

000 001 002 003 004 005 006 007 008 009 010 011 012 013 014 015 016 017 018 019 020 021 022 023 024 025 026 027 028 029 030 031 032 033 034 035 036 037 038 039 040 041 042 043 044 045 046 047 048 049 050 051 052 053 SEEDTHINK: TEST-TIME CONTROL VIA SEED-THOUGHT INITIALIZATION

Anonymous authors

Paper under double-blind review

ABSTRACT

Large reasoning models (LRMs) achieve impressive performance through extended chains of thought, but this substantially increases inference overhead, making efficiency a critical bottleneck. In this paper, we first show that initializing the reasoning process with high-quality seed thoughts can steer the model away from unproductive “overthinking” and produce more efficient reasoning trajectories. Critically, we find that the optimal granularity of this seed — from a high-level outline to a detailed solution — depends on problem difficulty. Motivated by this, we propose SeedThink, a novel framework that adaptively selects the seed granularity based on an estimate of problem difficulty. Specifically, SeedThink features two core innovations: (1) a **difficulty-aware seeding policy that dynamically generates seed thoughts** to reduce repetitive verification and prune unproductive branches; and (2) **seamless integration with enhanced speculative decoding**, where seed thoughts are repurposed as a model-free draft corpus to achieve dual-path acceleration — shorter reasoning traces and faster token generation. Our experiments show that SeedThink significantly reduces inference costs while largely preserving performance. Notably, our method achieves up to a 4.1 \times end-to-end speedup and a 68% reduction in generation length with minimal accuracy degradation, highlighting the promise of adaptive initialization for balancing reasoning quality and efficiency.

1 INTRODUCTION

Large language models increasingly benefit from test-time compute, a phenomenon known as test-time scaling (Snell et al., 2024; Muennighoff et al., 2025). Recent large reasoning models (LRMs), such as OpenAI o1 (Jaech et al., 2024), DeepSeek-R1 (Guo et al., 2025), and Qwen3 (Yang et al., 2025), demonstrate that extended chains of thought (Wei et al., 2022a), paired with reflection, backtracking, and self-verification techniques, significantly enhance performance on complex reasoning problems. Given a query, these models generate a deliberate think trajectory before producing a concise final solution, thereby yielding better results across mathematical and scientific reasoning benchmarks (Cobbe et al., 2021; Hendrycks et al., 2021; He et al., 2024; Rein et al., 2024; Lewkowycz et al., 2022).

However, these advances introduce two significant and interconnected challenges: the inherent **inefficiency** and **path dependence** of test-time computation. While extended chains of thought can substantially improve performance, they often lead to “overthinking” — expending extra tokens on repetitive verification, unproductive branching, and exploration long after a sufficient solution has been found (Chen et al., 2024; Cuadron et al., 2025). A single *Thinking* trajectory is typically five times longer than a direct *No-Thinking* response, even on simple problems where both modes produce the same correct answer (Zhang et al., 2025a; Fan et al., 2025; Liang et al., 2025).

A natural approach to improving efficiency is the mode-switching paradigm, widely studied in recent works (Zhang et al., 2025a;b; Li et al., 2025a; Luo et al., 2025; Liang et al., 2025; Ha et al., 2025; Xing et al.). It switches between the efficient *No-Thinking* mode for easy problems and the effective *Thinking* mode for hard ones. However, this paradigm has a critical limitation: its efficacy declines sharply as problem difficulty increases. It gains efficiency only on problems that are already within the *No-Thinking* mode’s capability, failing to address the core inefficiency and path dependence of “Thinking” on truly challenging tasks. An emerging direction to bridge this gap is to integrate the

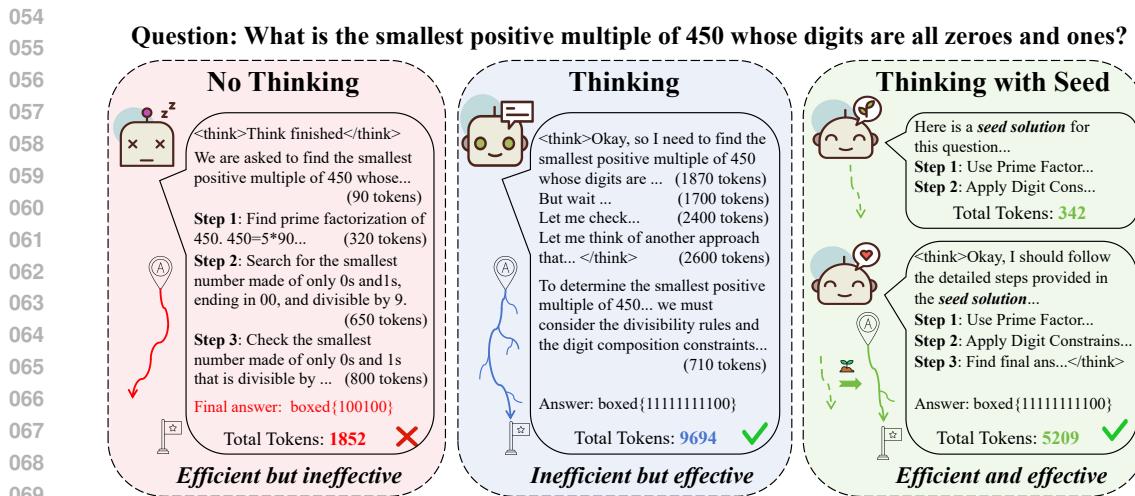


Figure 1: Qwen3-32B’s Thinking and No-Thinking mode generating process for an example from the MATH (Hendrycks et al., 2021) benchmark. No Thinking executes a single-pass, myopic trajectory that under-explores the solution space, while Thinking broadens coverage via multi-pass, multi-angle checks, but spends many tokens on unproductive trajectories. Thinking with Seed initializes the reasoning process with a concise seed that steers the trajectory to productive regions, shortens the reasoning path, and prunes unnecessary branches, thereby improving the efficiency of model reasoning.

two modes or multiple agents into a cooperative pipeline (Fan et al., 2025; Pan et al., 2025; Wang et al., 2025d).

The inefficiency is further compounded by the highly path-dependent nature of the reasoning trajectory — the same thought budget can produce dramatically different outcomes depending on how the reasoning process is initialized. Both empirically and intuitively, early steps constrain what the model explores next, so the initial state of the reasoning process can determine whether the model converges quickly on a correct explanation or wanders into unproductive regions, and can even affect the model’s instruct-following and safety performance (Fan et al., 2025; Lin et al., 2025; Wu et al., 2025c).

To address these challenges, in this paper, we investigate a **test-time steering technique via seed-thought initialization**: how to start the thinking process so that limited test-time compute is steered toward productive regions of the search space. We operationalize this idea using seed thoughts: concise, high-quality initial trajectories that shape the early steps of reasoning. As shown in Figure 1, *No-Thinking* is fast but myopic, while standard *Thinking* broadens coverage at a significant token cost. Our **Thinking with Seed initializes reasoning with a concise seed that captures essential pivot reasoning anchors** (e.g., subgoals, key lemmas, variable definitions, candidate equation forms), which can steer exploration toward productive regions and prune unnecessary branches.

The **thought generation process in LRM**s can be conceptualized as a non-convex search through language space. This search is initialized with the user prompt and a seed thought. The resulting thought trajectories — sequences of tokens evaluated by either process reward models (PRMs) or outcome reward models (ORMs) — represent the model’s paths to a solution. Therefore, our method fundamentally alters the search dynamics: rather than starting from scratch, the *Thinking with Seed* mode is initialized at a point already oriented toward a promising basin in the search space. This head start significantly shortens the reasoning trajectory. Furthermore, the seed acts as a lightweight greedy backbone that effectively constrains the branching factor during subsequent exploration, yet remains revisable. By pruning vast regions of the search space that are unproductive from the outset, the method improves efficiency and mitigates path dependence by steering the reasoning process toward high-probability solution basins, thereby increasing the chances of rapid and correct convergence.

Having established the importance of seed thoughts, a natural question arises: *does a one-size-fits-all seed exist?*. In practice, seed granularity is a direct lever on the trade-off between guidance

108 and flexibility. In our pilot experiments, we consistently observed that for easier problems, more
 109 detailed seeds collapse the search and improve efficiency without increasing error, because even
 110 imperfect specifics rarely mislead the model. For harder problems, however, higher-level seeds are
 111 preferable, because they provide strategic scaffolding and flexibility without locking the model into
 112 potentially misleading, concrete steps. These findings motivate a **difficulty-aware seeding policy**
 113 **that adjusts seed granularity** to dynamically shape the reasoning trajectory, instead of relying on
 114 fixed initialization strategies.

115 The utility of the generated seed thoughts extends beyond guiding the reasoning search. We further
 116 show that the two-stage generation paradigm naturally lends itself to Speculative Decoding (SD)
 117 (Leviathan et al., 2023; Li et al., 2024; Cai et al., 2024; Liu et al., 2024; Gloeckle et al., 2024; Li
 118 et al., 2025b; Sun et al., 2025) by producing highly compatible token sequences. Since the second-
 119 stage *Thinking with Seed* generation is explicitly conditioned on the first-stage seed thoughts, the
 120 outputs exhibit strong token-level alignment — a critical factor for speculative acceleration. As
 121 shown in Figure 1, the *Thinking with Seed* trajectory closely follows the reasoning path established
 122 by the seed, resulting in higher token acceptance rates than standard auto-regressive drafting or
 123 corpus-based suffix matching.

124 **Our contributions are as follows.**

125

- 126 • We conduct a systematic study of how the granularity of seed thoughts affects the performance
 127 of LRM s.
- 128 • We propose a difficulty-aware seed-thought generation framework that adaptively selects
 129 granularity based on problem difficulty.
- 130 • To the best of our knowledge, we are the first to integrate *Thinking* and *No-Thinking* mode
 131 with speculative decoding, achieving dual-path speedups — shorter deliberative traces and
 132 faster token generation enabled by reusing seed thoughts as a model-free speculative corpus
 133 — within a single cohesive framework.
- 134 • Our experiments show up to $3.1 \times$ end-to-end acceleration and 45.0% reduction in generation
 135 tokens with only 2.4% accuracy degradation on the MATH500 dataset on Qwen3
 136 models, delivering superior efficiency–accuracy trade-offs compared to existing methods
 137 and advancing the state of the art. Seed thoughts effectively warm-up speculative decoding
 138 corpora, obtaining 16% additional reasoning acceleration than the hybrid SAM with
 EAGLE3.

139

140 2 RELATED WORKS

141

142 **Chain-of-Thought and Reasoning Models.** Chain-of-Thought (CoT) reasoning denotes a
 143 paradigm for enhancing the transparency and accuracy of large language models (LLMs) in complex
 144 reasoning tasks by explicitly generating intermediate logical steps that culminate in a final
 145 output (Wei et al., 2022b). Early foundational works (Wei et al., 2022b; Kojima et al., 2022) demonstrated
 146 that CoT prompting — either via few-shot examples or zero-shot instructions — significantly
 147 improves LLM performance on mathematical and logical benchmarks. After that, diverse CoT variants,
 148 including structured CoT (Li et al., 2023), ToT (Yao et al., 2023a), self-reflection (Ji et al.,
 149 2023), and self-correction (Huang et al., 2023), are proposed to enhance traceability and accuracy.

150 Chain-of-Thought and its variants guide LLMs to learn from in-context prompts and generate internal
 151 reasoning before generating the final answers. These works establish the cornerstones of recent
 152 reasoning models, agent, and Agentic AI, where models are trained to obtain the intrinsic capability
 153 of generating CoTs (Yao et al., 2023b; Jaech et al., 2024; Muennighoff et al., 2025; Guo et al.,
 154 2025; Yang et al., 2025). However, the initial solution or guidance generated during thinking may
 155 be inefficient and ineffective, resulting in overthinking and underthinking (Cuadron et al., 2025).

156 **Efficient Reasoning.** Recently, many training-free or training-based methods are proposed to re-
 157 duce the number of generated tokens, improving the efficiency of test-time scaling. Auto-thinking
 158 utilizes external classifiers or LRM s’ intrinsic task complexity classification capabilities to automati-
 159 cally switch between fast and slow thinking (Zhang et al., 2025c; Xing et al.). Self-evaluation
 160 methods prompt LRM s to assess their own confidence and decide when to stop reasoning (Wang
 161 et al., 2025c). Model merging directly merge LLM weights to trade-off the reasoning capability and
 cost (Team et al., 2025; Wu et al., 2025a). Multi-model frameworks use auxiliary evaluation mod-

162 els, reward models, or thought proposers to guide decoding (Fu et al., 2024; Kuhn et al., 2023; Sun
 163 et al., 2024). There are also concurrent works on improve LRM efficiency by generating thoughts
 164 with smaller reasoning models for speculative reasoning (Pan et al., 2025; Wang et al., 2025b;d).
 165 CoThink (Fan et al., 2025) leverages an instruction model to guide reasoning and reduce reasoning
 166 steps. Supervised Fine-Tuning (SFT) or Reinforcement Learning (RL) based methods train LRMs
 167 to with compact reasoning trajectories or length-aware rewards, respectively (Chen et al., 2024; Hou
 168 et al., 2025; Xia et al., 2025; Zhang et al., 2025a; Chen et al., 2025; Fang et al., 2025).

169 **Speculative Decoding.** Due to the lossless advantage, speculative decoding emerged as a widely
 170 studied latency mitigation paradigm by introducing a "draft-verify" pipeline: a lightweight draft
 171 model proposes candidate token sequences, and the target LLM verifies these candidates in parallel,
 172 accepting valid tokens to reduce sequential computation(Leviathan et al., 2023; Chen et al., 2023).
 173 After that, n-gram based drafter (He et al., 2023; Zhao et al., 2024; Luo et al., 2024; Hu et al., 2024;
 174 Oliaro et al., 2025), and auxiliary lightweight trained model(Li et al., 2024; 2025b; Cai et al., 2024;
 175 Liu et al., 2024; Gloeckle et al., 2024) based methods, infrastructure optimization (Miao et al., 2024)
 176 are proposed to improve the practical inference. Sun et al. (2025) systemically compared different
 177 existing speculative decoding methods for reasoning acceleration. Recently, He et al. (2025) pro-
 178 posed to accelerate RL training with SD enhanced with rollout responses tokens in adjacent training
 179 epochs.

180 Our work differs from prior studies in two key respects: 1) We focus on the synergy between Thinking
 181 and No-Thinking modes, rather than simply switching between them. Seed thoughts are gen-
 182 erated in the No-Thinking mode and capture critical constraints and pivot **reasoning anchors** of
 183 the searching trajectories. They serve as additional hints that improve LRM confidence, speedup
 184 search convergence, and guide effective and efficient thinking, rather than searching from scratch.
 185 2) We study how to leverage additional contexts, including seed thoughts, to warm-up speculative
 186 decoding corpora to further accelerate LRMs.

3 PRELIMINARY AND OBSERVATIONS

3.1 BACKGROUND AND NOTATIONS

To ground our setting, we first fix notation and delineate the generation paradigms considered in this work. Let M denote a standard LRM. Given an input question q , the model generates a response sequence r . Two special tokens, $\langle \text{BOT} \rangle$ and $\langle \text{EOT} \rangle$, are used to demarcate the beginning and end of the explicit reasoning block.

Thinking Mode. The model generates a full chain-of-thought reasoning process before producing the final answer. The generation can be formally described as:

$$r_{\text{thinking}} = M(q + \langle \text{BOT} \rangle) = \langle \text{BOT} \rangle [\text{Thoughts}] \langle \text{EOT} \rangle [\text{Conclusion}]$$

No-Thinking Mode. The model is prompted to bypass the explicit reasoning process and generate the final answer directly. This is achieved by prepending a specific instruction inside the reasoning block that forces an immediate conclusion (Ma et al., 2025). The generation is:

$$r_{\text{nothinking}} = M(q + \langle \text{BOT} \rangle + s_{\text{skip}} + \langle \text{EOT} \rangle) = \langle \text{BOT} \rangle s_{\text{skip}} \langle \text{EOT} \rangle [\text{Conclusion}]$$

where s_{skip} is a fixed string (e.g., "Okay, I think I have finished thinking.") that signals the model to skip deliberative thinking .

Thinking with Seed. To mitigate the inefficiency and path-dependence inherent in the standard *Thinking* mode, we introduce a hybrid generation strategy termed *Thinking with Seed*. This mode leverages a concise, high-quality seed thought s_{seed} to initialize and guide the subsequent deliberative reasoning process. The seed s_{seed} acts as structured scaffolding that shapes the subsequent reasoning trajectory by providing an initial sketch of the solution path. The model is then prompted to generate a full reasoning trajectory, explicitly conditioned on the seed s . The generation is formalized as:

$$r_{\text{SeedThink}} = M(q + s_{\text{seed}} + \langle \text{BOT} \rangle) = \langle \text{BOT} \rangle [\text{Extended Thoughts}] \langle \text{EOT} \rangle [\text{Conclusion}]$$

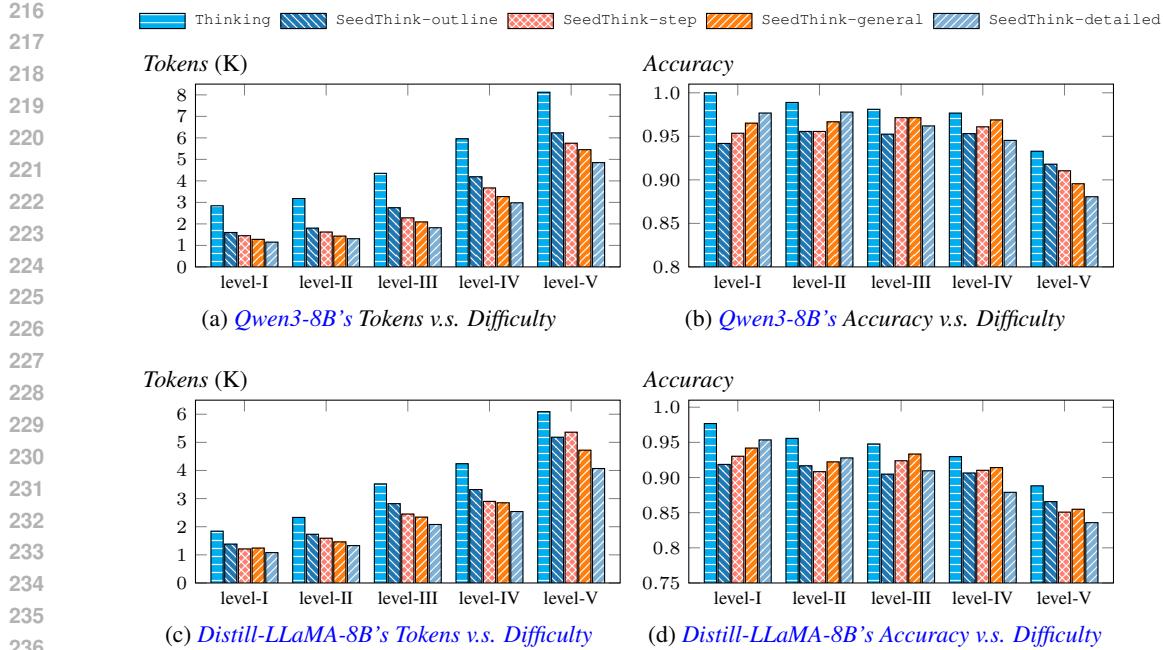


Figure 2: Comparison of Thinking mode and four different Thinking with Seed Thoughts (Seed-Think) variants using Qwen3-8B and DeepSeek-R1-Distill-LLaMA-8B across different difficulty levels of MATH500 dataset.

3.2 OBSERVATION

In this section, we investigate how the level of detail in seed thoughts affects the reasoning trajectories of LRM_s, revealing opportunities for optimization in LRM_s’ reasoning efficiency.

Experimental settings. We conduct a pilot study using the MATH500 datasets (Hendrycks et al., 2021) to analyze the impact of seed granularity on the performance of two widely used LRM_s: Qwen3-8B and DeepSeek-R1-Distill-LLaMA-8B. We define four levels of seed granularity:

- *Outline seed*: High-level strategic breakdown focusing only on outline without numbers, formulas, or calculations
- *Key-steps seed*: Main reasoning steps with important formulas/concepts mentioned.
- *General-approach seed*: Step-by-step plan with brief explanations, including formulas and concepts, but avoiding extensive calculations
- *Detailed-Steps seed*: Near-complete solution with formulas, initial setups, and simple calculations, leaving only complex reasoning for completion

We compare these against the baseline *Thinking* approach (no seed). For each problem, we generate seeds at each granularity level using specialized prompts. More details about parameter settings and prompts can be found in Appendix B.

Results and analysis. Our pilot study yields a clear central finding: the optimal granularity of a seed thought follows a non-monotonic relationship with problem difficulty. As illustrated in Figure 2, we observe a striking pattern: on the simplest problems (level-I/II), the most detailed seeds achieve the highest accuracy alongside the greatest efficiency gains; for problems of intermediate difficulty (level-III/IV), a balanced general approach seed proves most accurate; while on the hardest problems (level-V), high-level outline seeds become most robust, outperforming detailed alternatives.

This tri-phasic pattern highlights a sophisticated trade-off between **guidance** and **flexibility**. On simple problems, a highly specific seed provides strong, beneficial constraints that focus the search without causing harm. At medium difficulty, an intermediate level of guidance offers the ideal balance of direction and flexibility. For the most complex problems, however, high-level strategic scaffolding becomes essential to avoid leading the model down incorrect, overly constrained paths.

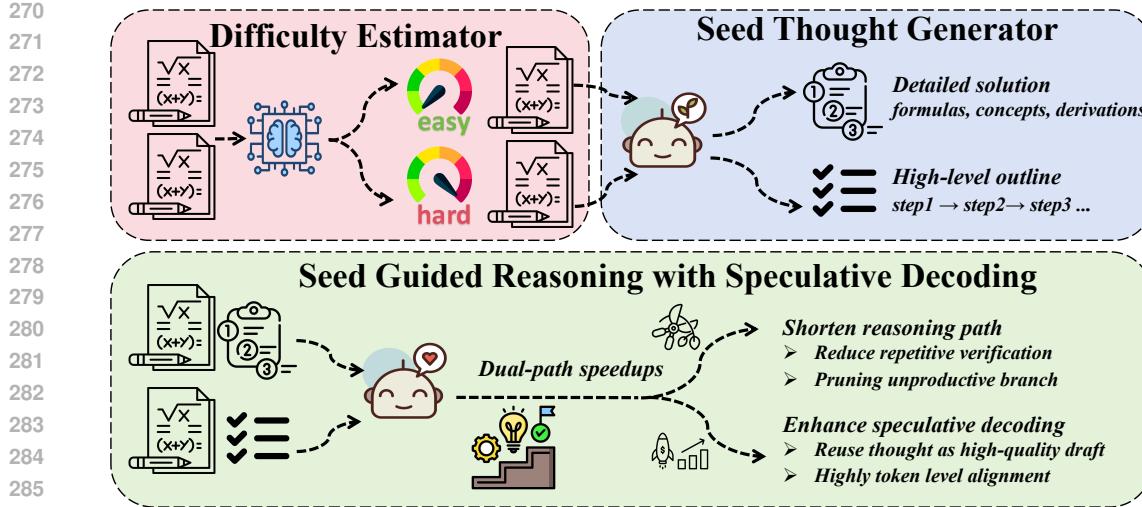


Figure 3: The SeedThink framework comprises three core modules: (1) Difficulty Estimator: Assesses problem complexity; (2) Seed Thought Generator: Produces a difficulty-aware seed solution; (3) Seed-Guided Reasoning: Utilizes the seed for focused reasoning and as a high-quality draft for speculative decoding, achieving dual-path acceleration.

The universal efficiency gains of SeedThink (23%-59% token reduction at all levels) confirm its value for efficient test-time control. The dramatic shift in the relative performance of seed types across the difficulty spectrum, however, reveals that a static seeding strategy is fundamentally sub-optimal. This finding directly motivates our subsequent proposal of a difficulty-aware seeding policy, which dynamically modulates seed granularity to track this shifting optimum, as detailed in the next section.

4 METHOD

As visualized in Figure 3, SeedThink improves reasoning efficiency by proposing difficulty-aware seed thoughts as additional thinking hints and token corpora to improve thinking generation confidence and enhance speculative decoding acceptance rate, respectively. The framework consists of three modules: 1) Difficulty estimator for more accurate alignment with problem difficulties in Section 4.2, 2) Seed thought generator, which generate suitable seed granularity based on the previous prompt difficulty estimation, in Section 4.3, 3) Seed guided reasoning with enhanced speculative decoding for shorter and faster test-time scaling in Section 4.4.

4.1 PROBLEM FORMULATION

Our SeedThink framework involves a two-stage generation process: (1) generating a seed thought with appropriate granularity, and (2) producing the final reasoning trajectory conditioned on the seed as formulated in Section 3.1. Thus, the key to this process is selecting the suitable seed granularity for diverse prompt difficulties to trade-off efficiency and performance.

We formulate the seed thought selection as an optimization problem that maximizes a composite objective function balancing efficiency against accuracy. Given a problem q with estimated difficulty d , we select the optimal seed granularity $g^* \in \mathcal{G}$ from our predefined seed set.

Formally, for a problem with difficulty d , the ideal seed granularity $g^* \in \mathcal{G}$ should maximize:

$$g^*(d) = \arg \max_{g \in \mathcal{G}} [\mathcal{E}(g, d) + \lambda \cdot \mathcal{A}(g, d)]$$

where $\mathcal{E}(g, d) = \frac{T_{\text{think}}(d) - T_g(d)}{T_{\text{think}}(d)}$ represents relative efficiency gain, $\mathcal{A}(g, d) = \frac{A_g(d) - A_{\text{think}}(d)}{A_{\text{think}}(d)}$ represents relative accuracy change, and $\lambda \geq 0$ is a trade-off coefficient. Here, $T_g(d)$ and $A_g(d)$ are the number of tokens and accuracy under granularity g at difficulty d , while $T_{\text{think}}(d)$ and $A_{\text{think}}(d)$ are the corresponding quantities under the original *Thinking* mode.

324 4.2 DIFFICULTY ESTIMATOR
325

326 Existing research from Liu et al. (2025) indicates that LRM^s fundamentally lack the capacity for ac-
327 curate, autonomous difficulty cognition. Equipping models with this metacognitive ability typically
328 necessitates carefully designed SFT or RL, which entail significant computational expense. For the
329 sake of efficiency, we avoid this training overhead by introducing a separate difficulty estimator as
330 a crucial first step in our methodology. Accordingly, we frame the problem of difficulty assess-
331 ment as a text classification task that is decoupled from the LRM^s own reasoning process. The
332 specific architecture, detailed training procedure, and an evaluation of the estimator[’]s effectiveness
333 are provided in Appendix D. The difficulty estimator module assesses the complexity level of input
334 problems to determine the appropriate seeding strategy. Formally, it maps a problem q to a difficulty
335 level $d \in \mathcal{D}$, where $\mathcal{D} = \{\text{Simple}, \text{Moderate}, \text{Hard}\}$ is the set of difficulty categories.
336

337 4.3 SEED THOUGHT GENERATOR

338 Based on our empirical findings in Section 3.2, we propose a deterministic seeding policy $\pi(d)$ that
339 directly maps difficulty to seed granularity:

$$\pi(d) = \begin{cases} \text{detailed steps} & \text{if } d = \text{Simple} \\ \text{key steps} & \text{if } d = \text{Moderate} \\ \text{outline} & \text{if } d = \text{Hard} \end{cases}$$

344 The seed generator then produces:
345

$$346 \quad \text{Seed}_{\pi(d)} = M(\mathcal{T}_{\pi(d)}(q)) \quad (1)$$

347 where \mathcal{T}_g is the prompt template for granularity g .
348

349 4.4 SEED GUIDED REASONING WITH ENHANCED SPECULATIVE DECODING
350

351 Building upon the difficulty-aware generation of the seed thought $\text{Seed}_{\pi(d)}$, the seed guided reason-
352 ing process produces the final output by explicitly conditioning on the seed:

$$353 \quad r_{\text{SeedThink}} = M(q, \text{Seed}_{\pi(d)}) \quad (2)$$

355 The sequential dependency between Eq. (1) and Eq. (2) is the cornerstone of our method. The final
356 reasoning output $r_{\text{SeedThink}}$ is explicitly conditioned on the $\text{Seed}_{\pi(d)}$, creating a strong **path dependence**.
357 Combined with the **model consistency** of using the same model M for both generations,
358 this design ensures a high degree of token-level alignment between the two sequences. It is this
359 inherent alignment that makes the seed from Eq. (1) a naturally high-quality draft for speculative
360 decoding, as the verification of the seed tokens during the generation of Eq. (2) achieves higher ac-
361 ceptance rates than drafts lacking such a causal link. Thus, the seed thought serves a dual purpose:
362 it structures the reasoning path deterministically, while its token sequence functions as an internal,
363 optimized draft for accelerated generation.

363 We formally analyze why this conditioning leads to high token-level alignment in Appendix F,
364 providing theoretical guarantees for our method[’]s efficiency.
365

366 5 EVALUATION
368

369 5.1 EXPERIMENTAL SETUP

370 **Datasets.** We evaluate on three math datasets of increasing difficulty: **GSM8K**(1,319 grade-school
371 problems) (Cobbe et al., 2021), **MATH500** (a 500-problem subset of the MATH benchmark con-
372 taining high-school competition problems) (Hendrycks et al., 2021), and **AIME 2024** (30 challeng-
373 ing Olympiad-level problems). We report solution accuracy, wall time, and the average number of
374 generated tokens (response length) as key metrics. We report accuracy as Pass@1 (percentage of
375 problems solved correctly by the first sample). All generated answers are checked using Hugging-
376 Face¹’s official Math-Verify¹ for fairness and reproducibility. Due to the small size of AIME 2024,
377

¹<https://github.com/huggingface/Math-Verify>

378
379
380
381
382
383 Table 1: Accuracy, average response length (Length), and average runtime of different methods on
384 three math benchmarks. The best and second results are bolded and underlined, respectively. For
385 ease of comparison, the results include absolute accuracy differences, speedup ratios relative to the
386 Think baseline, and relative length differences.
387

Method	GSM8K			MATH 500			AIME 2024		
	Accuracy \uparrow	Runtime (s) \downarrow	Length \downarrow	Accuracy \uparrow	Runtime (s) \downarrow	Length \downarrow	Accuracy \uparrow	Runtime (s) \downarrow	Length \downarrow
<i>DeepSeek-R1-Distill-LLaMA-8B</i>									
<i>Think</i>	76.50%	113.9	4029	87.60%	104.6	3627	46.67%	299.1	11288
<i>NoThink</i>	76.80% \pm 0.3%	12.0 \pm 0.5 \times	320 \pm 0.1 \times	66.20% \pm 21.4%	19.7 \pm 3.3 \times	682 \pm 1.2%	10.00% \pm 36.7%	60.6 \pm 4.9 \times	1917 \pm 83.0%
<i>JoinThink</i>	80.12% \pm 3.6%	143.2 \pm 0.8 \times	4968 \pm 23.3%	90.20% \pm 2.6%	144.2 \pm 0.7 \times	5018 \pm 24.5%	43.33% \pm 3.3%	478.0 \pm 0.6 \times	15220 \pm 34.8%
<i>CoThink</i>	77.64% \pm 1.1%	98.1 \pm 1.2 \times	3433 \pm 14.8%	81.20% \pm 6.4%	85.7 \pm 1.2 \times	2890 \pm 19.9%	40.00% \pm 6.6%	233.0 \pm 1.3 \times	8630 \pm 23.5%
SeedThink (ours)	78.22% \pm 1.7%	61.3 \pm 1.9 \times	3647 \pm 9.5%	82.80% \pm 4.8%	45.1 \pm 2.3 \times	2611 \pm 28.0%	43.33% \pm 3.3%	131.3 \pm 2.3 \times	9049 \pm 19.9%
<i>Qwen3-8B</i>									
<i>Think</i>	95.00%	94.6	2405	96.20%	191.4	5432	64.16%	358.6	11863
<i>NoThink</i>	90.22% \pm 4.8%	101.3 \pm 9.2 \times	956 \pm 89.0%	83.80% \pm 12.4%	36.2 \pm 5.3 \times	956 \pm 82.4%	26.67% \pm 37.5%	116.1 \pm 3.1 \times	3634 \pm 69.4%
<i>JoinThink</i>	95.83% \pm 0.8%	109.4 \pm 0.9 \times	2783 \pm 15.7%	97.00% \pm 0.8%	251.4 \pm 0.8 \times	7040 \pm 29.6%	60.00% \pm 4.16%	568.1 \pm 0.6 \times	17823 \pm 0.2%
<i>CoThink</i>	94.62% \pm 0.4%	37.6 \pm 2.5 \times	865 \pm 64.0%	94.20% \pm 2.0%	146.4 \pm 1.3 \times	4015 \pm 26.1%	63.33% \pm 0.8%	401.6 \pm 0.9 \times	12730 \pm 7.3%
SeedThink (ours)	94.84% \pm 0.2%	24.9 \pm 3.8 \times	801 \pm 46.7%	94.40% \pm 1.8%	72.2 \pm 2.7 \times	3630 \pm 33.2%	61.16% \pm 3.0%	176.4 \pm 2.0 \times	10268 \pm 13.5%
<i>Qwen3-14B</i>									
<i>Think</i>	95.53%	75.5	1881	96.80%	176.9	4741	66.67%	395.6	11291
<i>NoThink</i>	90.22% \pm 4.8%	111.9 \pm 6.3 \times	277 \pm 85.3%	86.40% \pm 10.4%	34.5 \pm 5.1 \times	896 \pm 81.1%	35.33% \pm 33.3%	159.5 \pm 2.5 \times	4453 \pm 60.6%
<i>JoinThink</i>	96.12% \pm 0.6%	96.7 \pm 0.8 \times	2388 \pm 27.0%	97.40% \pm 0.6%	232.3 \pm 0.8 \times	6076 \pm 28.2%	66.67% \pm 0.0%	611.4 \pm 0.6 \times	17057 \pm 1.1%
<i>CoThink</i>	93.86% \pm 1.7%	28.2 \pm 2.7 \times	680 \pm 63.8%	93.80% \pm 3.0%	114.5 \pm 1.5 \times	2959 \pm 38.0%	72.50% \pm 5.83%	350.5 \pm 1.1 \times	9879 \pm 12.5%
SeedThink (ours)	94.39% \pm 1.1%	18.4 \pm 4.1 \times	588 \pm 48.7%	94.60% \pm 2.2%	60.7 \pm 2.9 \times	2761 \pm 42.0%	70.83% \pm 4.16%	188.5 \pm 2.1 \times	8744 \pm 22.4%
<i>Qwen3-32B</i>									
<i>Think</i>	95.30%	101.6	1731	97.20%	282.6	4657	75.00%	836.3	12571
<i>NoThink</i>	91.52% \pm 3.8%	18.4 \pm 5.5 \times	286 \pm 83.5%	86.40% \pm 10.8%	46.9 \pm 6.0 \times	774 \pm 83.4%	26.67% \pm 48.3%	170.9 \pm 4.9 \times	2769 \pm 78.0%
<i>JoinThink</i>	96.41% \pm 1.1%	123.6 \pm 0.8 \times	2099 \pm 21.3%	97.60% \pm 0.4%	358.3 \pm 0.8 \times	5924 \pm 27.2%	75.00% \pm 0.0%	1267.9 \pm 0.7 \times	19337 \pm 5.8%
<i>CoThink</i>	94.47% \pm 0.8%	42.3 \pm 2.4 \times	620 \pm 64.2%	94.40% \pm 2.8%	163.0 \pm 1.7 \times	2891 \pm 38.0%	73.33% \pm 1.7%	588.6 \pm 1.4 \times	9416 \pm 25.1%
SeedThink (ours)	93.93% \pm 1.4%	24.6 \pm 4.1 \times	560 \pm 67.6%	94.80% \pm 2.4%	90.1 \pm 3.1 \times	2559 \pm 45.0%	72.50% \pm 2.5%	301.7 \pm 2.8 \times	8408 \pm 33.1%

399 we generate 4 responses per problem and report the average performance. For all models, we use
400 a 16K context window. Following the recommendations to balance diversity and coherence in the
401 official model cards, we set the temperature to 0.6 for *Thinking* mode and 0.7 for *No-Thinking* mode,
402 respectively.

403 **Models.** We conduct our main experiments on four publicly available Large Reasoning Models
404 (LRMs) of varying scales: DeepSeek-R1-Distill-LLaMA-8B (Guo et al., 2025), Qwen3-8B, Qwen3-
405 14B, and Qwen3-32B (Yang et al., 2025). These models are specifically designed for complex
406 reasoning and Qwen3 natively support both *Thinking* and *No-Thinking* generation modes, making
407 them ideal testbeds for our proposed efficiency improvements.

408 **Baselines.** We compare our approach against four representative baseline methods:

- 410 • *Think*: The standard *Thinking* mode, where the model generates a full, multi-step reasoning
411 trace before producing a final answer. This represents the high-cost, high-performance
412 upper bound for LRMs.
- 413 • *NoThink*: The standard *No-Thinking* mode, where the model generates a direct, concise an-
414 swer without an explicit reasoning trace. This represents the low-cost, lower-performance
415 baseline.
- 416 • *JointThink* (Wu et al., 2025b): A calibration-based method that first generates answers in
417 parallel using both *Thinking* and *No-Thinking* modes. If the answers from two modes are
418 inconsistent, it triggers a second *Thinking* round conditioned on both candidates’ answers
419 to improve robustness.
- 420 • *CoThink* (Fan et al., 2025): A two-stage pipeline where an Instruct model first creates a
421 very short reasoning outline, which is then refined by a reasoning model.

5.2 MAIN RESULTS

424 **Superior Efficiency-Effectiveness Trade-off.** As shown in Table 1, our proposed SeedThink
425 method achieves a remarkable balance between computational efficiency and reasoning accuracy
426 across all model scales and benchmarks, achieving up to 4.1 \times reasoning acceleration. On the chal-
427 lenging MATH 500 dataset, SeedThink maintains competitive accuracy while accelerating inference
428 by 2.3 \sim 3.1 \times and reducing token length by 28.0%–45.0% compared to standard Thinking mode,
429 outperforming the state-of-the-art (SOTA) baseline CoThink (Fan et al., 2025) in all three metrics.
430 While CoThink utilizes only coarse-grain outlines, which proves sub-optimal across diverse diffi-
431 culty levels, our method dynamically selects seed granularity based on the problem difficulty. This
432 adaptability allows SeedThink to provide detailed guidance for simple problems where specificity is

432 beneficial, while maintaining the flexibility of high-level outlines for complex problems where rigid
 433 guidance may be detrimental.

434
 435 The advantages of SeedThink become particularly evident when compared to alternative efficiency-
 436 oriented methods. While JointThink achieves modest accuracy improvements of 0.4%-2.6% on
 437 MATH500, it leads to $20 \sim 30\%$ slowdown and requires $24 \sim 29\%$ longer reasoning trajectories. In
 438 contrast, SeedThink delivers substantially greater efficiency gains while maintaining accuracy loss
 439 within 2.4% of the standard Thinking baseline. On the more challenging AIME 2024 benchmark,
 440 SeedThink consistently outperforms both alternatives, often matching or even improving accuracy
 441 while improving efficiency more than $2\times$ across all model scales. These results demonstrate that
 442 SeedThink achieves a superior efficiency-effectiveness trade-off, providing the practical benefits of
 443 dramatically reduced computational costs while preserving the reasoning quality of LRM.

444 **Dual-path speedups.** SeedThink achieves up to $4.1\times$ acceleration through two complementary
 445 mechanisms: (1) reduced reasoning trajectory length via strategic seed thought guidance, and (2)
 446 speculative decoding acceleration enabled by high token-level alignment between seed thoughts and
 447 final reasoning paths. The $9.5 \sim 68.8\%$ token length reductions directly translate to proportionally
 448 fewer decoding steps, while the even greater speedup of $1.9 \sim 4.1\times$ indicates additional gains
 449 from the reused seed thought as high-quality drafts for speculative decoding. We further conduct an
 450 ablation study in Section 5.3 about the effect of reusing them for warming up SD corpora.

452 5.3 ABLATION STUDY ON SPECULATIVE DECODING

453 To validate the core hypothesis from Section 4.4—that the **path dependence** between seed gen-
 454 eration and final reasoning creates superior token-level alignment for speculative decoding—we
 455 conduct a carefully designed ablation study comparing three distinct paradigms.

456 First, we reproduce existing speculative decoding approaches: **Think + SAM** (Hu et al., 2024) and
 457 **Think + SAM + Eagle3** (Li et al., 2025b), which represent SOTA model-free and model-based
 458 methods applied to standard reasoning trajectories (Sun et al., 2025). Second, we test a warm-up
 459 setting where seeds serve only as external draft corpora for SAM, isolating the pure speculative
 460 utility of seeds from our path-dependent generation paradigm in Eq. (2).

461 Table 2 reveals a clear but modest distinction: while the warm-up configuration brings gains ($1.45\times$
 462 and $2.08\times$ speedups), these are consistently lower than SeedThink’s $1.56\times$ and $2.24\times$ speedups with
 463 the same components.

464 This performance gap provides direct evidence
 465 for our theoretical claim: the sequential depen-
 466 dence between Eqs. (1) and (2) is the cor-
 467 nerstone of our method’s effectiveness. When
 468 seed thoughts are used only as external cor-
 469 pora for warm-up, MAT gains remain limited.
 470 In contrast, **explicitly conditioning final rea-**
 471 **soning on seed thoughts creates genuine path**
 472 **dependence and yields measurably higher**
 473 **MAT.**

474 In addition, we investigated how the length of
 475 the seed affects speculative performance. The
 476 corresponding experimental results and analy-
 477 sis are provided in the appendix E.

478 The results conclusively demonstrate that it is not merely about supplying extra corpora from seed
 479 thoughts, but their **integrated role in the generation process** that enables superior speculative
 480 performance.

481 Our two-stage framework creates a cohesive reasoning trajectory where each step naturally follows
 482 from the previous, producing the high-quality token alignment that enhances both model-free and
 483 model-based speculative decoding approaches.

Method	MAT	Throughput	Speed up
Think	1.00	28.4	—
+ SAM	1.75	38.8	$1.37\times$
+ SAM + w	1.88	41.2	$1.45\times$
+ SAM + Eagle3	3.26	54.5	$1.92\times$
+ all components	3.41	59.0	$2.08\times$
SeedThink	1.00	28.6	$1.00\times$
+ SAM	1.94	44.2	$1.56\times$
+ SAM + Eagle3	3.65	63.6	$2.24\times$

484 Table 2: Ablation study on MATH500 using Qwen3-
 485 4B, comparing speculative decoding performance. “w”
 486 denotes warm-up. MAT: Mean Accepted Tokens;
 487 Throughput: tokens per second. Speed up are nor-
 488 malized to baseline Thinking.

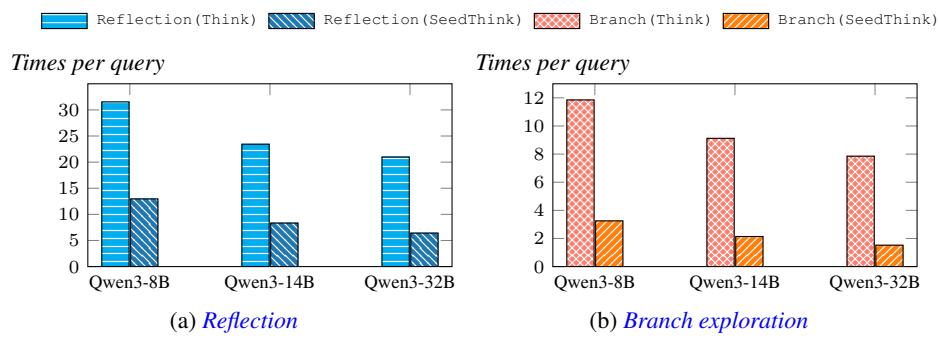


Figure 4: Comparison of Reflection token and Branch exploration tokens of Think and SeedThink on Qwen3-8B, Qwen3-14B, Qwen3-32B

5.4 ANALYSIS OF REASONING TRAJECTORY OPTIMIZATION

The inefficiency of current LRM s is largely driven by two unproductive behaviors: *repetitive verification and redundant branching explorations*. These behaviors can be measured with the number of reflection and branch tokens, which are defined below:

Reflection represents self-verification behaviors where the model double-checks its reasoning through repetitive validation cycles. This manifests linguistically through phrases like “wait”, “let me check”, and “let me verify”—indicators of conservative reasoning patterns that often continue well after a solution has been adequately established (Wang et al., 2025a).

Branch exploration encompasses the model’s tendency to consider alternative solution paths, signaled by terms like “alternatively”, “another way”, and “different approach”. While potentially valuable for complex problem-solving, uncontrolled branching frequently leads to computational inefficiency as the model explores unpromising directions.

Our analysis reveals that SeedThink significantly reduces these two inefficient reasoning behaviors. As shown in Figure 4, for Qwen3-8B, reflection tokens decrease by 58.9% while branch tokens drop by 72.6%. This pattern holds consistently across model scales, with Qwen3-32B models showing the most dramatic reductions: 69.5% fewer reflection tokens and 80.6% fewer branch tokens.

The mechanism behind this optimization lies in the seed’s role as a **reasoning anchor**. By providing high-quality initial guidance, SeedThink preemptively addresses the uncertainties that typically trigger verification cycles and alternative path exploration. The seed establishes a focused solution trajectory, reducing the need for extensive backtracking and minimizing unproductive branching.

6 CONCLUSIONS

In this work, we addressed the dual challenges of inefficiency and path dependence in large reasoning models (LRMs) that leverage test-time compute. Our proposed solution, SeedThink, fundamentally reframes the initialization of the reasoning process. By generating a concise, high-quality “seed thought” to guide the LRM, we steer its exploration toward productive regions of the search space. We demonstrated that the granularity of these seeds is critical and introduced a difficulty-aware seeding policy that adapts seed detail to problem complexity—using high-level seeds for hard problems to maintain flexibility and detailed seeds for easier ones to maximize efficiency.

This two-stage framework provides a dual-path acceleration. First, it shortens the reasoning trajectory by pruning unproductive branches, reducing token generation by up to 45.0%. Second, we showed for the first time that this paradigm naturally integrates with speculative decoding, using the seed thought as a highly quality draft. This synergy resulted in up to a 4.1 \times end-to-end speedup, achieving a state-of-the-art efficiency-accuracy trade-off. Our findings underscore the importance of guided initialization in complex, multi-step generation tasks. Future work could explore learning the optimal seeding policy directly, rather than relying on a difficulty classifier. Ultimately, by shaping the start of the journey, seed thoughts enable LRM s to reason more efficiently, making powerful deliberative reasoning more practical for real-world applications.

540 REFERENCES
541

542 Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D Lee, Deming Chen, and Tri
543 Dao. Medusa: Simple llm inference acceleration framework with multiple decoding heads. *arXiv
544 preprint arXiv:2401.10774*, 2024.

545 Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John
546 Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint
547 arXiv:2302.01318*, 2023.

548 Qiguang Chen, Dengyun Peng, Jinhao Liu, HuiKang Su, Jiannan Guan, Libo Qin, and Wanxiang
549 Che. Aware first, think less: Dynamic boundary self-awareness drives extreme reasoning effi-
550 ciency in large language models. *arXiv preprint arXiv:2508.11582*, 2025.

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

554 Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser,
555 Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to
556 solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

557 Alejandro Cuadron, Dacheng Li, Wenjie Ma, Xingyao Wang, Yichuan Wang, Siyuan Zhuang, Shu
558 Liu, Luis Gaspar Schroeder, Tian Xia, Huanzhi Mao, et al. The danger of overthinking: Examining
559 the reasoning-action dilemma in agentic tasks. *arXiv preprint arXiv:2502.08235*, 2025.

560 Siqi Fan, Peng Han, Shuo Shang, Yequan Wang, and Aixin Sun. Cothink: Token-efficient reasoning
561 via instruct models guiding reasoning models. *arXiv preprint arXiv:2505.22017*, 2025.

562 Gongfan Fang, Xinyin Ma, and Xinchao Wang. Thinkless: Llm learns when to think. *arXiv preprint
563 arXiv:2505.13379*, 2025.

564 Yichao Fu, Junda Chen, Siqi Zhu, Zheyu Fu, Zhongdongming Dai, Aurick Qiao, and Hao Zhang.
565 Efficiently serving llm reasoning programs with certainindex. *arXiv preprint arXiv:2412.20993*,
566 2024.

567 Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Syn-
568 naeve. Better & faster large language models via multi-token prediction. *arXiv preprint
569 arXiv:2404.19737*, 2024.

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

573 Rui Ha, Chaozhuo Li, Rui Pu, and Sen Su. From "aha moments" to controllable thinking: Toward
574 meta-cognitive reasoning in large reasoning models via decoupled reasoning and control. *arXiv
575 preprint arXiv:2508.04460*, 2025.

576 Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu,
577 Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for
578 promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint
579 arXiv:2402.14008*, 2024.

580 Jingkai He, Tianjian Li, Erhu Feng, Dong Du, Qian Liu, Tao Liu, Yubin Xia, and Haibo
581 Chen. History rhymes: Accelerating llm reinforcement learning with rhymerl. *arXiv preprint
582 arXiv:2508.18588*, 2025.

583 Zhenyu He, Zexuan Zhong, Tianle Cai, Jason D Lee, and Di He. Rest: Retrieval-based speculative
584 decoding. *arXiv preprint arXiv:2311.08252*, 2023.

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

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

597

598 Yuxuan Hu, Ke Wang, Xiaokang Zhang, Fanjin Zhang, Cuiping Li, Hong Chen, and Jing Zhang.
 599 Sam decoding: Speculative decoding via suffix automaton. *arXiv preprint arXiv:2411.10666*,
 600 2024.

601

602 Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song,
 603 and Denny Zhou. Large language models cannot self-correct reasoning yet. *arXiv preprint*
 604 *arXiv:2310.01798*, 2023.

605

606 Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec
 607 Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv*
 608 *preprint arXiv:2412.16720*, 2024.

609

610 Ziwei Ji, Tiezheng Yu, Yan Xu, Nayeon Lee, Etsuko Ishii, and Pascale Fung. Towards mitigating
 611 llm hallucination via self reflection. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 1827–1843, 2023.

612

613 Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large
 614 language models are zero-shot reasoners. *Advances in neural information processing systems*,
 35:22199–22213, 2022.

615

616 Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for
 617 uncertainty estimation in natural language generation. *arXiv preprint arXiv:2302.09664*, 2023.

618

619 Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative
 620 decoding. In *Proc. Int. Conf. Machine Learning*, 2023.

621

622 Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ra-
 623 masesh, Ambrose Sloane, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative
 624 reasoning problems with language models. *Advances in neural information processing systems*,
 35:3843–3857, 2022.

625

626 Jia Li, Ge Li, Yongmin Li, and Zhi Jin. Structured chain-of-thought prompting for code generation,
 627 2023. URL <https://arxiv.org/abs/2305.06599>.

628

629 Yang Li, Youssef Emad, Karthik Padthe, Jack Lanchantin, Weizhe Yuan, Thao Nguyen, Jason Weston,
 630 Shang-Wen Li, Dong Wang, Ilia Kulikov, et al. Naturalthoughts: Selecting and distilling
 631 reasoning traces for general reasoning tasks. *arXiv preprint arXiv:2507.01921*, 2025a.

632

633 Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle: Speculative sampling requires
 634 rethinking feature uncertainty. *arXiv preprint arXiv:2401.15077*, 2024.

635

636 Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. Eagle-3: Scaling up inference acceleration
 637 of large language models via training-time test. *arXiv preprint arXiv:2503.01840*, 2025b.

638

639 Guosheng Liang, Longguang Zhong, Ziyi Yang, and Xiaojun Quan. Thinkswitcher: When to think
 640 hard, when to think fast. *arXiv preprint arXiv:2505.14183*, 2025.

641

642 Zhengkai Lin, Zhihang Fu, Ze Chen, Chao Chen, Liang Xie, Wenxiao Wang, Deng Cai, Zheng
 643 Wang, and Jieping Ye. Controlling thinking speed in reasoning models. *arXiv preprint*
 644 *arXiv:2507.03704*, 2025.

645

646 Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao,
 647 Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint*
 648 *arXiv:2412.19437*, 2024.

649

650 Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike
 651 Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining
 652 approach. *arXiv preprint arXiv:1907.11692*, 2019.

648 Yongjiang Liu, Haoxi Li, Xiaosong Ma, Jie Zhang, and Song Guo. Think how to think: Mitigating
 649 overthinking with autonomous difficulty cognition in large reasoning models. *arXiv preprint*
 650 *arXiv:2507.02663*, 2025.

651

652 Haotian Luo, Haiying He, Yibo Wang, Jinluan Yang, Rui Liu, Naiqiang Tan, Xiaochun Cao,
 653 Dacheng Tao, and Li Shen. Ada-r1: Hybrid-cot via bi-level adaptive reasoning optimization.
 654 *arXiv preprint arXiv:2504.21659*, 2025.

655

656 Xianzhen Luo, Yixuan Wang, Qingfu Zhu, Zhiming Zhang, Xuanyu Zhang, Qing Yang, and
 657 Dongliang Xu. Turning trash into treasure: Accelerating inference of large language models
 658 with token recycling. *arXiv preprint arXiv:2408.08696*, 2024.

659

660 Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. Reasoning
 661 models can be effective without thinking. *arXiv preprint arXiv:2504.09858*, 2025.

662

663 Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae
 664 Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, et al. Specinfer: Accelerating large lan-
 665 guage model serving with tree-based speculative inference and verification. In *Proceedings of the*
 666 *29th ACM International Conference on Architectural Support for Programming Languages and*
 667 *Operating Systems, Volume 3*, pp. 932–949, 2024.

668

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

672

673 Gabriele Oliaro, Zhihao Jia, Daniel F Campos, and Aurick Qiao. Suffixdecoding: Extreme specu-
 674 lative decoding for emerging ai applications. In *The Thirty-ninth Annual Conference on Neural*
 675 *Information Processing Systems*, 2025.

676

677 Rui Pan, Yinwei Dai, Zhihao Zhang, Gabriele Oliaro, Zhihao Jia, and Ravi Netravali. Specrea-
 678 son: Fast and accurate inference-time compute via speculative reasoning. *arXiv preprint*
 679 *arXiv:2504.07891*, 2025.

680

681 David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Di-
 682 rani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a bench-
 683 mark. In *First Conference on Language Modeling*, 2024.

684

685 Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally
 686 can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

687

688 Hanshi Sun, Momin Haider, Ruiqi Zhang, Huitao Yang, Jiahao Qiu, Ming Yin, Mengdi Wang, Peter
 689 Bartlett, and Andrea Zanette. Fast best-of-n decoding via speculative rejection. *arXiv preprint*
 690 *arXiv:2410.20290*, 2024.

691

692 Shengyin Sun, Yiming Li, Xing Li, Yingzhao Lian, Weizhe Lin, Hui-Ling Zhen, Zhiyuan Yang,
 693 Chen Chen, Xianzhi Yu, Mingxuan Yuan, et al. Scaling up, speeding up: A benchmark of specu-
 694 lative decoding for efficient llm test-time scaling. *arXiv preprint arXiv:2509.04474*, 2025.

695

696 Kimi Team, A Du, B Gao, B Xing, C Jiang, C Chen, C Li, C Xiao, C Du, C Liao, et al. Kimi k1. 5:
 697 Scaling reinforcement learning with llms, 2025. URL <https://arxiv.org/abs/2501.12599>, 2025.

698

699 Chenlong Wang, Yuanning Feng, Dongping Chen, Zhaoyang Chu, Ranjay Krishna, and Tianyi
 700 Zhou. Wait, we don't need to" wait"! removing thinking tokens improves reasoning efficiency.
 701 *arXiv preprint arXiv:2506.08343*, 2025a.

702

703 Jikai Wang, Juntao Li, Lijun Wu, and Min Zhang. Efficient reasoning for llms through speculative
 704 chain-of-thought. *arXiv preprint arXiv:2504.19095*, 2025b.

705

706 Yiming Wang, Pei Zhang, Siyuan Huang, Baosong Yang, Zhuosheng Zhang, Fei Huang, and Rui
 707 Wang. Sampling-efficient test-time scaling: Self-estimating the best-of-n sampling in early de-
 708 coding. *arXiv preprint arXiv:2503.01422*, 2025c.

702 Zhihai Wang, Jie Wang, Jilai Pan, Xilin Xia, Huiling Zhen, Mingxuan Yuan, Jianye Hao, and
 703 Feng Wu. Accelerating large language model reasoning via speculative search. *arXiv preprint*
 704 *arXiv:2505.02865*, 2025d.

705 Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny
 706 Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in*
 707 *neural information processing systems*, 35:24824–24837, 2022a.

708 Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny
 709 Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in*
 710 *neural information processing systems*, 35:24824–24837, 2022b.

711 Han Wu, Yuxuan Yao, Shuqi Liu, Zehua Liu, Xiaojin Fu, Xiongwei Han, Xing Li, Hui-Ling Zhen,
 712 Tao Zhong, and Mingxuan Yuan. Unlocking efficient long-to-short llm reasoning with model
 713 merging. *arXiv preprint arXiv:2503.20641*, 2025a.

714 Haotian Wu, Bo Xu, Yao Shu, Menglin Yang, and Chengwei Qin. Thinking with nothinking cali-
 715 bration: A new in-context learning paradigm in reasoning large language models. *arXiv preprint*
 716 *arXiv:2508.03363*, 2025b.

717 Tong Wu, Chong Xiang, Jiachen T Wang, G Edward Suh, and Prateek Mittal. Effectively controlling
 718 reasoning models through thinking intervention. *arXiv preprint arXiv:2503.24370*, 2025c.

719 Heming Xia, Yongqi Li, Chak Tou Leong, Wenjie Wang, and Wenjie Li. Tokenskip: Controllable
 720 chain-of-thought compression in llms. *arXiv preprint arXiv:2502.12067*, 2025.

721 Zeyu Xing, Xing Li, Huiling Zhen, Xianzhi Yu, Mingxuan Yuan, and Sinno Jialin Pan. Large
 722 reasoning models know how to think efficiently. In *ES-FoMo III: 3rd Workshop on Efficient*
 723 *Systems for Foundation Models*.

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

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

730 Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao.
 731 React: Synergizing reasoning and acting in language models. In *International Conference on*
 732 *Learning Representations (ICLR)*, 2023b.

733 Jiajie Zhang, Nianyi Lin, Lei Hou, Ling Feng, and Juanzi Li. Adaptthink: Reasoning models can
 734 learn when to think. *arXiv preprint arXiv:2505.13417*, 2025a.

735 Shengjia Zhang, Junjie Wu, Jiawei Chen, Changwang Zhang, Xingyu Lou, Wangchunshu Zhou,
 736 Sheng Zhou, Can Wang, and Jun Wang. Othink-r1: Intrinsic fast/slow thinking mode switching
 737 for over-reasoning mitigation. *arXiv preprint arXiv:2506.02397*, 2025b.

738 Wencheng Zhang, Shiqin Qiao, Lingjie Luo, Yinfeng Li, Chuanyang Zheng, Qian Xu, Meng Li,
 739 Yong Gui, Yijun He, Jianing Qiu, et al. Synapseroute: An auto-route switching framework on
 740 dual-state large language model. *arXiv preprint arXiv:2507.02822*, 2025c.

741 Yao Zhao, Zhitian Xie, Chen Liang, Chenyi Zhuang, and Jinjie Gu. Lookahead: An inference
 742 acceleration framework for large language model with lossless generation accuracy. In *Proceed-
 743 ings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD
 744 '24, pp. 6344–6355. Association for Computing Machinery, 2024. ISBN 9798400704901. doi:
 745 10.1145/3637528.3671614.

752

753

754

755

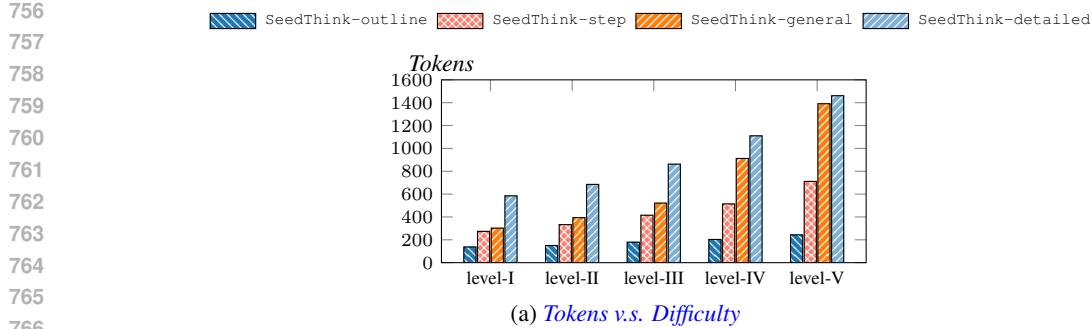


Figure 5: Different seed prompt and it's generated seed length of Qwen3-14B on Math500 dataset.

A THE USE OF LARGE LANGUAGE MODELS (LLMs)

Large Language Models (LLMs) were employed solely for the purpose of enhancing the linguistic clarity and stylistic refinement of this manuscript.

B SUPPLEMENTARY MATERIALS FOR SECTION 3.2

As described in Section 3.2, we conduct a pilot study using the MATH500 datasets to analyze the impact of seed granularity on the performance of [two widely used LRM](#)s [Qwen3-8B](#) and [DeepSeek-R1-Distill-LLaMA-8B](#). For seed thought generation, we use the No-Thinking mode with four different prompt to control the seed granularity. For the second round of generation, we use the thinking mode while providing the seed as an additional reference through the thinking with seed prompt. Both the seed generation prompt and the thinking with seed prompt can be found in C.

We use an 8K context window for seed thought generation and 16K context window for thinking with seed reasoning process. Following the recommendations to balance diversity and coherence in the official model cards, we set the temperature to 0.6 for Thinking mode and 0.7 for No-Thinking mode, respectively.

The generated seed length of four different SeedThink variants are reported in 5.

C DETAILS ON PROMPT

Outline seed generation prompt

<|User|>You are a reasoning assistant. Your job is to break down a complex problem into 2 to 4 high-level reasoning steps. Focus only on outlining the general approach or strategy. Do not include any numbers, formulas, or final answers. Avoid specific calculations or details|only describe the logic behind solving the problem. Please break down the following problem. [question]
<|Assistant|><think></think> \n

key steps seed generation prompt

<|User|>You are a reasoning assistant. Your task is to rapidly provide key steps for solving the given math problem. Include the main reasoning steps without performing detailed calculations. Mention any important formulas or concepts needed, but do not compute numerical results or give the final answer. [question]
<|Assistant|><think></think> \n

General approach seed generation prompt

810
811
812
813
814
815
816
817

<|User|>You are a reasoning assistant. Quickly provide a general approach to solve the given math problem. Include a step-by-step plan with brief explanations of each step. You may mention formulas and concepts, but avoid extensive calculations. The goal is to set up the problem for detailed reasoning later. [question]
<|Assistant|><think></think> \n

818 Detailed steps seed generation prompt

819
820
821
822
823
824
825
826
827

<|User|>You are a reasoning assistant. Your task is to promptly provide detailed steps for solving the given math problem. Include necessary formulas, initial setups, and perform simple calculations if needed. However, leave the most complex calculations or reasoning for the next stage. Provide as much detail as possible without giving the final answer. [question]
<|Assistant|><think></think> \n

828 Thinking with seed generation prompt

829
830
831
832
833
834
835
836
837
838

<|User|> Reference Seed solution from a instruct model:
[Reference answer]
It is crucial to critically evaluate the information provided in the reference, recognizing that it may be biased or incorrect. If you think the reference is incorrect, try to correct it to become your answer.
[question]
<|Assistant|><think>\n

839 D DIFFICULTY ESTIMATOR AS A TEXT CLASSIFICATION TASK

840
841
842
843
844
845
846
847

To implement this classifier, we employ a RoBERTa model (Liu et al., 2019). This choice is motivated by RoBERTa’s well-established and outstanding performance on a wide range of text classification benchmarks, where it consistently delivers robust and accurate predictions. Furthermore, compared to the large reasoning models that are the focus of our study, a RoBERTa-based classifier is relatively lightweight, allowing for efficient inference without introducing significant computational burden. This combination of high classification accuracy and operational efficiency makes it an ideal candidate for our difficulty estimation step.

848
849
850
851
852
853
854
855
856
857

We formulate difficulty estimation as a 3-class classification task. Specifically, we train the model on a mathematical dataset comprising 12,500 samples with inherent difficulty annotations. The original fine-grained labels (on a scale of 1-5) are mapped to three broader categories: labels 1-2 are grouped as *easy*, labels 3-4 as *moderate*, and label 5 as *hard*. This categorization captures meaningful distinctions in problem complexity while maintaining sufficient data for each class.

858
859
860
861
862

Our implementation uses the RoBERTa-base architecture, which contains 125 million parameters configured with 12 transformer layers, a hidden state size of 768, and 12 attention heads. The model is trained for 10 epochs with a batch size of 16. This demonstrates the model’s strong capability in reliably assessing problem difficulty.

863

The analysis below provide a comprehensive breakdown of the estimator’s performance on the Math500 dataset. It shows that even when the estimator misclassifies a problem, it does not lead to a catastrophic failure. Instead, it results in a predictable and often acceptable trade-off between computational cost and solution accuracy. As the table3 shows, we can analyze the two distinct types of misclassification scenarios.

The first type occurs when a **harder problem is misclassified as easier** (e.g., a True Hard problem predicted as Medium). In this case, the model receives a seed that is more detailed than opti-

864 Table 3: A comprehensive breakdown of the estimator’s performance on the Math500 dataset.
865

866 Label	867 Predicted: Easy			868 Predicted: Medium			869 Predicted: Hard		
	870 Cnt.	871 Acc.	872 Length	873 Cnt.	874 Acc.	875 Length	876 Cnt.	877 Acc.	878 Length
879 Easy	880 125	881 96.8%	882 1,206	883 7	884 100.0%	885 1,676	886 1	887 100.0%	888 2,853
889 Medium	890 30	891 90.0%	892 1,950	893 191	894 97.9%	895 2,067	896 12	897 100.0%	898 2,222
899 Hard	900 2	901 50.0%	902 3,105	903 26	904 84.6%	905 3,705	906 106	907 92.5%	908 5,257

871 mal. While this can lead to a decrease in accuracy—for instance, ‘Hard’ problems misclassified
872 as ‘Medium’ see their accuracy drop from 92.5% to 84.6%—this is coupled with a **corresponding**
873 **reduction in generation length**. The average generation length for these same problems plummets
874 from **5258 to 3705 tokens**. This outcome represents a clear trade-off: a modest reduction in
875 accuracy for a substantial improvement in computational speed.

876 The second type of error is when an **easier problem is misclassified as more difficult** (e.g., a True
877 Easy problem predicted as Medium). Here, the model is given a higher-level, more abstract seed
878 than necessary. This prompts a more cautious reasoning process, which our data shows **actually**
879 **increases accuracy**. All those misclassified types achieve **100% accuracy**, an improvement over
880 the 96.8% for those classified correctly. This gain in robustness comes at the expected cost of effi-
881 ciency, with the average token length increasing from 1206 to 1676. This scenario again illustrates
882 a trade-off: sacrificing efficiency for a gain in accuracy.

883 In summary, The vast majority of samples (422/500, or 84.4%) are classified correctly, achieving a
884 high accuracy of **96.2%**. On the 78 misclassified instances, performance remains strong at **89.7%**
885 accuracy.

886 E ABLATION STUDY ON SEED LENGTH

887 To investigate whether longer or more abstract seeds lead to saturation in speculative gains, we
888 conducted additional experiments. As shown in Table 4, rather than observing saturation, we find
889 that **longer seeds generally increase speculative gains** (speed up from **1.47 to 1.64** and MAT from
890 **1.65 to 1.96**), as they produce more opportunities for suffix matches. **We do not observe saturation**
891 **within the studied range**, and we note that seed length is intentionally constrained: excessively long
892 seeds would undermine the two-stage pipeline’s efficiency. Within this realistic range, speculative
893 gains (Mean accept tokens) grow monotonically.

894 Table 4: Speculative gains (using SAM decoding) of different seed prompt Qwen3-14B on the
895 Math500 dataset. MAT short for Mean Accept Token.

901 Seeds	902 MAT	903 Throughput (tokens/s)	904 Speed up from SD	905 Length	906 Accuracy
907 Think (w/o SAM)	908 0	909 28.44	910 –	911 4,741	912 96.8%
913 Think (w/ SAM)	914 1.65	915 40.26	916 $\times 1.42$	917 4,776	918 96.8%
919 SeedThink (outline seed)	920 1.73	921 41.72	922 $\times 1.47$	923 2,959	924 93.8%
925 SeedThink (key_steps seed)	926 1.82	927 43.88	928 $\times 1.54$	929 2,847	930 93.4%
931 SeedThink (general_approach seed)	932 1.88	933 45.04	934 $\times 1.58$	935 2,735	936 95.6%
937 SeedThink (detailed_steps seed)	938 1.96	939 46.87	940 $\times 1.64$	941 2,650	942 94.4%

911 F TOKEN-LEVEL ALIGNMENT: n -GRAM PRESERVATION UNDER 912 SEED-GUIDED REASONING

913 We provide a theoretical justification for why n -grams appearing in the seed sequence tend to re-
914 appear with high probability in the final reasoning trace. The key assumption is that the change of
915 prompt template between seed generation and reasoning introduces only a small perturbation in the
916 model’s input representation.

918 F.1 PRELIMINARIES
919

920 Let the seed sequence be

921
$$s = (s_1, s_2, \dots, s_k),$$

922 generated by the model M under a seed template. The final reasoning sequence is

923
$$r = (r_1, r_2, \dots),$$

925 generated by the same model M but conditioned on (q, s) .926 For any prefix u , denote the next-token distributions by

927
$$p_s(\cdot | u), \quad p_r(\cdot | u),$$

929 corresponding respectively to the seed-generation context and the final-reasoning context. We define
930 the stepwise total variation distance

931
$$\varepsilon_t(u) = \|p_s(\cdot | u) - p_r(\cdot | u)\|_{\text{TV}}.$$

932 We further define its expectation with respect to prefixes encountered during reasoning:

933
$$\bar{\varepsilon}_t = \mathbb{E}_{r_{<t} \sim P_r} [\varepsilon_t(r_{<t})].$$

935 **Assumption (Template Stability).** The difference between the seed template and the reasoning
936 template induces at most a small perturbation δ in the model's internal embeddings. Together with
937 the Lipschitz continuity of the model's logit mapping and the stability of the softmax function, this
938 implies the existence of a constant ε such that

939
$$\bar{\varepsilon}_t \leq \varepsilon \quad \text{for all } t.$$

941 F.2 MAIN LEMMA: n -GRAM ALIGNMENT AT MATCHED POSITIONS
942943 **Lemma 1.** For any n -gram $g = s_{i:i+n-1}$ appearing in the seed, the probability that the final
944 reasoning sequence reproduces the same n -gram at the same position satisfies

945
$$\Pr[r_{i:i+n-1} = g] \geq \prod_{t=i}^{i+n-1} (1 - \bar{\varepsilon}_t).$$

948 Under the uniform bound $\bar{\varepsilon}_t \leq \varepsilon$, this becomes

949
$$\Pr[r_{i:i+n-1} = g] \geq (1 - \varepsilon)^n \geq 1 - n\varepsilon.$$

951 *Proof.* Condition on the event that $r_{i:t-1} = s_{i:t-1}$. Under this condition, the total variation distance
952 between $p_s(\cdot | r_{<t})$ and $p_r(\cdot | r_{<t})$ is at most $\varepsilon_t(r_{<t})$. By the optimal coupling characterization of
953 total variation distance,

955
$$\Pr[r_t = s_t | r_{i:t-1} = s_{i:t-1}] \geq 1 - \varepsilon_t(r_{<t}).$$

956 Taking expectation over possible prefixes and applying the chain rule yields

957
$$\Pr[r_{i:i+n-1} = g] \geq \prod_{t=i}^{i+n-1} (1 - \bar{\varepsilon}_t).$$

960 The uniform bound follows immediately. □962 F.3 COROLLARY: APPEARANCE AT ANY POSITION
963964 Let $J \subseteq \{1, 2, \dots, T\}$ be any candidate set of positions. By applying a union bound to Lemma 1,
965 we obtain:966 **Corollary 1.** For any n -gram g in the seed,

967
$$\Pr[\exists j \in J : r_{j:j+n-1} = g] \geq 1 - \sum_{j \in J} \left(1 - \prod_{t=j}^{j+n-1} (1 - \bar{\varepsilon}_t)\right).$$

970 Under the assumption $\bar{\varepsilon}_t \leq \varepsilon$,

971
$$\Pr[\exists j \in J : r_{j:j+n-1} = g] \geq 1 - |J|(1 - (1 - \varepsilon)^n) \approx 1 - |J|n\varepsilon.$$

972
973

F.4 INTERPRETATION

974 Under the Template Stability Assumption—which bounds the embedding perturbation δ between the
975 seed-generation and reasoning-generation contexts—the resulting bound on stepwise total variation
976 distance guarantees that short and medium-length n -grams in the seed are preserved with high prob-
977 ability in the model’s subsequent reasoning. The alignment arises directly from the shared model
978 parameters and the strong causal dependence between the seed and the final reasoning trace.979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025