)} \neq a^\star$ and $R_t^{(1)} \geq P$. Using a union bound over wrong answers,

$$\Pr\left[\widehat{a}_t^{(1)} \neq a^\star \ \wedge \ R_t^{(1)} \geq P\right] \leq \sum_{a \in \mathcal{A} \setminus \{a^\star\}} \Pr\left[\widehat{\pi}_t^{(1)}(a) \geq P\right]. \quad (32)$$

For any fixed wrong answer a, $C_{t,a}^{(1)}$ is Binomial-like when the remaining mass is aggregated, and for $q_{t,a}^{(1)} < P$, a Chernoff bound yields

$$\Pr\left[\widehat{\pi}_t^{(1)}(a) \geq P\right] \leq \exp\Big(\text{-}N D\Big(P \,\|\, q_{t,a}^{(1)}\Big)\Big), \quad (33)$$

where the binary KL divergence is

$$D(u \,\|\, v) := u \log \frac{u}{v} + (1 \text{-} u) \log \frac{1 \text{-} u}{1 \text{-} v}. \quad (34)$$

This shows an exponential reduction in false triggers as N grows *when* all wrong answers have true probability below P. The failure in Figure 11 corresponds exactly to the regime where a wrong answer attains $q_{t,a}^{(1)} \geq P$, in which case no concentration argument can prevent a confident but wrong convergence.

Baseline B and spurious agreement. With M prompts and N=1, baseline B triggers when all single draws agree. Let $A_t^{(m,1)} \sim \pi_t^{(m)}$ be the sampled answer under prompt m. Then the agreement probability is

$$\Pr\left[A_t^{(1,1)} = \cdots = A_t^{(M,1)}\right] = \sum_{a \in \mathcal{A}} \prod_{m=1}^{M} \pi_t^{(m)}(a). \quad (35)$$

The wrong agreement probability is the same sum restricted to $a \neq a^\star$,

$$\Pr[\text{agree on wrong}] = \sum_{a \in \mathcal{A} \setminus \{a^\star\}} \prod_{m=1}^{M} q_{t,a}^{(m)}. \quad (36)$$

When prompt distributions are similar, $q_{t,a}^{(m)} \approx q_{t,a}$, the dominant wrong term scales as $q_{t,a}^{M}$, which decreases with M. Yet, Figure 13 arises because single draws have maximal variance: even if $p_t^{(m)}$ is only slightly larger than a competing wrong probability, a one-sample decision can easily flip, and occasional chance alignment is unavoidable.

Effect of per prompt parallelism. In baseline B+C, each prompt uses N samples and outputs the prompt-wise mode $\widehat{a}_t^{(m)}$. A minimal but informative approximation is the two answer competition between the correct answer and the strongest wrong contender. Let $p_t^{(m)}$ be the correct probability and let $r_t^{(m)}$ be the probability of the strongest wrong contender, with $p_t^{(m)} + r_t^{(m)} \leq 1$. Let $\tau = \lfloor N/2 \rfloor + 1$ denote the majority threshold. If we ignore the remaining mass for clarity, the probability that majority voting selects the correct answer is:

$$\Pr\left[\widehat{a}_t^{(m)} = a^\star\right] = \sum_{k=\tau}^{N} \binom{N}{k} \left(p_t^{(m)}\right)^k \left(r_t^{(m)}\right)^{N-k}. \quad (37)$$

For any fixed margin $p_t^{(m)}$ - $r_t^{(m)} > 0$, the above tail probability increases rapidly with N, which explains the reduction in spurious prompt-wise modes once component C is added.

Why dual consistency is safer. Dual consistency triggers only when (i) prompt-wise modes agree and (ii) the minimum internal modal ratio satisfies $\min_m R_t^{(m)} \geq P$. For a wrong answer a, define the per-prompt event

$$E_{t,a}^{(m)} := \left\{\widehat{a}_t^{(m)} = a \ \wedge \ R_t^{(m)} \geq P\right\}. \quad (38)$$

A wrong dual consistency trigger implies $\bigcap_{m=1}^{M} E_{t,a}^{(m)}$ for some $a \neq a^\star$, hence Let E_{err} denote the event that the dual consistency mechanism triggers on an incorrect answer. This implies that for some wrong answer $a \neq a^\star$, the consistency condition is met across all M prompts. By the union bound, we have:

$$\Pr[E_{\text{err}}] \leq \sum_{a \in \mathcal{A} \setminus \{a^\star\}} \prod_{m=1}^{M} \Pr[E_{t,a}^{(m)}]. \quad (39)$$

Now consider the individual failure probability for a specific prompt m. If the true probability $q_{t,a}^{(m)} < P$, the event $E_{t,a}^{(m)}$ implies that the empirical frequency exceeds the threshold P. By applying the Chernoff bound, we derive:

$$\Pr[E_{t,a}^{(m)}] \leq \Pr\left[\widehat{\pi}_t^{(m)}(a) \geq P\right] \leq \exp\Big(-N\, D\big(P \,\|\, q_{t,a}^{(m)}\big)\Big). \quad (40)$$

Substituting Eq. (40) back into Eq. (39) yields the final compact bound:

$$\Pr[E_{\text{err}}] \leq \sum_{a \in \mathcal{A} \setminus \{a^\star\}} \exp\Big(-N \sum_{m=1}^{M} D\big(P \,\|\, q_{t,a}^{(m)}\big)\Big). \quad (41)$$

This explains why DC and RED achieve substantially lower WESR and Crisk@Agree in Table 21: the probability of a confident, prompt invariant wrong answer decays exponentially in both N and M once the internal threshold is enforced.

Coverage as a hitting time. The price is reduced coverage. Let $T_{\max}$ be the maximum allowed decoding length, and define the first trigger time Let $\mathcal{C}_t$ denote the event that the dual-consistency trigger condition is satisfied at decoding step t. We define the early-exit step T as the first hit time:

$$T := \inf\Big\{t \le T_{\max} : \mathbb{1}_{\mathcal{C}_t} = 1\Big\}, \qquad (42)$$

where $\mathbb{1}$ is the indicator function. If the condition is never met, we set $T = T_{\max}$. Then the early stop coverage can be written as

$$\mathrm{ESC} = 100 \cdot \Pr[T < T_{\max}], \qquad (43)$$

and the conditional risk is

$$\mathrm{Crisk@Agree} = 100 \cdot \Pr\Big[\widehat{Y} \neq Y \,\Big|\, T < T_{\max}\Big], \qquad (44)$$

where Y is the ground truth and $\widehat{Y}$ is the triggered answer. Increasing P, M, or N generally decreases the risk bounds above but also increases T in expectation, thereby lowering ESC. This formalizes the empirical tradeoff observed between B+C and DC.

Why refining the reasoning state helps. Finally, the role of Refine First can be modeled as improving the entire family of prompt-induced distributions by shifting probability mass toward $a^\star$ earlier. One simple parameterization is a monotone growth model

$$p_t^{(m)} = \sigma\big(\alpha_m(t - \tau_m)\big), \qquad \sigma(x) := \frac{1}{1 + e^{-x}}, \qquad (45)$$

where τ_m is the prompt specific time when the latent answer becomes confident. A successful refinement reduces these effective times, that is, $\tau_m \mapsto \tau_m - \Delta$ with $\Delta > 0$, which decreases the expected hitting time T while leaving the concentration bounds for wrong triggers unchanged or improved. This provides a mechanistic explanation for why RED simultaneously improves ESC and reduces WESR relative to DC in Table 21.

J Analysis of Latency Overhead

To rigorously assess the latency overhead introduced by our RED (comprising both the entropy-based intervention in Section 4.2 and the probe-based early exit in Section 4.3), we conducted comprehensive latency profiling across all evaluation benchmarks. All experiments were executed on a high-performance computing cluster equipped with NVIDIA A100-80G accelerators.

Crucially, to isolate the raw latency cost of these additional operations from the acceleration benefits of early exiting, we implemented a **"non-stopping" stress test**.

In this configuration, the *Refining* module remains fully active, performing real-time entropy variance monitoring and triggering negative sampling interventions when thresholds are breached. Simultaneously, the *Discarding* module executes the parallel probing and consistency checks at every interval K. However, we intentionally disable the final termination trigger even when the *dual-consistency* condition is met.

This forces the model to traverse the full generation trajectory while bearing the maximum cumulative computational burden of both monitoring computations and periodic probing. We recorded the *total wall-clock time* required to complete inference over the **entire test set** (e.g., the full validation splits of GSM8K, MATH, and GPQA-Diamond) to quantify this worst-case latency overhead.

Table 22 presents the comparison between the vanilla decoding baseline and our method. The results indicate that under the **"non-stopping" stress test setting**—where the model is forced to execute all monitoring and probing operations without early exiting—we observe an **Average Relative Latency Overhead** of approximately **4.6%** across all evaluated models and benchmarks. This confirms that the intrinsic computational cost of our dual-mechanism framework is marginal, primarily attributed to the parallel design of the probe which minimizes interference with the main decoding pipeline. Crucially, in practical deployment scenarios where the early-exit mechanism is active, this minimal overhead is easily amortized by the substantial reduction in generation steps (often saving >50% of tokens as shown in Table 3). Consequently, the additional latency becomes negligible, and the system achieves a net gain in overall inference efficiency.

Table 23 presents the real-world inference runtime, excluding AlphaOne and GRPO due to their prohibitive computational costs. While RED introduces a minor operational overhead (∼4.6%), the results demonstrate that this cost is negligible compared to the massive acceleration achieved through adaptive termination, consistently yielding the lowest or near-lowest total wall-clock time

Model	Method	AIME24	AIME25	MATH500	GSM8K	GPQA-Diamond
Qwen3-8B-Thinking	Vanilla	02:26	02:38	06:24	05:47	04:12
	RED (raw)	02:27	02:38	06:34	06:03	04:20
Qwen3-32B-Thinking	Vanilla	04:25	04:34	10:04	08:09	06:21
	RED (raw)	04:35	04:49	10:27	08:26	06:38
DeepSeek-R1-Qwen-7B	Vanilla	02:13	02:12	04:11	02:26	04:52
	RED (raw)	02:24	02:21	04:23	02:38	05:15
DeepSeek-R1-Qwen-32B	Vanilla	04:00	04:10	07:06	01:45	07:29
	RED (raw)	04:12	04:20	07:20	01:51	07:42
DeepSeek-R1-Llama-8B	Vanilla	02:10	02:06	04:49	02:38	04:59
	RED (raw)	02:16	02:28	05:09	02:43	05:10
DeepSeek-R1-Llama-70B	Vanilla	06:41	06:37	10:47	03:42	10:25
	RED (raw)	06:57	06:50	11:01	03:49	10:37

Table 22: Total inference runtime (mm:ss) comparison across benchmarks. **Vanilla** represents the baseline runtime, while **RED (raw)** denotes our method under stress test, where early-exit is disabled to isolate the operational overhead of probing and interventions

Model	Method	AIME24	AIME25	MATH500	GSM8K	GPQA-Diamond
Qwen3-8B-Thinking	Vanilla	02:26	02:38	06:24	05:47	04:12
	Think or Not	02:14	02:40	06:32	04:11	04:25
	DAST	01:35	01:28	03:50	02:08	02:19
	RL +length penalty	01:48	01:56	04:57	03:38	02:21
	S-GRPO	01:44	01:46	04:23	03:22	02:18
	RED (ours)	01:09	01:16	03:02	01:49	02:13
Qwen3-32B-Thinking	Vanilla	04:25	04:34	10:04	08:09	06:21
	Think or Not	04:51	05:04	09:39	06:55	03:02
	DAST	02:56	02:57	03:35	04:20	04:06
	RL +length penalty	02:57	03:01	07:09	06:12	06:21
	S-GRPO	03:39	03:30	06:34	06:08	06:00
	RED (ours)	02:27	02:26	06:16	02:43	03:46
DeepSeek-R1-Qwen-7B	Vanilla	02:13	02:12	04:11	02:26	04:52
	Think or Not	01:48	01:51	03:02	02:44	04:31
	DAST	01:42	01:59	02:01	01:15	02:15
	RL +length penalty	01:03	01:30	01:58	01:33	01:42
	S-GRPO	00:55	01:05	01:42	01:21	01:25
	RED (ours)	00:58	01:19	01:48	01:39	02:45
DeepSeek-R1-Qwen-32B	Vanilla	04:00	04:10	07:06	01:45	07:29
	Think or Not	04:01	04:45	08:09	01:40	08:45
	DAST	02:28	03:07	04:18	00:59	05:27
	RL +length penalty	02:12	02:48	05:11	01:03	05:56
	S-GRPO	01:41	01:47	06:31	01:13	05:03
	RED (ours)	02:01	02:28	05:43	00:53	04:31
DeepSeek-R1-Llama-8B	Vanilla	02:10	02:06	04:49	02:38	04:59
	Think or Not	02:46	02:39	04:53	02:07	04:02
	DAST	02:12	02:09	04:15	02:24	02:33
	RL +length penalty	01:18	01:23	04:17	02:06	01:57
	S-GRPO	00:59	01:08	03:36	02:05	02:29
	RED (ours)	01:15	01:03	02:57	02:05	02:08
DeepSeek-R1-Llama-70B	Vanilla	06:41	06:37	10:47	03:42	10:25
	Think or Not	06:33	06:14	11:27	03:55	11:56
	DAST	05:36	05:04	08:26	02:37	08:19
	RL +length penalty	05:48	05:11	04:20	02:27	08:50
	S-GRPO	05:22	03:24	06:38	02:27	08:28
	RED (ours)	03:39	04:56	05:59	02:25	06:32

Table 23: Total inference runtime (mm:ss) comparison across baselines. The experiments enable the early-exit mechanism.**Vanilla** represents the baseline runtime, while **RED (ours)** denotes our method.

across benchmarks.

Specifically, across all evaluated reasoning models and benchmarks, `RED` consistently achieves the lowest or near-lowest total wall-clock time. For instance, on the DeepSeek-R1-Llama-70B model, our method reduces the inference time for the AIME24 benchmark from 06:41 to 03:39—a speedup of nearly $1.8\times$. This confirms that the substantial reduction in generated tokens effectively "amortizes" the per-step probing and intervention costs. Ultimately, the experimental data suggest that the minor latency introduced by intermediate probing is a highly efficient cost, yielding significant net savings in total wall-clock time.

K Theoretical Analysis:FoE Makes the First Solution the Best

This section provides an information-theoretic and probabilistic account of why *extending* reasoning (thus producing subsequent solutions `Subs`) can increase the probability of error rather than decrease it. The key mechanism is that errors in a reasoning trace are not isolated: once a wrong artifact enters the context, it can be repeatedly reused and amplified, forming a *forest* of causally linked error nodes whose growth budget increases with test-time.

K.1 FoE mechanism (informal, non-asymptotic).

We represent the internal error structure of a solution trace as a *directed forest* of wrong artifacts. **FoE is a causal forest of errors:** each error node has at most one parent, defined as the nearest earlier error that *directly induces* it via explicit artifact reuse (wrong values, wrong assumptions, wrong bindings, wrong rules). Because new root errors can arise over time, the resulting structure is a forest rather than a single chain. **Root dominance and strong parent–child dependence:** fixing a descendant without fixing its ancestor typically fails to stop the creation of new errors because the uncorrected ancestor remains in-context as a persistent "error factory"; in contrast, correcting the root of a formed tree strongly suppresses subsequent growth in both depth and breadth. Empirically, `First` **has a smaller and slower-growing FoE than** `Subs` under both static forest metrics (number of trees, nodes per tree, depth) and a dynamic reproduction-rate metric. Moreover, **root-error genesis correlates with the *joint* elevation of uncertainty and volatility:** neither a single uncertainty statistic (entropy) nor a single volatility statistic (entropy variance) suffices alone, while their joint elevation is most predictive for new root errors. Finally, **self-reflection does not reliably prune FoE (and degrades in `Subs`):** later "corrections" are often refusals or "fake corrections" that edit surface text but preserve the underlying wrong artifact, hence additional exploration tends to preserve and amplify wrong artifacts rather than remove them. Consistently, **latent instability is higher in `Subs` even when the final answer looks correct:** perturbing the generation by resampling from intermediate states causes `Subs` to drift to incorrect answers more easily, indicating weaker "lock-in" toward the correct solution trajectory.

K.2 A probabilistic FoE model.

Let Y be the ground-truth answer for question Q. A reasoning model generates a sequence of tokens (or steps) forming a solution trace S, and outputs an extracted final answer $\hat{Y}(S)$. We represent the error structure inside a trace S by a directed forest

$$\mathcal{F}(S) = (\mathcal{V}(S), \mathcal{E}(S)), \tag{46}$$

where $\mathcal{V}(S)$ is the set of identified error nodes (artifacts) and $\mathcal{E}(S)$ links each error to its nearest inducing predecessor (or marks it as a root).

K.2.1 Root-error triggering is driven by entropy and entropy-variance.

Let $p_t(\cdot)$ denote the token distribution at generation step t, and define step entropy

$$H_t \;=\; -\sum_x p_t(x) \log p_t(x). \tag{47}$$

Over a local window W_t, define the mean entropy and entropy variance

$$\begin{aligned} h_t \;&=\; \frac{1}{|W_t|} \sum_{i \in W_t} H_i, \\ v_t \;&=\; \frac{1}{|W_t| - 1} \sum_{i \in W_t} (H_i - h_t)^2. \end{aligned} \tag{48}$$

Let R_t be the indicator that a *new root error* is created at step t. The FoE findings imply that root triggering is *supermodular* in (h_t, v_t): joint elevation is more dangerous than either alone. A convenient statistical abstraction is a monotone interaction model The probability is modeled as:

$$\Pr(R_t = 1 \mid Q, S_t) \le \sigma\big(\eta_t\big), \tag{49}$$

where S_t denotes the state at time t, and η_t represents the logit term defined as:

$$\eta_t = \beta_0 + \beta_h h_t + \beta_v v_t + \beta_{hv} h_t v_t. \tag{50}$$

where $\sigma(\cdot)$ is a sigmoid link and $\beta_{hv} > 0$ captures the empirically observed synergy between entropy and entropy-variance in producing *root* errors.

K.2.2 FoE growth as a branching process (root dominance formalized).

Once a root error is created, it can induce descendant errors through artifact reuse. We approximate each root-initiated tree as a truncated Galton–Watson process with solution-dependent mean reproduction ρ (capturing the FoE reproduction metric). Let $G(\ell; \rho)$ be the expected total number of nodes produced by a single root when ℓ steps remain:

$$G(\ell; \rho) \;=\; \sum_{d=0}^{\ell} \rho^d. \tag{51}$$

Let T be the length (in steps) of the trace, and define the expected total number of error nodes

$$\mathbb{E}\big[|\mathcal{V}(S)|\big] \;\le\; \sum_{t=1}^{T} \Pr(R_t = 1)\, G(T-t; \rho). \tag{52}$$

This expression isolates the **root dominance** mechanism: decreasing $\Pr(R_t = 1)$ (fewer roots) or decreasing ρ (slower reproduction) suppresses *the entire* downstream error forest.

K.2.3 From FoE size to answer error: a general bound.

The extracted answer $\hat{Y}(S)$ is wrong if at least one *decision-critical* error influences the final decision. Associate each error node $v \in \mathcal{V}(S)$ with an event C_v indicating that v becomes decision-critical (directly or through descendants) for the final answer. Assume a bounded per-node criticality probability:

$$\Pr(C_v = 1 \mid v \in \mathcal{V}(S)) \;\le\; \kappa, \tag{53}$$

for some task/model-dependent constant κ.

By Boole's inequality (union bound), the error probability is bounded by the total critical risk across all nodes in $\mathcal{V}(S)$. Conditioning on the random error set and taking the expectation, we have:

$$\begin{aligned} \Pr\big(\hat{Y}(S) \neq Y\big) &= \Pr\Big(\bigcup_{v \in \mathcal{V}(S)} C_v\Big) \\ &\le \mathbb{E}\Big[\sum_{v \in \mathcal{V}(S)} \Pr(C_v = 1 \mid v)\Big]. \end{aligned} \tag{54}$$

Recalling the bounded per-node criticality $\Pr(C_v = 1) \le \kappa$ from (53), this simplifies to:

$$\Pr\big(\hat{Y}(S) \neq Y\big) \le \kappa\, \mathbb{E}\big[|\mathcal{V}(S)|\big]. \tag{55}$$

Let $\mathcal{M}(S) := \mathbb{E}[|\mathcal{V}(S)|]$ denote the expected size of the error forest. By substituting the branching process result (52) into (55), we obtain the explicit FoE-to-error bound:

$$\Pr(\hat{Y}(S) \neq Y) \le \Phi(T, \mathbf{P}_R, \rho) := \kappa \cdot \mathcal{M}(S), \tag{56}$$

where the expected forest size $\mathcal{M}(S)$ is expanded as:

$$\mathcal{M}(S) = \sum_{t=1}^{T} \Pr(R_t = 1) G(T-t; \rho). \tag{57}$$

Here, $\mathbf{P}_R := \{\Pr(R_t = 1)\}_{t=1}^{T}$ denotes the root-triggering profile.

K.2.4 Bounds for `First` and `Subs` (why `The First is The Best`).

Let $S_{\texttt{First}}$ denote the `First` solution trace, and $S_{\texttt{Subs}}$ denote a representative subsequent solution trace. Define the FoE-based upper bounds for the first and subsequent traces as:

$$\begin{aligned} \Phi_{\texttt{First}} &:= \Phi(T_{\texttt{First}}, \mathbf{P}_R^{\texttt{First}}, \rho_{\texttt{First}}), \\ \Phi_{\texttt{Subs}} &:= \Phi(T_{\texttt{Subs}}, \mathbf{P}_R^{\texttt{Subs}}, \rho_{\texttt{Subs}}), \end{aligned} \tag{58}$$

where $\mathbf{P}_R^{\texttt{First}}$ and $\mathbf{P}_R^{\texttt{Subs}}$ denote the respective root-triggering profiles $\{\Pr(R_t^{(\cdot)} = 1)\}_{t=1}^{T_{(\cdot)}}$.

The FoE findings imply the following structural dominance relations (holding in expectation, and often pointwise along the trace):

$$\begin{aligned} \Pr(R_t^{\texttt{First}} = 1) \;&\le\; \Pr(R_t^{\texttt{Subs}} = 1), \\ \rho_{\texttt{First}} \;&\le\; \rho_{\texttt{Subs}}, \\ T_{\texttt{First}} \;&\le\; T_{\texttt{Subs}}. \end{aligned} \tag{59}$$

Since $G(\ell; \rho) = \sum_{d=0}^{\ell} \rho^d$ is nondecreasing in both ℓ and ρ, the bound function $\Phi(T, \mathbf{P}_R, \rho)$ is monotone in (i) the horizon T, (ii) the componentwise root-trigger profile $\mathbf{P}_R$, and (iii) the reproduction rate ρ. Therefore, (59) implies

$$\Phi_{\texttt{First}} \;\le\; \Phi_{\texttt{Subs}}. \tag{60}$$

Consequently, the error probabilities satisfy the following *non-asymptotic* upper bounds:

$$\begin{aligned} \Pr\big(\hat{Y}(S_{\texttt{First}}) \neq Y\big) &\le \Phi_{\texttt{First}} \le \Phi_{\texttt{Subs}}, \\ \Pr\big(\hat{Y}(S_{\texttt{Subs}}) \neq Y\big) &\le \Phi_{\texttt{Subs}}. \end{aligned} \tag{61}$$

These relations indicate that **Subs** operates in a regime of higher *informational stress* (higher $\mathbf{P}_R$) with a weaker *error-pruning* capability (higher ρ), compounded by a longer accumulation horizon (T), which yields a strictly looser FoE-based risk upper bound.

K.2.5 A bound for "Subs biases First".

We formalize the event that subsequent reasoning segments override a correct first solution ("**Subs** biases **First** ") as the event that the first solution is correct, but the final decision after continued exploration is incorrect. Let $\hat{Y}_{\text{final}}$ denote the extracted answer after the model continues reasoning beyond the first solution. Define the misguidance event

$$\mathcal{B} := \{\hat{Y}(S_{\texttt{First}}) = Y \wedge \hat{Y}_{\text{final}} \neq Y\}. \quad (62)$$

Information contamination (artifact reuse across segments). The continuation is generated under the *same* context as $S_{\texttt{First}}$, so its error dynamics are conditioned on the full history $\mathcal{H}_{\texttt{First}}$ of the first trace. In particular, attention-based artifact reuse can "contaminate" the continuation: previously introduced (possibly non-decision-critical) artifacts can be revisited and amplified, effectively increasing root-triggering propensity and/or reproduction.

Let $\mathcal{T}_{\text{ext}} := \{T_{\texttt{First}} + 1, \ldots, T_{\text{final}}\}$ denote the time indices of the continuation segment. Conditioning on $\mathcal{H}_{\texttt{First}}$, we can upper-bound the final-error probability by reusing the FoE bound on the continuation:

To simplify, let $\mathcal{M}_{\text{ext}}(\mathcal{H}_{\texttt{First}})$ denote the expected size of the error forest generated during the continuation, conditioned on the first trace's history $\mathcal{H}_{\texttt{First}}$:

$$\mathcal{M}_{\text{ext}} = \sum_{t \in \mathcal{T}_{\text{ext}}} \Pr(R_t^{\text{ext}} = 1 \mid \mathcal{H}_{\texttt{First}}) \cdot G(T_{\text{final}} - t; \rho_{\text{ext}}). \quad (63)$$

Conditioning on $\mathcal{H}_{\texttt{First}}$, the error probability of the final decision can be upper-bounded by:

$$\Pr(\hat{Y}_{\text{final}} \neq Y \mid \mathcal{H}_{\texttt{First}}) \leq \kappa \cdot \mathcal{M}_{\text{ext}}(\mathcal{H}_{\texttt{First}}). \quad (64)$$

Using a pessimistic domination that captures degraded reflection in later segments, we bound the conditional error probability by its **Subs**-regime parameters. Let $\Phi_{\text{ext}} := \kappa \cdot \mathcal{M}_{\text{ext}}^{\texttt{Subs}}$ denote the error bound for the continuation segment, where:

$$\Phi_{\text{ext}} = \kappa \sum_{t \in \mathcal{T}_{\text{ext}}} \Pr(R_t^{\texttt{Subs}} = 1)\, G(T_{\text{final}} - t; \rho_{\texttt{Subs}}). \quad (65)$$

Then, the probability that the final decision is incorrect given a correct first solution is bounded by:

$$\Pr(\hat{Y}_{\text{final}} \neq Y \mid \hat{Y}(S_{\texttt{First}}) = Y) \leq \Phi_{\text{ext}}. \quad (66)$$

Therefore, The probability of the misguidance event $\mathcal{B}$ then simplifies to the product of the first solution's success probability and the continuation's error bound:

$$\Pr(\mathcal{B}) \leq \Pr(\hat{Y}(S_{\texttt{First}}) = Y) \cdot \Phi_{\text{ext}}. \quad (67)$$

K.3 Takeaway.

Equations (60) and (61) formalize a FoE-based risk characterization: when root errors are more likely under elevated entropy–variance and amplified via higher reproduction over a longer horizon, the resulting error upper bound is tighter for **First** than for **Subs**. In other words, **Subs** operates under higher *informational stress* and weaker *error-pruning*, yielding a looser non-asymptotic risk bound.

Equation (67) further shows that generating **Subs** is not merely "extra compute": it opens an additional failure channel whereby continued exploration overrides a correct **First** solution, primarily through artifact reuse that introduces new root errors.

Together, these results support the theoretical statement that extending reasoning beyond the first solution is not guaranteed to improve reliability and can strictly worsen it—hence, *FoE makes the first solution the best*.

L Few-Shot Prompt of Parent-Children Score(PCS) Judging

Prompts are as shown as below.

FEW-SHOT PROMPT OF PCS JUDGING

```
# PCS (Parent-Children Score) -
    Few-shot Prompt for FoE Causal
    Scoring

## Role
You are a **forensic causal
    annotator** for a *Forest of
    Errors (FoE)* reasoning trace.

## Objective
Given an earlier error node **e_i
    ** (candidate parent) and a
    later error node **e_j** (child
    ), plus the full reasoning
    prefix up to and including **
    e_j**, output a **PCS score**
```

```
    measuring whether **e_i
    directly induces e_j**.

You are scoring **direct causal
    induction** (error propagation)
    , not topical similarity.

---

## Inputs
You will receive:

### CONTEXT_UP_TO_CHILD
The full reasoning prefix up to
    and including the child node **
    e_j**.

### CANDIDATE_PARENT_NODE (earlier
    )
The text of error node **e_i** (
    chronologically earlier than
    e_j).

### CHILD_NODE (later)
The text of error node **e_j**.

---

## Output (STRICT)
Output **exactly one line**
    containing **one number**:
- range: **1.0 to 5.0** (inclusive
    )
- format: **exactly one decimal
    place**
- regex: `^(?:[1-4]\.[0-9]|5\.0)$`

**Do not output anything else**:
    no explanation, no labels, no
    JSON, no punctuation, no extra
    whitespace.

---

## Core definitions (do not
    reinterpret)

### Wrong artifact
A "wrong artifact" introduced by
    **e_i** can be any incorrect
    item that later gets reused,
    including:
- wrong numeric value /
    denominator / count / derived
    quantity,
- wrong formula choice (e.g.,
    using circumference for area),
- wrong theorem or rule
    application (e.g., inclusion-
    exclusion used with a wrong
    intersection term),
- wrong constraint interpretation
    or assumption ("disjoint", "
    independent", "without
    replacement", etc.),
- wrong definition / variable
    binding ("Let T=...", wrong
    meaning of a symbol),
- wrong intermediate statement (
    equation, inequality,
    recurrence, invariant, case
    split rule, etc.),
- wrong algorithmic step / update
    rule.

### Direct parent (direct inducer)
**e_i is a direct parent of e_j**
    iff BOTH hold:
1) **Dependency**: e_j's wrongness
     depends on at least one wrong
    artifact introduced by e_i; AND
2) **Directness**: within the
    provided prefix, there is **no
    closer, more-specific error
    step** that better explains the
     particular artifact reuse in
    e_j.

---

## High-score propagation patterns
High PCS scores are appropriate
    for **any strong artifact flow
    **, including:
- **numeric propagation** (wrong
    value reused downstream),
- **formula propagation** (wrong
    formula chosen, then used
    downstream),
- **theorem/rule propagation** (
    wrong theorem usage yields a
    derived equation/constraint,
    then reused),
- **assumption propagation** ("
    independent", "disjoint", "
    monotone", etc.),
- **definition/binding propagation
    ** (a symbol is bound wrong and
     reused).

---

## Threshold safety rule (keep it
    conservative)
Because downstream will treat **
    PCS >= 4.0** as "connect an
    edge", be conservative.

You may output **PCS >= 4.0** only
     when the context provides **
    concrete evidence** that:
- the child reuses the parent's
    wrong artifact (same value/
    symbol/assumption/derived
    equation), AND
- there is no nearer step in the
    prefix that more directly
    explains the reused artifact.

If these are not satisfied, output
     **PCS <= 3.9**.

Tie-break (critical):
- If you are torn between **3.9
    and 4.0**, output **3.9**.
```

```
---

# Anchor meanings (1.0 / 2.0 / 3.0
    / 4.0 / 5.0) - detailed

## 5.0 - Certain direct parent (
    near-certain direct induction)
Use 5.0 only if:
- the artifact flow is **explicit
    ** and **central** (the child's
     error is overwhelmingly a
    consequence of reusing the
    parent's artifact), AND
- fixing e_i would **almost
    certainly** remove or
    materially change e_j, AND
- there is no plausible competing
    parent.

Typical 5.0 cases:
- child directly computes from a
    wrong number produced by parent
    ,
- child directly computes from a
    wrong formula/theorem result
    produced by parent,
- child is essentially the "
    application step" of the parent
    's wrong conclusion.

Disqualifier:
- if the child has a substantial
    independent error that would
    remain even if e_i were
    corrected, prefer 4.0+.

## 4.0 - Likely direct parent (
    threshold)
Use 4.0 when:
- there is concrete artifact reuse
     evidence, AND
- e_i is the most plausible direct
     inducer, BUT
- there is non-trivial uncertainty
     (e.g., child also contains an
    additional independent mistake,
     or the dependency is not
    maximally explicit).

Typical 4.0 cases:
- child reuses the artifact, but
    also introduces another
    separate error,
- dependency is clear but not "
    maximally explicit",
- mild competing-cause ambiguity
    exists, but e_i is still the
    best direct cause.

## 3.0 - Meaningfully related, but
     probably not direct
Use 3.0 when there is **meaningful
     relatedness**, yet direct
    parenthood is not established,
    such as:
- e_i is a plausible **ancestor**
    but a closer mediator exists,
- the dependency seems plausible
    but lacks concrete signals in
    the prefix,
- the child's wrongness would
    likely persist even if e_i were
     corrected.

3.0 means: "There is a real
    relationship, but not a direct
    parent edge."

## 2.0 - Almost unrelated; only
    weak, surface relatedness
Use 2.0 when:
- the nodes are in the same
    general topic/segment or share
    variable names, BUT
- there is **no credible artifact
    flow**, and any causal story
    would be speculative.

2.0 is **near-unrelated** but not
    fully: there is slight overlap
    (same symbols/topic/proximity),
     yet no dependency.

## 1.0 - Unrelated
Use 1.0 when:
- there is no meaningful link at
    all,
- different subgoals/branches/
    domains,
- no shared entities or any shared
     tokens are clearly from the
    original problem statement
    rather than reuse.

---

# One-decimal scoring rule
Output **exactly one number** in
    **[1.0, 5.0]** with **exactly
    one decimal place**.
Choose the score based on the
    anchor meanings and the
    strength of direct artifact
    flow.
No other text.

---

# Few-shot examples (anchor points
    )
IMPORTANT: Each example's output
    is **only the score**, one line
    .

## Anchor = 5.0 (3 examples)

### Example 5.0-A (simple numeric
    propagation)
**CONTEXT_UP_TO_CHILD**
[01] There are 6 chapters with 18
    pages each.
[02] Total pages = 6 * 18 = 96.
[03] Printing costs $0.10 per page,
     so total cost = 96 * 0.10 = $9
    .60.
```

```
**CANDIDATE_PARENT_NODE**
[02] Total pages = 6 * 18 = 96.

**CHILD_NODE**
[03] Printing costs $0.10 per page,
     so total cost = 96 * 0.10 = $9
    .60.

**OUTPUT**
5.0

### Example 5.0-B (simple formula
    propagation)
**CONTEXT_UP_TO_CHILD**
[01] Radius r = 4. We need the
    area of the circle.
[02] Area = 2 * pi * r = 8 * pi.
[03] Therefore area ~= 8 * pi ~=
    25.13.

**CANDIDATE_PARENT_NODE**
[02] Area = 2 * pi * r = 8 * pi.

**CHILD_NODE**
[03] Therefore area ~= 8 * pi ~=
    25.13.

**OUTPUT**
5.0

### Example 5.0-C (complex
    propagation: misjudgment ->
    wrong theorem use -> derived
    conclusion)
**CONTEXT_UP_TO_CHILD**
[01] Let A be numbers in {1..100}
    divisible by 2, and B be
    numbers divisible by 5.
[02] |A| = 50 and |B| = 20.
[03] Since 2 and 5 are coprime,
    assume |A \cap B| = 0.
[04] By inclusion-exclusion, |A \
    cup B| = |A| + |B| - |A \cap B|
     = 50 + 20 - 0 = 70.

**CANDIDATE_PARENT_NODE**
[03] Since 2 and 5 are coprime,
    assume |A \cap B| = 0.

**CHILD_NODE**
[04] By inclusion-exclusion, |A \
    cup B| = |A| + |B| - |A \cap B|
     = 50 + 20 - 0 = 70.

**OUTPUT**
5.0

---

## Anchor = 4.0 (3 examples)

### Example 4.0-A (direct
    dependency, but child also has
    an extra mistake)
**CONTEXT_UP_TO_CHILD**
[01] A box has 9 blue marbles and
    5 green marbles.
[02] Total marbles = 9 + 5 = 12.
[03] Probability(green) = 5/12 ~=
    0.45.

**CANDIDATE_PARENT_NODE**
[02] Total marbles = 9 + 5 = 12.

**CHILD_NODE**
[03] Probability(green) = 5/12 ~=
    0.45.

**OUTPUT**
4.0

### Example 4.0-B (complex
    propagation: wrong theorem form
     reused; child has extra
    arithmetic error)
**CONTEXT_UP_TO_CHILD**
[01] We want |A \cup B \cup C|.
    Given |A|=30, |B|=25, |C|=20, |
    A \cap B|=10, |A \cap C|=5, |B
    \cap C|=4, and |A \cap B \cap C
    |=3.
[02] Use inclusion-exclusion: |A \
    cup B \cup C| = |A|+|B|+|C| - |
    A \cap B| - |A \cap C| - |B \
    cap C|.
[03] Plug in: 30+25+20 = 70, so |A
     \cup B \cup C| = 70 - 10 - 5 -
     4 = 51.

**CANDIDATE_PARENT_NODE**
[02] Use inclusion-exclusion: |A \
    cup B \cup C| = |A|+|B|+|C| - |
    A \cap B| - |A \cap C| - |B \
    cap C|.

**CHILD_NODE**
[03] Plug in: 30+25+20 = 70, so |A
     \cup B \cup C| = 70 - 10 - 5 -
     4 = 51.

**OUTPUT**
4.0

### Example 4.0-C (simple formula
    propagation, plus child
    introduces an extra arithmetic
    mistake)
**CONTEXT_UP_TO_CHILD**
[01] Radius r = 4. We need the
    area of the circle.
[02] Area = 2 * pi * r = 8 * pi.
[03] Therefore area ~= 8 * 3.14 =
    23.12.

**CANDIDATE_PARENT_NODE**
[02] Area = 2 * pi * r = 8 * pi.

**CHILD_NODE**
[03] Therefore area ~= 8 * 3.14 =
    23.12.

**OUTPUT**
4.0

---
```

```
## Anchor = 3.0 (3 examples)

### Example 3.0-A (ancestor: a
    nearer mediator is more direct)
**CONTEXT_UP_TO_CHILD**
[01] A box has 9 blue marbles and
    5 green marbles.
[02] Total marbles = 9 + 5 = 12.
[03] Let T = 12 be the total
    number of marbles.
[04] Probability(green) = 5/T =
    5/12.

**CANDIDATE_PARENT_NODE**
[02] Total marbles = 9 + 5 = 12.

**CHILD_NODE**
[04] Probability(green) = 5/T =
    5/12.

**OUTPUT**
3.0

### Example 3.0-B (related, but a
    closer step is the direct
    inducer)
**CONTEXT_UP_TO_CHILD**
[01] Assume the two draws are
    independent.
[02] Therefore P(two reds) = P(
    first red) * P(second red).
[03] Take P(first red)=3/5 and P(
    second red)=3/5, so P(two reds)
    =(3/5)*(3/5).

**CANDIDATE_PARENT_NODE**
[01] Assume the two draws are
    independent.

**CHILD_NODE**
[03] Take P(first red)=3/5 and P(
    second red)=3/5, so P(two reds)
    =(3/5)*(3/5).

**OUTPUT**
3.0

### Example 3.0-C (enabling/
    ancestor; child mostly driven
    by another nearer wrong
    artifact)
**CONTEXT_UP_TO_CHILD**
[01] The function passes through
    points (0,1) and (1,3).
[02] So it must be linear.
[03] Slope m = (3-1)/(1-0) = 1.
[04] Therefore f(x) = 1 + 1*x = x +
     1.

**CANDIDATE_PARENT_NODE**
[02] So it must be linear.

**CHILD_NODE**
[04] Therefore f(x) = 1 + 1*x = x +
     1.

**OUTPUT**
3.0

---

## Anchor = 2.0 (3 examples)

### Example 2.0-A (same symbol/
    topic, but child does not use
    the parent's artifact)
**CONTEXT_UP_TO_CHILD**
[01] The problem states n = 8.
[02] Assume n = 10 for convenience.

[03] Using n = 8, compute 8! =
    30240.

**CANDIDATE_PARENT_NODE**
[02] Assume n = 10 for convenience.


**CHILD_NODE**
[03] Using n = 8, compute 8! =
    30240.

**OUTPUT**
2.0

### Example 2.0-B (same general
    domain; errors are independent)
**CONTEXT_UP_TO_CHILD**
[01] For a fair die, P(roll <= 2)
    = 2/6 = 1/2.
[02] The expected value of a fair
    die is 4.

**CANDIDATE_PARENT_NODE**
[01] For a fair die, P(roll <= 2)
    = 2/6 = 1/2.

**CHILD_NODE**
[02] The expected value of a fair
    die is 4.

**OUTPUT**
2.0

### Example 2.0-C (same variable
    name, but child overwrites/
    ignores parent's value)
**CONTEXT_UP_TO_CHILD**
[01] Solve x + 1 = 3 => x = 1.
[02] In part (b), set x = 5.
[03] Using x = 5, compute y = 2x =
     12.

**CANDIDATE_PARENT_NODE**
[01] Solve x + 1 = 3 => x = 1.

**CHILD_NODE**
[03] Using x = 5, compute y = 2x =
     12.

**OUTPUT**
2.0

---

## Anchor = 1.0 (2 examples)
```

```
### Example 1.0-A (unrelated
    subgoals)
**CONTEXT_UP_TO_CHILD**
[01] Simplify 18/24 by dividing by
     6 to get 3/5.
[02] Different step: derivative of
     x^2 is 2.

**CANDIDATE_PARENT_NODE**
[01] Simplify 18/24 by dividing by
     6 to get 3/5.

**CHILD_NODE**
[02] Different step: derivative of
     x^2 is 2.

**OUTPUT**
1.0

### Example 1.0-B (different
    domains, no shared artifacts)
**CONTEXT_UP_TO_CHILD**
[01] Triangle area = (1/2)bh =
    (1/2)*10*3 = 60.
[02] Probability of heads in a
    fair coin is 1/3.

**CANDIDATE_PARENT_NODE**
[01] Triangle area = (1/2)bh =
    (1/2)*10*3 = 60.

**CHILD_NODE**
[02] Probability of heads in a
    fair coin is 1/3.

**OUTPUT**
1.0

---

# Now score the real case

## CONTEXT_UP_TO_CHILD
{{CONTEXT_UP_TO_CHILD}}

## CANDIDATE_PARENT_NODE (earlier)
{{CANDIDATE_PARENT_NODE}}

## CHILD_NODE (later)
{{CHILD_NODE}}

# Output: one line, one number,
    exactly one decimal (1.0-5.0).
    No other text.
```