SpecCoT: Accelerating Chain-of-Thought Reasoning through Speculative Exploration

Junhan Shi[1], **Yijia Zhu**[3], **Zhenning Shi**[1], **Dan Zhao**[2], **Qing Li**[2]*, **Yong Jiang**[1,2]*

[1] Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, China
[2] Peng Cheng Laboratory, Shenzhen, China [3] Xidian University, Xian, China
{shijh23,shizn23}@mails.tsinghua.edu.cn
zhuyijia@stu.xidian.edu.cn, {zhaod01, liq}@pcl.ac.cn
jiangy@sz.tsinghua.edu.cn

Abstract

Large Reasoning Models (LRMs) demonstrate strong performance on complex tasks through chain-of-thought (CoT) reasoning.However, they suffer from high inference latency due to lengthy reasoning chains. In this paper, we propose SpecCoT, a collaborative framework that combines large and small models for effective yet efficient reasoning. Unlike traditional speculative decoding, which operates at the token level, SpecCoT adopts a step-level verification strategy: the large model first establishes the reasoning direction, and for each intermediate step, the small model generates multiple candidate drafts in parallel. The large model then verifies these drafts, either selecting the most suitable one or rejecting them all and generating its own. SpecCoT approach balances reasoning quality with inference efficiency through fine-grained model cooperation. Experiments across diverse tasks show SpecCoT reduces inference latency by 1.7-4.1× while maintaining comparable accuracy to standard large model inference.

1 Introduction

Recent advances in Large Reasoning Models (LRMs) (Jaech et al., 2024; Team, 2025; DeepSeek-AI et al., 2025) have transformed AI reasoning through chain-of-thought (CoT) (Wei et al., 2022a) mechanisms. By decomposing complex problems into intermediate steps, these models achieve state-of-the-art performance across various domains. Unlike traditional Large Language Models, LRMs first generate and refine step-by-step reasoning before producing conclusions. However, this process incurs substantial computational costs—lengthy reasoning chains combined with billions of parameters result in inference latency ranging from minutes to hours, rendering these powerful systems impractical for time-sensitive applications.

Current approaches address this challenge by either developing smaller reasoning models through distillation techniques (Wang et al., 2025b) or reducing CoT generation length using methods like StepSkip (Liu et al., 2024) and TokenSkip (Xia et al., 2025). However, these approaches face a fundamental limitation when reasoning models encounter problems of varying complexity: large models are computationally inefficient for simple tasks, while smaller models or compressed reasoning chains often fail to handle complex problems that require detailed step-by-step deduction. This critical trade-off between computational efficiency and reasoning thoroughness presents a key open problem and motivates further exploration of adaptive reasoning frameworks. (Xia et al., 2025; Zhang et al., 2025; Liu et al., 2024).

The success of speculative decoding (Leviathan et al., 2023; Chen et al., 2023; Xia et al., 2024) illustrates how collaboration between large and small models can effectively reduce inference latency. In this paradigm, a small model rapidly drafts potential continuations while a large model verifies their quality, achieving significant speedup in text generation (Ryu and Kim, 2024). This collaborative approach is particularly well-suited for chain-of-thought reasoning due to its structural characteristics—unlike standard text generation, where tokens contribute equally, reasoning chains comprise distinct steps of varying complexity and importance, from crucial logical deductions to straightforward elaborations. This inherent structure suggests an opportunity to enhance the speculative paradigm by leveraging semantic-level verification rather than requiring exact token matches.

In this paper, we propose SpecCoT (Speculative Chain-of-Thought), a collaborative reasoning framework that improves efficiency by combining small-model draft generation with large-model step-level verification. Instead of compressing models or simplifying reasoning chains, SpecCoT

* Corresponding authors.

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 24405–24415
November 4-9, 2025

employs a strategic multi-phase process: initially, a large model generates the opening tokens to establish the correct reasoning trajectory; subsequently, for each intermediate step, a lightweight small model efficiently produces multiple candidate continuations in parallel. The large model then evaluates these drafts simultaneously, either accepting the most suitable candidate or intervening to generate the step itself when none are acceptable. This design preserves reasoning accuracy while improving efficiency by delegating intermediate step generation to the small model and reserving the large model for only validation and critical reasoning.

We evaluate SpecCoT on diverse reasoning tasks spanning different complexity levels. Our experiments demonstrate that SpecCoT reduces inference latency by up to 4.1× compared to standard large model inference while maintaining comparable reasoning accuracy. The efficiency gains are particularly significant for complex mathematical problems requiring long reasoning chains, where our method effectively balances computational cost and reasoning quality. Our key contributions are:

- We propose SpecCoT, a collaborative reasoning framework that strategically combines large and small models to achieve an optimal balance between reasoning quality and computational efficiency.
- We develop a parallel verification mechanism with a dynamic fallback strategy, enabling efficient evaluation of multiple reasoning candidates while preserving reasoning accuracy through selective large model intervention.
- Experiments across reasoning benchmarks demonstrate SpecCoT reduces inference latency by 1.7-4.1× while maintaining comparable accuracy, with stronger gains on complex problems requiring extensive reasoning.

2 Motivation

Complex reasoning tasks present unique challenges and opportunities for language models of different sizes. In this section, we explore the inherent trade-offs between models of varying scales and identify key opportunities for optimization.

2.1 Model Size vs. Reasoning Efficiency

Small models generate content faster but with reduced reasoning reliability. Small language models offer substantial computational efficiency advantages, particularly in generation speed. However, this efficiency advantage does not necessarily translate to faster overall reasoning time. While small models can generate tokens 3-4× faster than large models, their limited reasoning capabilities lead to significantly more verbose solutions, requiring up to 4× more tokens(Table 1). This inefficiency in token usage ultimately results in similar end-to-end latency compared to larger models. More critically, the increased token consumption often reflects underlying reasoning issues, where small models struggle to maintain coherent logic and frequently fall into circular reasoning patterns.

Table 1: Performance and efficiency tradeoffs in DeepSeek R1 distilled Qwen models in GSM8K.

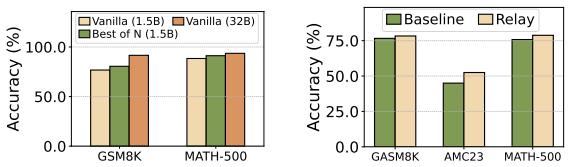
Distill-Qwen	**1.5B**	**7B**	**14B**	**32B**
Accuracy (%)	75.51	87.49	90.83	91.58
Avg Time (s)	10.94	5.20	6.86	10.33
Avg Token	2736.46	974.19	758.51	634.11
Speed (token/s)	256.38	200.41	120.39	67.90

Large models demonstrate superior reasoning capabilities and token efficiency. While operating at a slower token generation pace, large language models showcase advanced reasoning abilities. As observed in Table 1, within the same model family, larger models not only achieve higher accuracy (91% vs 75%) but also produce more concise and effective reasoning paths, requiring less than a quarter of the tokens needed by small models for the same tasks. This efficiency suggests an ability to maintain a more direct and consistent logical trajectory.

2.2 Collaborative Reasoning Principles

Complex reasoning can be decomposed into subtasks of varying difficulty. The complexity of reasoning tasks is not uniformly distributed across all steps. While some pivotal stages require sophisticated problem-solving abilities, such as initial problem analysis or solution strategy formulation, many intermediate steps involve more straightforward operations. Small models, despite their limitations in solving end-to-end complex problems, can effectively handle these intermediate steps, including basic calculations, logical deductions, and case analyses. This non-uniform distribution of reasoning complexity suggests opportunities for efficient model deployment.

Reasoning quality depends on logical correctness rather than exact expression. Unlike tasks requiring precise wording or stylistic consistency,



(a) Performance of Best-of-N on GSM8K and MATH-500.

(b) Impact of initial guidance on reasoning accuracy.

Figure 1: Performance analysis of Best-of-N sampling and initial guidance impact on reasoning tasks.

reasoning tasks prioritize the advancement of logical insights over exact token matching. This is fundamentally different from most speculative decoding techniques, where small models must precisely replicate the output of larger models. In reasoning, multiple valid expressions can advance the same logical step, making the process naturally suited for collaboration between models of different capabilities. This characteristic enables a flexible approach where small models can contribute effectively to the reasoning process despite variations in their specific outputs.

2.3 Optimization Potential for Small Models

High-quality initial guidance impacts subsequent reasoning. The initial formulation of a reasoning approach plays a crucial role in determining the overall solution quality. A well-structured initial analysis can effectively decompose complex problems into more manageable steps and establish a clear reasoning framework. More importantly, high-quality initial guidance helps prevent error accumulation in the reasoning chain - early mistakes in problem interpretation or strategy selection often cascade into more significant errors in later steps. This observation suggests that leveraging more capable models at the beginning of the reasoning process can provide substantial benefits for the entire solution chain.

Parallel candidate generation enables efficient exploration. While individual generations from small models might lack reliability, generating multiple candidates in parallel significantly increases the likelihood of obtaining high-quality reasoning steps. The computational overhead of this multi-candidate generation remains manageable with optimization of inference frameworks such as vLLM(Kwon et al., 2023). This parallel exploration of the solution space helps overcome the limitations of small models in generating high-quality reasoning steps.

3 Method

We propose Speculative Chain-of-Thought (SpecCoT), a collaborative reasoning framework that leverages the complementary strengths of large and small language models. The key insight of our approach is to decompose the chain-of-thought reasoning process into speculative iterations, where a large model provides initial guidance and performs efficient verification, while a small model generates multiple diverse intermediate reasoning steps throughout the process. This collaborative approach enables us to maintain the reasoning quality and reflective capabilities of large models while achieving significant computational efficiency improvements through parallel candidate generation.

3.1 Formulation

Given an input problem x, SpecCoT aims to generate a chain of reasoning steps $C = \{c_1, c_2, ..., c_n\}$ to derive the final answer y. The reasoning process starts with an initial guidance g generated by the large model to set a strong foundation. For each subsequent step i, the small model generates N candidate continuations $\{\hat{c}_i^j\}_{j=1}^N$ based on previous reasoning steps $c_{1:i-1}$. The large model then serves as a verifier V that selects the most promising continuation or indicates the need for its own generation when no candidate meets the quality threshold. This reasoning process is formally formulated as:

$$x \xrightarrow{C} y, \quad \text{where } C = \{g\} \cup \{c_1, c_2, ..., c_n\}. \quad (1)$$

This formulation reflects a key observation: while the large model's guidance and verification are crucial for maintaining reasoning quality, delegating the intermediate generation steps to a small model with parallel candidate generation can significantly reduce computational overhead.

3.2 Initial Guidance

SpecCoT begins by employing a large target model M_T to generate an initial guidance g with k tokens conditioned on the input problem. This process can be formally expressed as:

$$g = M_T(x)[k], \quad (2)$$

where g serves as the foundation for the entire reasoning chain. This design choice is motivated by our observation that a strong initial direction is crucial for the overall reasoning trajectory. The target model's superior reasoning capabilities help establish a logically sound and contextually appropriate

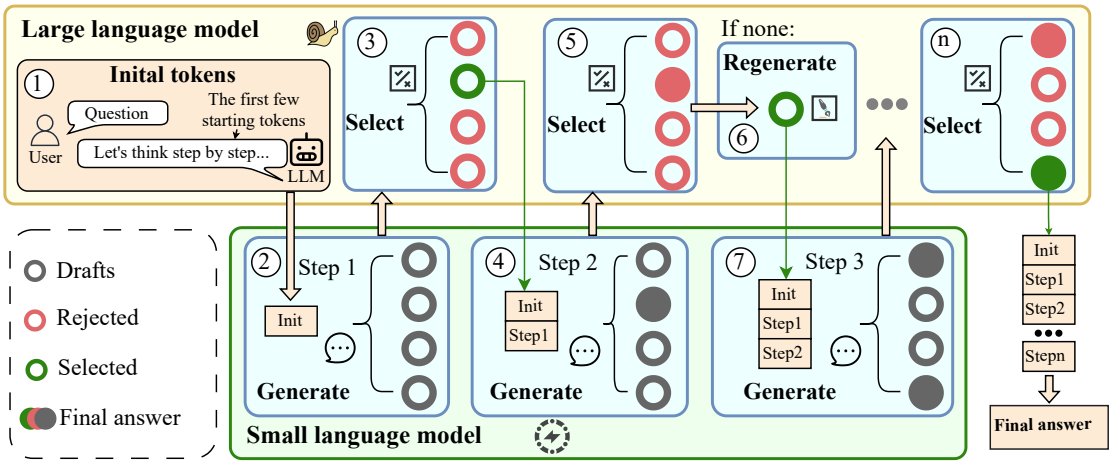


Figure 2: Overview of SpecCoT. The process begins with a LLM generating initial guidance tokens (1), followed by a small language model generating multiple candidate continuations for each reasoning step (2,4,7). For each step, the LLM verifies and selects the most promising candidate (3,5,n) or regenerates the content itself if no candidate meets quality standards (6). This collaborative approach combines the reasoning quality of large models with the computational efficiency of small models, culminating in a final answer after multiple reasoning steps.

starting point, which effectively constrains the solution space for subsequent generations by the draft model while minimizing computational overhead.

3.3 Speculative Exploration

Following the initial guidance, SpecCoT employs an efficient exploration strategy at each reasoning step i to advance the solution process. We leverage a lightweight draft model M_D to generate multiple candidate continuations simultaneously:

$$\{\hat{c}_i^j\}_{j=1}^N = M_D(g, c_{1:i-1}), \tag{3}$$

where N represents the number of parallel candidates generated based on the initial guidance g and previous reasoning steps $c_{1:i-1}$.

The parallel generation approach addresses the inherent limitations of smaller models while exploiting modern inference engines' optimization for batch sampling, where generating N samples does not incur a linear increase in computation time. While any single generation might fall short of the target model's quality, the diversity across multiple candidates substantially increases the probability of producing at least one high-quality continuation. We employ a moderate sampling temperature during generation to balance exploration diversity with logical coherence.

3.4 Efficient Verification and Fallback

We introduce a computationally efficient verification mechanism that evaluates all candidate continuations through a single forward pass of the target model. By augmenting the candidate set with an explicit reject option, the target model can either select a promising continuation or indicate the need for its own generation:

$$s = V_T(\{\hat{c}_i^j\}_{j=1}^N \cup \{\texttt{reject}\}), \tag{4}$$

where V_T represents the verification function that assesses both logical coherence and reasoning progress of each candidate. The implementation is highly efficient—requiring only a single forward pass with the output of a single token indicating the selected choice.

The reject option serves as a critical quality control mechanism. When all candidate continuations contain logical errors or fail to meaningfully advance the reasoning process, the target model rejects them and takes over generation:

$$c_i = M_T(g, c_{1:i-1}). \tag{5}$$

This verification and fallback approach creates a robust framework that maintains reasoning quality while minimizing computational costs. The system leverages the draft model for efficient exploration of the solution space, while the target model provides oversight through selective intervention. This strategic deployment of the target model's capabilities—precisely when they are most needed-establishes an effective balance between computational efficiency and reasoning reliability.

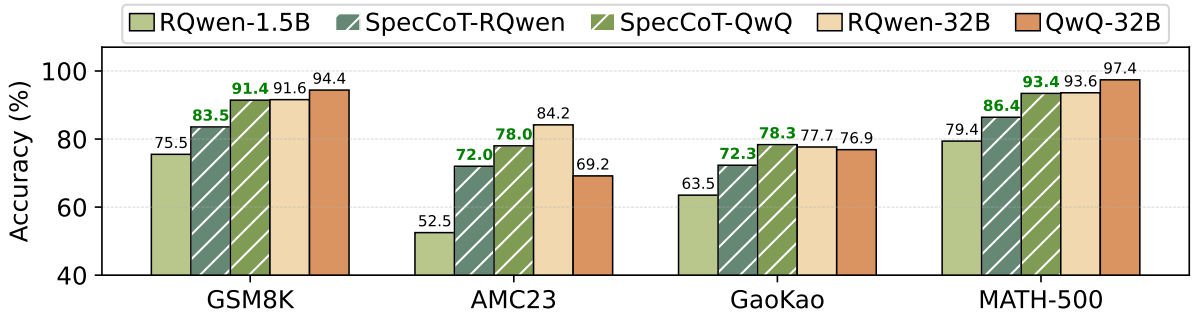


Figure 3: Accuracy comparison (%) of SpecCoT variants against baseline models across four reasoning benchmarks. SpecCoT implementations use Deepseek-R1-Distill-Qwen-1.5B (RQwen-1.5B) as the draft model, with either QwQ-32B or Deepseek-R1-Distill-Qwen-32B (RQwen-32B) as the target model.

3.5 Relation to Speculative Decoding

Our SpecCoT framework shares a fundamental synergy with speculative decoding techniques through their common use of draft models for efficient generation. While speculative decoding operates at the token level, SpecCoT extends this speculation paradigm to higher-level reasoning steps. This architectural alignment suggests the potential for a unified speculation framework that operates across different granularities. By utilizing the same draft model for both token-level speculation and reasoning step generation, we can achieve a hierarchical efficiency optimization while maintaining the target model's quality assurance through verification.

4 Experiments

4.1 Setup

We evaluate SpecCoT using Deepseek-R1-Distill-Qwen-1.5B (DeepSeek-AI et al., 2025) as the draft model and QwQ-32B (Team, 2025) and Deepseek-R1-Distill-Qwen-32B as the target models. Both model series demonstrate strong chain-of-thought reasoning capabilities. For brevity, we refer to the Deepseek-R1-Distill-Qwen series models as RQwen in the following sections. Our evaluation spans reasoning datasets of varying difficulty levels, including GSM8K (Cobbe et al., 2021) with basic multi-step arithmetic problems, MATH-500 (Hendrycks et al., 2021) containing standard high school mathematics questions, GaoKao-En-2023 (Liao et al., 2024) featuring Chinese college entrance examination problems, and AMC23 (MAA, 2023) with challenging competition mathematics. This selection encompasses a comprehensive spectrum of mathematical complexity. All experiments were conducted on four NVIDIA A100 GPUs using vLLM (Kwon et al., 2023) as the inference engine, with five different runs per configuration to ensure statistical reliability. The max reasoning token budget is set to be 8192 with a temperature of 0.6. We establish baselines by running vanilla inference using both the draft model and the target model independently.

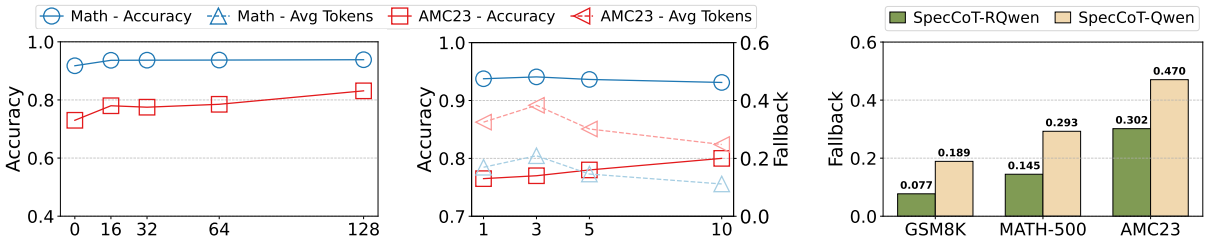
4.2 Main Results

Accuracy Improvement. Figure 3 illustrates SpecCoT's accuracy across four mathematical reasoning datasets. The results reveal that SpecCoT consistently and substantially outperforms the draft model across all benchmarks, particularly on GSM8K and MATH-500, where it approaches the performance level of the target model. Interestingly, on AMC23 and GaoKao datasets, QwQ-32B baseline performs slightly below RQwen and SpecCoT due to its need for additional tokens to complete solutions. Nevertheless, QwQ-32B exhibits remarkable reasoning capabilities—when deployed as the target model within our framework, it generally yields superior results compared to configurations using RQwen-32B. These performance gains stem from the step-wise guidance and verification mechanism provided by the target model, which effectively identifies and corrects potential errors from the draft model before they propagate through the reasoning chain.

Lower Latency. SpecCoT delivers remarkable improvements in computational efficiency across all evaluated datasets. When paired with QwQ-32B as the target model, our approach achieves approximately 3× reduction in inference time on both GSM8K and MATH-500 benchmarks. While the efficiency gains naturally diminish on more challenging tasks, SpecCoT still maintains impressive speedup factors of 2.3× and 2.0× on GaoKao and AMC23 respectively. This gradual decrease in ac-

Table 2: **Performance across four mathematical reasoning tasks (GSM8K, MATH-500, GaoKao, AMC23).** Metrics include average tokens, latency (Lat, in seconds), and improvement ratio r of SpecCoT over baselines.

Model	GSM8K			MATH-500			GaoKao			AMC23		
	Token	Lat	r	Token	Lat	r	Token	Lat	r	Token	Lat	r
RQwen-1.5B	1096.8	4.5	–	2609.3	10.6	–	3456.6	14.4	–	5298.8	21.64	–
QwQ-32B	2053.8	33.5	–	3086.8	50.8	–	4102.0	68.0	–	5885.8	97.78	–
SpecCoT(ours)	1023.3	10.1	3.3	1495.9	16.2	3.1	2316.5	30.1	2.3	3633.0	48.6	2.0
RQwen-32B	574.3	9.3	–	1998.6	32.6	–	2833.4	46.7	–	4605.5	76.2	–
SpecCoT(ours)	239.1	2.9	3.2	636.9	7.8	4.1	1674.7	24.0	1.9	3031.6	44.7	1.7



(a) Impact of initial guidance token count on model accuracy across datasets.

(b) Relationship between number of drafts and accuracy/fallback rate.

(c) Comparison of fallback rates across different reasoning datasets.

Figure 4: Ablation studies and sensitivity analysis of SpecCoT performance.

celeration ratios aligns with our expectations, as more complex problems often require additional verification steps from the target model. Similarly, when using RQwen-32B as the target model, we observe speedups ranging from 4.1× on MATH-500 to 1.7× on AMC23. This performance pattern across tasks of increasing mathematical complexity demonstrates how our approach effectively balances computational efficiency with the enhanced reasoning demands of more difficult problems.

Token Efficiency. SpecCoT demonstrates remarkable reductions in token consumption across all datasets. When compared with QwQ-32B as the target model, our approach reduces token counts by 50.2% on GSM8K (from 2053.8 to 1023.3) and 51.5% on MATH-500 (from 3086.8 to 1495.9). The efficiency gains are even more pronounced when using RQwen-32B configurations, where we observe token reductions of 58.4% on GSM8K and 68.1% on MATH-500. This substantial decrease in token usage directly contributes to the latency improvements discussed earlier, as fewer tokens require less computational processing. The token efficiency stems from our verification mechanism, which allows the target model to select optimal reasoning paths early in the process, effectively preventing both the wandering exploration characteristic of smaller models and the verbose reasoning typical of larger models. This guided approach produces more concise solutions while maintaining or even improving accuracy.

4.3 Ablations

Number of initial tokens. The number of initial tokens refers to the number of guiding tokens generated by the target model when initiating the collaborative generation process. When this number is set to zero, the target model does not generate any guiding tokens at the outset; instead, the draft model directly generates n possible intermediate reasoning steps. Our experimental results demonstrate that the presence of guiding tokens led to improved accuracy across all three datasets, albeit with a slight increase in inference time. However, this increased inference time remains substantially lower than that of the base model. We attribute the accuracy improvement to the high-quality initial guidance provided by the target model, which prevents the draft model from making early mistakes and effectively steers its generation process.

Number of Drafts. The number of drafts parameter controls how many candidate continuations the draft model generates for each reasoning step. As illustrated in Fig. 4b, increasing this parameter yields interesting performance dynamics. While additional candidates do not guarantee accuracy improvements, we observe a substantial decrease in the fallback rate, particularly when the number reaches 10. This indicates that with more candidates, the target model is more likely to find acceptable continuations without needing to generate its own. This pattern reveals that while multiple drafts cannot fundamentally enhance the draft model's

Table 3: **SpecCoT performance on MATH-500 and GSM8K.** Metrics include accuracy (%), tokens, time (s), and fallback rate (%) using two draft models (RQwen-1.5B and Qwen2.5-1.5B-Instruct) with three 32B target models (RQwen for RQwen-32B, Qwen-Inst. for Qwen2.5-32B-Instruct, and QwQ for QwQ-32B).

Draft Model	Metric	MATH-500			GSM8K		
		RQwen	Qwen-Inst.	QwQ	RQwen	Qwen-Inst.	QwQ
RQwen-1.5B	Accuracy	83.6	80.7	91.4	86.8	83.5	93.5
	Tokens	242.2	265.1	1021.9	638.6	319.3	1495.9
	Time	2.9	2.9	9.8	7.5	3.2	16.2
	Fallback	8.6	2.5	6.5	16.4	2.7	14.5
Qwen2.5-1.5B Instruct	Accuracy	75.7	77.0	81.8	68.5	67.0	74.4
	Tokens	254.1	247.3	347.8	324.8	292.2	536.3
	Time	5.5	2.9	5.0	6.3	3.4	8.6
	Fallback	5.6	4.7	18.9	13.5	4.9	29.3

problem-solving capabilities beyond its inherent limitations, they effectively improve the probability of producing at least one continuation that meets the target model's standards. However, for especially complex reasoning junctures that exceed the draft model's capabilities, even a larger candidate pool may not provide sufficient quality, requiring the target model to intervene.

4.4 Analysis

4.4.1 Impact of Model Capabilities

To analyze how model capabilities affect collaboration, we tested various combinations of draft and target models. Target models ranged from basic instruction-following (Qwen-32B-Instruct) to enhanced reasoning (RQwen-32B) and sophisticated chain-of-thought reasoning (QwQ-32B), while draft models included reasoning-enhanced RQwen-1.5B and standard Qwen2.5-1.5B. Results show stronger target models achieve higher accuracy but require more computational resources, with QwQ-32B reaching 93.5% accuracy but consuming 1495.9 tokens on average. When paired with RQwen-1.5B as the draft model, even basic Qwen-32B-Instruct maintains good efficiency (265.1 tokens, 2.9s inference time) while achieving 80.7% accuracy. Draft model capability significantly impacts performance - RQwen-1.5B enables notably higher accuracy across all target models (91.4% vs 81.8% with QwQ-32B on MATH-500) with lower fallback rates (6.5% vs 18.9%). However, even with weaker Qwen2.5-1.5B drafts, QwQ-32B still achieves 74.4% accuracy on GSM8K, demonstrating the robustness of our collaborative approach.

4.4.2 Analysis of Fallback Rate

The fallback rate represents the proportion of the draft model's intermediate steps rejected by the target model. On relatively simpler datasets like GSM8K, we observe low fallback rates of approximately 0.08, indicating that the draft model produces acceptable steps for most reasoning stages(Fig. 4c). However, this rate increases dramatically to 0.47 on challenging benchmarks like AMC23, reflecting the draft model's diminished capability to generate reliable continuations for complex problems. Our analysis reveals that dataset difficulty serves as the primary determinant of fallback rates. While increasing the number of draft candidates moderately improves acceptance, this effect is less pronounced than the impact of inherent problem complexity. Interestingly, we find that stronger target models tend to exhibit higher fallback rates, as their enhanced reasoning capabilities enable more stringent evaluation of the draft model's proposals.

4.4.3 Impact of resource configurations

To examine SpecCoT's performance under different resource configurations, we evaluated the performance using vLLM with 2-way and 4-way tensor parallelism on 2 and 4 A100 GPUs, respectively. The results demonstrate consistent performance improvements across both datasets. In terms of inference time, SpecCoT achieves the fastest processing speed in both hardware settings, completing inference in 12.49s and 10.08s on GSM8K, and 20.55s and 16.17s on MATH-500 under 2-way and 4-way configurations. The throughput analysis reveals that while all models benefit from increased parallelism, the efficiency gains vary. Large models like RQwen-32B and QwQ-32B show substantial throughput improvements (>50%) when scaling from 2 to 4 GPUs, while SpecCoT exhibits moderate gains of 23.52% on GSM8K and 24.95% on MATH-500. This difference is attributed to the fact that larger target models benefit more from multi-

Table 4: **Inference efficiency comparison on GSM8K and MATH-500.** Time (seconds), throughput (tokens/second), and their improvements when scaling from 2×A100 to 4×A100 GPU configurations.

Dataset	Model	2×A100 Time (s)	4×A100 Time (s)	Time Speedup (%)	2×A100 Token/s	4×A100 Token/s	Throughput Gain (%)
GSM8K	RQwen-1.5B	12.51	10.94	12.55%	236.23	256.38	8.53%
	RQwen-32B	14.71	10.33	29.78%	44.56	67.90	52.38%
	QwQ-32B	51.86	33.57	35.27%	41.17	63.31	53.78%
	SpecCoT(ours)	**12.49**	**10.08**	19.30%	82.17	101.50	23.52%
MATH-500	RQwen-1.5B	34.28	33.13	3.35%	233.25	244.82	4.96%
	RQwen-32B	69.20	43.96	36.47%	39.88	61.59	54.44%
	QwQ-32B	81.54	53.96	33.83%	40.55	62.03	52.97%
	SpecCoT(ours)	**20.55**	**16.17**	21.31%	74.00	92.46	24.95%

GPU tensor parallelism compared to the smaller draft model used in SpecCoT.

5 Related Works

5.1 Efficient Reasoning

Recent LLMs have adopted chain of thought (CoT) reasoning(Wei et al., 2022b), enhancing problem-solving capabilities while introducing longer outputs and increased computational costs. Research on improving reasoning efficiency follows two main approaches: length compression methods like TokenSkip(Xia et al., 2025), SoftCoT(Xu et al., 2025), and Compressed CoT(Cheng and Van Durme, 2024) reduce verbose outputs while maintaining quality; early termination techniques such as Dynasor(Fu et al., 2025) and NoThinking(Ma et al., 2025) optimize reasoning by identifying when to stop processing. These advances enable LLM applications in resource-constrained environments without sacrificing quality. Several recent works combine speculative decoding with CoT reasoning, including SCoT(Wang et al., 2025a) using LoRA-tuned draft models, SpecReason(Pan et al., 2025) decomposing CoT into discrete speculative steps, and Speculative Thinking(Yang et al., 2025) triggering large model intervention at critical junctures identified by reflection keywords.

Our approach differs fundamentally from these concurrent works in two critical aspects. First, unlike these methods, which allow the small model to initiate reasoning, we use the large model to generate initial tokens, preventing the small model from starting on a flawed reasoning path. Second, rather than operating on complete reasoning trajectories like SCoT, our approach applies best-of-n selection and fallback at intermediate CoT steps, enabling the large model to guide the reasoning process at multiple points. This ongoing involvement allows for earlier intervention when the small model shows signs of error, leading to more accurate reasoning while maintaining efficiency.

5.2 Speculative Decoding

Speculative decoding mitigates inference latency in autoregressive language models by enabling parallel token generation without compromising output quality. Inspired by speculative execution in computing (Burton, 1985), seminal works by Leviathan et al.(Leviathan et al., 2023) and Chen et al.(Chen et al., 2023) demonstrated this approach's effectiveness. Implementation strategies vary widely: from leveraging smaller models(Leviathan et al., 2023; Chen et al., 2023; Spector and Re, 2023) and target model components(Cai et al., 2023; Zhang et al., 2024) to utilizing n-gram tables(Fu et al., 2024) and retrieval systems(He et al., 2023). Verification techniques have advanced from basic token-level checks(Leviathan et al., 2023) to sophisticated tree-structured methods(Miao et al., 2024). Recent breakthroughs include feature-level processing in EAGLE(Li et al., 2024b) and adaptive draft trees in EAGLE-2(Li et al., 2024a), delivering enhanced acceleration while preserving or improving quality.

6 Conclusion

In this paper, we presented SpecCoT, demonstrating that efficient chain-of-thought reasoning can be achieved through strategic collaboration between large and small models. Our approach achieves 1.7-4.1× reduction in reasoning time while maintaining strong reasoning capabilities across various benchmarks.

Limitations

While SpecCoT demonstrates significant efficiency improvements, it does not yet achieve lossless reasoning compared to using the target model alone.

As our current approach is training-free, incorporating task-specific training and better leveraging the target model's reflection capabilities could potentially bridge this performance gap in future work. Additionally, our framework currently operates independently from token-level speculative decoding. A more integrated system combining both step-level and token-level speculation through shared draft models could yield further acceleration benefits, representing an important direction for future research.

Acknowledgments

This work is supported by the Shenzhen R&D Program under grant NO. KJZD20230923114059020, and in part by Shenzhen Key Laboratory of Software Defined Networking under Grant ZDSYS20140509172959989.

References

F Warren Burton. 1985. Speculative computation, parallelism, and functional programming. *IEEE Transactions on Computers*, 100(12):1190–1193.

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, and Tri Dao. 2023. Medusa: Simple framework for accelerating llm generation with multiple decoding heads. *Retrieved December*, 3:2024.

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

Jeffrey Cheng and Benjamin Van Durme. 2024. Compressed chain of thought: Efficient reasoning through dense representations. *arXiv preprint arXiv: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*.

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

Yichao Fu, Peter Bailis, Ion Stoica, and Hao Zhang. 2024. Break the sequential dependency of llm inference using lookahead decoding. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

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*.

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

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

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. 2024. Openai o1 system card.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. 2023. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 19274–19286. PMLR.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2024a. Eagle-2: Faster inference of language models with dynamic draft trees. *arXiv preprint arXiv:2406.16858*.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. 2024b. Eagle: speculative sampling requires rethinking feature uncertainty. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Minpeng Liao, Chengxi Li, Wei Luo, Wu Jing, and Kai Fan. 2024. Mario: Math reasoning with code interpreter output-a reproducible pipeline. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 905–924.

Tengxiao Liu, Qipeng Guo, Xiangkun Hu, Cheng Jiayang, Yue Zhang, Xipeng Qiu, and Zheng Zhang. 2024. Can language models learn to skip steps? *arXiv preprint arXiv:2411.01855*.

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

MAA. 2023. American mathematics competition 2023 dataset. https://www.maa.org/math-competitions/amc-resources.

Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, Chunan Shi, Zhuoming Chen, Daiyaan Arfeen, Reyna Abhyankar, and Zhihao Jia. 2024. Specinfer: Accelerating large language model serving with tree-based speculative inference and verification. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*, ASPLOS '24, page 932–949, New York, NY, USA. Association for Computing Machinery.

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

Hyun Ryu and Eric Kim. 2024. Closer look at efficient inference methods: A survey of speculative decoding. *arXiv preprint arXiv:2411.13157*.

Benjamin Spector and Chris Re. 2023. Accelerating llm inference with staged speculative decoding. *arXiv preprint arXiv:2308.04623*.

Qwen Team. 2025. Qwq-32b: Embracing the power of reinforcement learning.

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

Junxiong Wang, Wen-Ding Li, Daniele Paliotta, Daniel Ritter, Alexander M. Rush, and Tri Dao. 2025b. M1: Towards scalable test-time compute with mamba reasoning models. *arXiv preprint arXiv:2504.10449*.

Peiyi Wang, Lei Li, Zhihong Shao, RX Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2023. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *arXiv preprint arXiv:2312.08935*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022a. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022b. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

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

Heming Xia, Zhe Yang, Qingxiu Dong, Peiyi Wang, Yongqi Li, Tao Ge, Tianyu Liu, Wenjie Li, and Zhifang Sui. 2024. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. *arXiv preprint arXiv:2401.07851*.

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. 2025. Softcot: Soft chain-of-thought for efficient reasoning with llms. *arXiv preprint arXiv:2502.12134*.

Wang Yang, Xiang Yue, Vipin Chaudhary, and Xiaotian Han. 2025. Speculative thinking: Enhancing small-model reasoning with large model guidance at inference time. *arXiv preprint arXiv:2504.12329*.

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

Jun Zhang, Jue Wang, Huan Li, Lidan Shou, Ke Chen, Gang Chen, and Sharad Mehrotra. 2024. Draft & verify: Lossless large language model acceleration via self-speculative decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11263–11282, Bangkok, Thailand. Association for Computational Linguistics.

A Appendix

A.1 Analysis of Step-level Quality

To provide a more detailed analysis of the intermediate steps, we evaluated their quality using a Process Reward Model (PRM) (Wang et al., 2023), a 7B model trained to assess mathematical reasoning steps. As shown in Table 5, for each reasoning step, the target model effectively selects a higher-quality continuation from the multiple drafts generated by the small model. The improvement is more pronounced on complex tasks like AMC23 (+6.8%), where the diversity from parallel sampling becomes crucial for finding an acceptable reasoning path.

Furthermore, we analyzed the occurrence of reflective tokens (e.g., "wait", "but", "hmm"). We found that the large model's interventions (fallback steps) contain a higher frequency of such keywords compared to the small model's drafts. This suggests that the large model intervenes precisely at points requiring deeper, more cautious reasoning, validating our framework's ability to strategically leverage model strengths at the step level.

Table 5: Step quality analysis using a Process Reward Model (PRM). The target model consistently selects higher-quality steps from the generated drafts.

Dataset	Draft Quality	Selected Quality
GSM8K	0.8000	0.8232
MATH	0.7093	0.7463
AMC23	0.5716	0.6107

A.2 Scalability with Batch Size

To address scalability concerns, we evaluated SpecCoT's performance across various batch sizes on 2×H100 GPUs, using problems from GSM8K. The results in Table 6 demonstrate that SpecCoT scales effectively, achieving an optimal wall-time speedup of 2.45× at a batch size of 128. This scalability benefits from SpecCoT's design, where the lightweight draft model handles most of the decoding workload. This reduces the computational burden on the large model in batch scenarios, enabling better resource utilization as the batch size increases and suggesting that the approach can be deployed effectively in high-throughput environments.

Table 6: SpecCoT scalability across different batch sizes on GSM8K, evaluated on 2×H100 GPUs.

Batch Size	Wall Time (s)	Throughput (tokens/s)	Speedup
1	416.47	65.42	1.00×
8	276.67	88.37	1.51×
32	324.10	96.97	1.29×
64	254.64	130.18	1.64×
128	170.16	154.02	2.45×

A.3 Synergy with Token-level Speculative Decoding

Furthermore, SpecCoT is complementary to vanilla speculative decoding (SD). While SpecCoT operates at the semantic step level, SD works at the token level. Combining SpecCoT with vLLM's speculative decoding yields additional speedups, reducing latency on GSM8K from 3.33s to 3.01s (a 9.6% speedup) and on MATH-500 from 15.68s to 13.07s (a 16.6% speedup). These results demonstrate that the two techniques can be effectively integrated for hierarchical optimization.