

Large Reasoning Models Know How to Think Efficiently

Zeyu Xing¹ Xing Li² HuiLing Zhen² Xianzhi Yu² Mingxuan Yuan² Sinno Jialin Pan¹

Abstract

Large Reasoning Models (LRMs) exhibit potential in problem-solving through extended Chain-of-Thought (CoT) generation, enhancing robustness and accuracy by iteratively revising user prompts. However, excessive CoT generation poses challenges in LLM inference, as prolonged decoding due to redundant tokens creates computational bottlenecks. This paper introduces two training-free self-thinking methods—Pre-judged Reasoning and Fallback Reasoning—which optimize inference efficiency via dynamic selection of fast thinking and reasoning strategies based on LRMs’ intrinsic task complexity classification capabilities. Evaluations on the MATH500 and AIME24 dataset demonstrate that Pre-judged Reasoning reduces token generation by up to 26.6% compared to slow reasoning without compromising accuracy. Similarly, Fallback Reasoning achieves a reduction of up to 24.0% in generated tokens, enabling significantly faster task completion. Both methods substantially reduce computational overhead while retaining the accuracy of LRMs.

1. Introduction

The reasoning capability of Large Reasoning Models (LRMs) is becoming increasingly important in solving real-world tasks. However, not all tasks require strict step-by-step logical reasoning. For some simpler tasks where conclusions can be directly drawn, issues such as excessive reflection and repeated attempts encountered by LRMs can lead to the generation of a large number of ineffective tokens, resulting in wasted computational resources and user costs. Therefore, effectively assessing the task’s difficulty and selecting an appropriate approach to reasoning has be-

¹Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong SAR, China ²Huawei Noah’s Ark Lab, Hong Kong SAR, China. Correspondence to: Zeyu Xing <zeyuxing@link.cuhk.edu.hk>.

come a critical issue to reduce LRM deployment cost and user waiting time.

For example, Table 1 compares the performance of step-by-step reasoning and fast thinking on the GSM8k dataset (Cobbe et al., 2021). In this case, reasoning produces nearly identical performance to fast thinking but at the cost of generating almost four times as many tokens. This observation necessitates the need of hybrid reasoning to automatically switch between slow and fast thinking based on problem complexity to better trade-off accuracy and computational cost of LRMs.

Metric	Fast Thinking	Reasoning
Accuracy	0.94	0.95
#Average Tokens	221	829 (3.75×)

Table 1. Comparison of efficiency and effectiveness of the fast thinking and reasoning of Qwen3-32B on the GSM8k dataset.

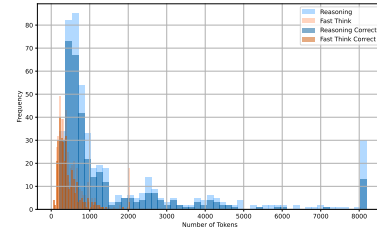


Figure 1. Distribution of generated token counts and their correctness rates of DeepSeek-R1-Distill-Qwen-14B on MATH-500 with the fast thinking and reasoning modes. Most responses are short in both thinking mode. The accuracy decreases noticeably over generation length in the reasoning mode.

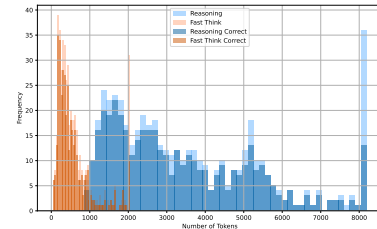


Figure 2. Distribution of generated token counts and their correctness rates of Qwen3-32B on MATH-500 with the fast thinking and reasoning modes. The accuracy is stably high over different complexity, except the ones over the limited token budgets. These two models have different reasoning patterns.

Figures 1 and 2 illustrate the frequency distributions of token generation lengths and their corresponding correctness rates for DeepSeek-R1-Distill-Qwen-14B and Qwen3-32B, respectively, on the MATH-500 dataset. Both models exhibit a

substantially higher mean token count in the reasoning mode compared to the fast thinking mode. Notably, fast thinking demonstrates elevated confidence and accuracy for simple questions requiring concise answers; however, accuracy declines precipitously beyond a token-length threshold, which correlates with increased problem complexity. Furthermore, the correctness distributions of the two models in reasoning mode diverge significantly on the same dataset, reflecting inherent disparities in their performance and cognitive architectures. Qwen3-32B generates extended chain-of-thought (CoT) reasoning across problems of varying difficulty, maintaining stable accuracy except for highly complex cases. In contrast, DeepSeek-R1-Distill-Qwen-14B predominantly produces abbreviated responses and struggles with intricate problem-solving. As a result, training based hybrid thinking methods not only alter the LRM capability but also introduce additional fine-tuning cost for each LRM (Jiang et al., 2025; Huang et al., 2025; Liang et al., 2025; Zhang et al., 2025). These observations underscore the non-trivial challenge of designing a **generalizable and efficient hybrid reasoning framework for LRMs that accommodates diverse capabilities and cognitive patterns**.

Typically, the development of prominent open-source LRMs, such as Deepseek-R1 (DeepSeek-AI, 2025) and Qwen3 (Yang et al., 2025), comprises two distinct training phases. Initially, a foundational LLM is trained using conventional next-token prediction tasks. Subsequently, this base model undergoes fine-tuning via reinforcement learning techniques to enhance its reasoning capabilities. Consequently, LRMs inherently retain the fast thinking capabilities from their underlying LLM architecture and weights that are not modified significantly (Team et al., 2025; Wu et al., 2025). **LRMs should know how to think efficiently by themselves, but the integration of contextually appropriate external stimuli remains indispensable**. For example, it is feasible to prompt LRMs to utilize their fast thinking skills by employing minor adjustments during the generation phase. A commonly effective method involves the insertion of an empty `<think></think>` block within the generation prompt to disable thinking, which typically suffices to activate fast-thinking functionality of most LRMs.

In this work, we present **two novel training-free methods for automatic, efficient self-thinking to improve inference efficiency using the inherent problem complexity recognition capability of LRMs**. The first method, *Pre-judged Reasoning*, leverages the self-assessment capability of the LRM to determine whether a task requires reasoning or can be solved with fast thinking. The second method, *Fall-back Reasoning*, dynamically switches to reasoning when the token limit for fast thinking is reached, thereby ensuring efficient computation without compromising on accuracy. Our experimental results demonstrate that these training-free methods reduce the number of generated tokens by up

to 26.6%, maintaining high accuracy, offering a promising opportunity for optimizing LRM inference in real-world applications.

2. Related Works

Efficient reasoning. Apart from improving token-level decoding efficiency with improved operators (Dao et al., 2022), quantization (Xiao et al., 2023a; Liu et al., 2024; Li et al., 2025), sparsity (Xiao et al., 2023b; Zhang et al., 2024; Yuan et al., 2025; Sun et al., 2024a), and speculative decoding (Leviathan et al., 2023; Sun et al., 2024b; Li et al., 2024), there are also recent works on directly reducing thinking length or LRM thinking cost with long2short model merging (Team et al., 2025; Wu et al., 2025), thinking compression (Sun et al., 2024c; Lin et al., 2025), and large-small model collaboration for speculative thoughts (Pan et al., 2025; Wang et al., 2025).

Concurrent hybrid reasoning. Qwen3 models offer prompt-based hybrid reasoning and the option to specify limited thinking budgets (Yang et al., 2025). In which, Qwen3 models do not switch between fast thinking and reasoning but require the external user guidance, which introduce additional complexity for users. In addition, there are also many training based methods to perform hybrid thinking of queries of varying difficulty and types (Jiang et al., 2025; Huang et al., 2025; Liang et al., 2025; Zhang et al., 2025). The fine-tuning of LRMs may lead to alterations in model capabilities or catastrophic forgetting. Existing approaches that introduce additional small classification modules not only incur extra training and deployment overheads, but also require separate training of these modules for LRMs with different capability levels or even distinct checkpoints.

In contrast, our self-thinking methodology eliminates the need for explicit training procedures by fully leveraging the intrinsic problem complexity recognition capacity inherent in LRMs. This fundamental characteristic endows our approach with superior generalization capabilities and significantly reduces deployment complexity compared to conventional solutions.

3. Method

In this section, we present two innovative training-free approaches for efficient self-thinking, which harness the intrinsic capabilities of LRMs to assess problem complexity and adaptively switch between fast thinking and reasoning. These methods enable the dynamic and automatic selection of reasoning strategies based on the self-assessment of task complexity.

In contrast to hybrid reasoning paradigms that depend on fine-tuned Learning and Reasoning Models (LRMs) or aux-

iliary smaller models, our methodology directly harnesses the intrinsic decision-making capabilities of LRMs themselves. Within a properly implemented inference system such as vLLM (Kwon et al., 2023) and SGLang (Zheng et al., 2024), Key-Value (KV) caching mechanisms from preliminary pre-judgment and fast thinking phases can be systematically reused as prefix cache, thereby enhancing both memory utilization and computational efficiency. This self-assessment framework aligns with the principle that LRMs should exhibit accurate self-awareness of their operational competencies, mitigating estimation biases introduced by external models.

3.1. Method 1: Pre-judged Reasoning

The principal concept behind Pre-judged Reasoning is to leverage the self-assessment capability of LRMs to gauge the difficulty of a given task autonomously. The model’s intrinsic understanding of its capabilities allows it to make more accurate judgments about whether a task requires reasoning or can be solved efficiently through fast thinking, compared to external or smaller-scale evaluators.

To operationalize this concept, we initially present the LRM with a prompt instructing it to quickly evaluate whether the given task necessitates comprehensive step-by-step reasoning (termed *slow thinking*) or can be effectively resolved via fast thinking. The LRM is explicitly instructed to produce a minimal JSON-formatted response containing solely a boolean flag `require_slow_thinking`, without explanations or intermediate reasoning. This approach reduces computational overhead and streamlines the decision-making process. Once we receive the JSON response, we parse it to extract the flag’s boolean value. If `true`, we invoke the reasoning module with an expanded token budget of *tokens_reasoning* tokens, suitable for detailed, intricate reasoning steps. Conversely, if `false`, the fast thinking capability is employed with a constrained token budget of *tokens_fast_thinking* tokens, optimizing efficiency.

The procedure diagram of the *Pre-judged Reasoning* method is presented in figure 3, and the prompts can be found in Appendix A.1.

3.2. Method 2: Fallback Reasoning

Fallback Reasoning is predicated upon the intuition that the complexity of a question correlates positively with the number of output tokens required for its resolution. Specifically, if an LRM’s fast thinking mechanism exhausts a predefined modest token budget without completing the response, the task is likely too complex to handle efficiently through only fast thinking.

Initially, we instruct the LRM to solve the given question through fast thinking, strictly limiting the token generation

to *tokens_trial* tokens. We then monitor whether the LRM’s response is prematurely terminated due to hitting this token limit. If the response is successfully completed within the *tokens_trial* token budget, the provided answer is accepted as sufficiently reliable and directly returned as the final output of self-thinking.

However, if the token limit halts the generation prematurely, indicating insufficient completion of the task, we subsequently engage reasoning. During this phase, we explicitly provide the model with the incomplete attempt as contextual reference, expanding the allowed token budget substantially to *tokens_reasoning* tokens, thereby accommodating thorough step-by-step reasoning.

The procedure diagram of the *Fallback Reasoning* method is summarized in Figure 3, and the prompts can be found in Appendix A.2.

4. Experimental Results

Our experiments are conducted using the HuggingFaceH4/MATH-500 dataset (HuggingFaceH4, 2024), a benchmark specifically designed to evaluate the mathematical reasoning capabilities of large-scale models and the HuggingFaceH4/aime_2024 dataset (HuggingFaceH4, 2025), a dataset consists of 30 problems from the 2024 AIME(American Invitational Mathematics Examination) I and AIME II. We select two prominent open-source LRMs for evaluation: DeepSeek-R1-Distill-Qwen-14B(in FP16) (DeepSeek-AI, 2025), denoted as DeepSeek-R1-14B, and Qwen3-32B(in INT8 quantization) (Yang et al., 2025).

We utilize Ollama¹ as the inference engine for deploying these LRMs, which provides optimized inference capabilities specifically suited for large-scale language and reasoning models. Experiments are executed on a computational platform comprising 2× NVIDIA RTX 4090 GPUs, each equipped with 24 GiB of VRAM. The CPU used is an Intel Core i9-13900, complemented by 128 GiB of RAM.

4.1. Accuracy and Average Numbers of Tokens Generated on Two Methods

We use accuracy and average generated token count as the primary metrics to evaluate the effectiveness of the proposed methods. These metrics allow us to assess the trade-off between reasoning efficiency and model performance. In terms of hyperparameters, we set *tokens_reasoning* to 8192, *tokens_fast_thinking* to 2048, as most questions can be resolved by fast thinking within this token budget. Additionally, we set *tokens_trial* to 512, which gives the token limit during the initial trial stage.

Table 2 summarizes the experimental performance compari-

¹<https://ollama.com/>

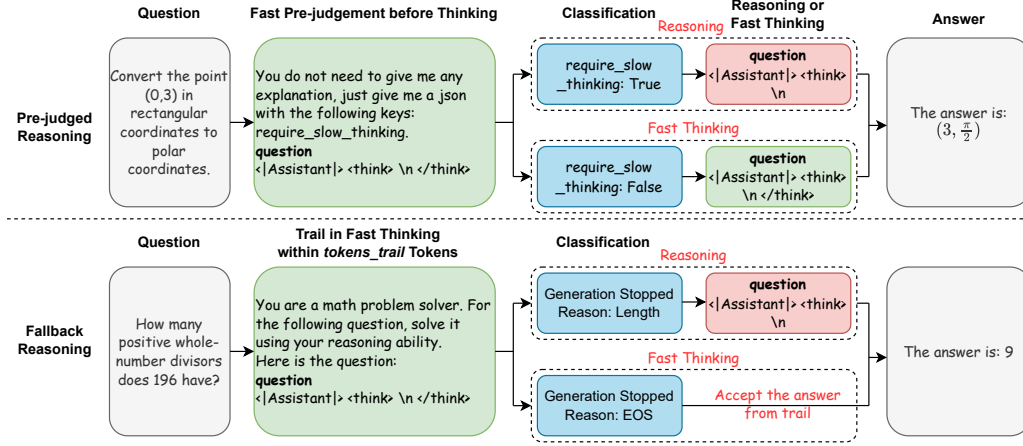


Figure 3. Procedure diagram of the proposed training-free self-thinking approaches *Pre-judged Reasoning* and *Fallback Reasoning*, which harness the intrinsic capabilities of LRMs to self-assess problem complexity for automatic thinking mode switching.

Metrics	GSM8K		MATH500		AIME24	
	Deepseek-R1-14B	Qwen3-32B	Deepseek-R1-14B	Qwen3-32B	Deepseek-R1-14B	Qwen3-32B
Fast Thinking Accuracy / #Tokens	0.722 / 169	0.931 / 147	0.594 / 375	0.610 / 392	0.100 / 1054	0.167 / 1576
Reasoning Accuracy / #Tokens	0.408 / 474	0.933 / 754	0.720 / 1981	0.834 / 3605	0.167 / 6164	0.433 / 7040
Pre-judged Reasoning Accuracy / #Tokens	0.632 / 280 -40.9%	0.933 / 437 -42.0%	0.708 / 1455 -26.6%	0.826 / 3072 -14.8%	0.167 / 5589 -9.33%	0.433 / 7040 0%
Fallback Reasoning Accuracy / #Tokens	0.726 / 166 -65.0%	0.934 / 149 -80.2%	0.710 / 1555 -21.5%	0.804 / 2740 -24.0%	0.233 / 5018 -18.59%	0.400 / 7000 -43.57%

Table 2. Comparison of fast thinking, reasoning, and the proposed self-thinking methods on the MATH500 dataset. For the average token count of Fallback Reasoning, it includes the combined tokens from both the fast thinking trial stage and the reasoning stage. For Pre-judged Reasoning, the average token count reflects only the tokens used during the reasoning stage, as the pre-judgment process can be efficiently integrated during the pre-filling phase by appending an additional head, thereby minimizing computational overhead in a well-implemented system.

son between fast thinking, reasoning, and the proposed pre-judged reasoning and fallback reasoning approaches using Deepseek-R1-14B and Qwen3-32B models on the GSM8K, MATH500 and AIME24 datasets. Metrics include accuracy, average token usage, frequency of triggering reasoning, and cases in which detailed reasoning significantly improves correctness. Results indicate that on MATH500 dataset, Pre-judged Reasoning and Fallback Reasoning efficiently balance accuracy and computational cost by selectively employing detailed reasoning only for challenging questions. They can achieve comparable accuracy to pure reasoning, while significantly generating 14.8% to 26.6% less tokens on average. An example illustrating the effectiveness of Pre-judged Reasoning can be found in Section B.

On the harder AIME24 dataset, the results indicate that both Pre-judged and Fallback Reasoning significantly improve accuracy over standard slow reasoning while reducing average token usage for Deepseek-R1-14B, demonstrating enhanced efficiency on challenging questions. Specifically, Fallback Reasoning improves accuracy by 6.6% and thinking efficiency by 18.59% simultaneously. However, for Qwen3-32B, the substantial relative performance gap between reasoning and fast thinking necessitates step-by-step slow reasoning in all tasks, limiting efficiency gains from automatic thinking mode switching. In such case, our method reliably defaults to slow reasoning to preserve complex problem-solving performance of LRMs, adhering to expected behavior given the higher priority of accuracy.

On the easier GSM8k dataset, Fast Thinking outperforms Reasoning for Deepseek-R1-14B, this may be due to the distillation process of the model. However, Fallback Reasoning still preserves comparable performance to the higher one, while shortening the sequence length comparable to Fast Thinking. For Qwen3-32B, Pre-judged Reasoning and Fallback Reasoning both preserve the performance as Reasoning, while improving the efficiency by 42.0% and 80.2% respectively.

A more detailed table showing the frequency that the reasoning is triggered can be found in appendix C.

5. Conclusion

This paper introduces two innovative training-free self-thinking methods, Pre-judged Reasoning and Fallback Reasoning, which improve the computational efficiency of Large Reasoning Models (LRMs) by adaptively choosing the most suitable reasoning strategy. By leveraging the inherent capabilities of LRMs to assess problem complexity, these methods ensure that tasks requiring detailed reasoning are handled with appropriate computational resources, while simpler tasks benefit from faster inference. Experimental results on the MATH500, AIME24 and GSM8K datasets show that both methods achieve noticeable reduction in generated token numbers while maintaining high accuracy, offering an effective solution to optimize LRM performance and resource usage in complex problem-solving scenarios.

References

- Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., and Schulman, J. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.
- Dao, T., Fu, D., Ermon, S., Rudra, A., and Ré, C. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.
- DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL <https://arxiv.org/abs/2501.12948>.
- Huang, X., Vangani, T. K., Liu, Z., Zou, B., and Aw, A. T. Adacot: Rethinking cross-lingual factual reasoning through adaptive chain-of-thought, 2025. URL <https://arxiv.org/abs/2501.16154>.
- HuggingFaceH4. Huggingfaceh4/math-500 · datasets at hugging face, Nov 2024. URL <https://huggingface.co/datasets/HuggingFaceH4/MATH-500>.
- HuggingFaceH4. Huggingfaceh4/aime_2024 · datasets at hugging face, Jan 2025. URL https://huggingface.co/datasets/HuggingFaceH4/aime_2024.
- Jiang, L., Wu, X., Huang, S., Dong, Q., Chi, Z., Dong, L., Zhang, X., Lv, T., Cui, L., and Wei, F. Think only when you need with large hybrid-reasoning models. *arXiv preprint arXiv:2505.14631*, 2025.
- Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J., Zhang, H., and Stoica, I. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023.
- Leviathan, Y., Kalman, M., and Matias, Y. Fast inference from transformers via speculative decoding, 2023. URL <https://arxiv.org/abs/2211.17192>.
- Li, X., Xing, Z., Li, Y., Qu, L., Zhen, H.-L., Liu, W., Yao, Y., Pan, S. J., and Yuan, M. Kvtuner: Sensitivity-aware layer-wise mixed precision kv cache quantization for efficient and nearly lossless llm inference, 2025. URL <https://arxiv.org/abs/2502.04420>.
- Li, Y., Wei, F., Zhang, C., and Zhang, H. Eagle: Speculative sampling requires rethinking feature uncertainty. *arXiv preprint arXiv:2401.15077*, 2024.
- Liang, G., Zhong, L., Yang, Z., and Quan, X. Thinkswitcher: When to think hard, when to think fast, 2025. URL <https://arxiv.org/abs/2505.14183>.
- Lin, W., Li, X., Yang, Z., Fu, X., Zhen, H.-L., Wang, Y., Yu, X., Liu, W., Li, X., and Yuan, M. Trimr: Verifier-based training-free thinking compression for efficient test-time scaling, 2025. URL <https://arxiv.org/abs/2505.17155>.
- Liu, Z., Yuan, J., Jin, H., Zhong, S., Xu, Z., Braverman, V., Chen, B., and Hu, X. Kivi: A tuning-free asymmetric 2bit quantization for kv cache. *arXiv preprint arXiv:2402.02750*, 2024.
- Pan, R., Dai, Y., Zhang, Z., Oliaro, G., Jia, Z., and Ne-travali, R. Specreason: Fast and accurate inference-time compute via speculative reasoning. *arXiv preprint arXiv:2504.07891*, 2025.
- Sun, H., Chang, L.-W., Bao, W., Zheng, S., Zheng, N., Liu, X., Dong, H., Chi, Y., and Chen, B. Shadowkv: Kv cache in shadows for high-throughput long-context llm inference. *arXiv preprint arXiv:2410.21465*, 2024a.
- Sun, H., Chen, Z., Yang, X., Tian, Y., and Chen, B. Tri-force: Lossless acceleration of long sequence generation with hierarchical speculative decoding. *arXiv preprint arXiv:2404.11912*, 2024b.
- Sun, H., Haider, M., Zhang, R., Yang, H., Qiu, J., Yin, M., Wang, M., Bartlett, P., and Zanette, A. Fast best-of-n decoding via speculative rejection. *arXiv preprint arXiv:2410.20290*, 2024c.
- Team, K., Du, A., Gao, B., Xing, B., Jiang, C., Chen, C., Li, C., Xiao, C., Du, C., Liao, C., et al. Kimi k1. 5: Scaling reinforcement learning with llms, 2025. URL <https://arxiv.org/abs/2501.12599>, 2025.
- Wang, J., Li, J., Wu, L., and Zhang, M. Efficient reasoning for llms through speculative chain-of-thought. *arXiv preprint arXiv:2504.19095*, 2025.
- Wu, H., Yao, Y., Liu, S., Liu, Z., Fu, X., Han, X., Li, X., Zhen, H.-L., Zhong, T., and Yuan, M. Unlocking efficient long-to-short llm reasoning with model merging, 2025. URL <https://arxiv.org/abs/2503.20641>.
- Xiao, G., Lin, J., Seznec, M., Wu, H., Demouth, J., and Han, S. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pp. 38087–38099. PMLR, 2023a.
- Xiao, G., Tian, Y., Chen, B., Han, S., and Lewis, M. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023b.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., Zheng, C., Liu, D., Zhou, F., Huang, F., Hu, F., Ge, H., Wei, H., Lin, H., Tang, J., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Zhou, J., Lin, J., Dang, K., Bao, K., Yang, K., Yu, L., Deng, L., Li, M., Xue, M., Li, M., Zhang, P., Wang, P., Zhu, Q., Men, R., Gao, R., Liu, S., Luo, S., Li, T., Tang, T., Yin, W., Ren, X., Wang, X., Zhang, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Zhang, Y., Wan, Y., Liu, Y., Wang, Z., Cui, Z., Zhang, Z., Zhou, Z., and Qiu, Z. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Yuan, J., Gao, H., Dai, D., Luo, J., Zhao, L., Zhang, Z., Xie, Z., Wei, Y. X., Wang, L., Xiao, Z., Wang, Y., Ruan, C., Zhang, M., Liang, W., and Zeng, W. Native sparse attention: Hardware-aligned and natively trainable sparse attention, 2025. URL <https://arxiv.org/abs/2502.11089>.

Zhang, X., Ruan, J., Ma, X., Zhu, Y., Zhao, H., Li, H., Chen, J., Zeng, K., and Cai, X. When to continue thinking: Adaptive thinking mode switching for efficient reasoning, 2025. URL <https://arxiv.org/abs/2505.15400>.

Zhang, Z., Sheng, Y., Zhou, T., Chen, T., Zheng, L., Cai, R., Song, Z., Tian, Y., Ré, C., Barrett, C., et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

Zheng, L., Yin, L., Xie, Z., Sun, C., Huang, J., Yu, C. H., Cao, S., Kozyrakis, C., Stoica, I., Gonzalez, J. E., Barrett, C., and Sheng, Y. Sglang: Efficient execution of structured language model programs, 2024. URL <https://arxiv.org/abs/2312.07104>.

A. Appendix: Prompts for Pre-judged reasoning and Fallback Reasoning

A.1. Prompt for Pre-judged Reasoning

Table 3 presents the prompt for Pre-judged Reasoning, specifically designed for DeepSeek-R1-14B. For Qwen3-32b, the prompt is similar, with the only modification being the replacement of the special tokens.

Variable Name	Prompt Template
Prompt for Pre-judge	< User > You are a math problem solver. For the following question, determine if it requires slow thinking or can be solved quickly. You do not need to give me any explanation, just give me a json with the following keys: require_slow_thinking. For example: {"require_slow_thinking": true} Here is the question: question < Assistant > <think> \n </think>
Prompt for Fast Thinking	< User > You are a math problem solver. You are a math problem solver. For the following question, solve it using your direct answering ability. Here is the question: question < Assistant > <think> \n </think>
Prompt for Reasoning	< User > You are a math problem solver. You are a math problem solver. For the following question, solve it using your reasoning ability. Here is the question: question < Assistant > <think>

Table 3. Pre-judge Reasoning Prompts used for DeepSeek-R1-14B

A.2. Prompt for Fallback Reasoning

Table 4 presents the prompt for Fallback Reasoning, specifically designed for DeepSeek-R1-14B. For Qwen3-32b, the prompt is similar, with the only modification being the replacement of the special tokens.

Variable Name	Prompt Template
Prompt for Fast Thinking	< User > You are a math problem solver. You are a math problem solver. For the following question, solve it using your direct answering ability. Here is the question: question < Assistant > <think> \n </think>
Prompt for Reasoning	< User > You are a math problem solver. You are a math problem solver. For the following question, solve it using your reasoning ability. Here is the question: question Here is an example from your previous trial, note that the answer may not be correct, it is just provided for your reference: output of fast thinking < Assistant > <think>

Table 4. Fallback Reasoning Prompts used for DeepSeek-R1-14B

B. An Example of Pre-judged Reasoning

Table 5 provides an example of Pre-judged Reasoning applied to the MATH-500 dataset, using DeepSeek-R1-14B. In this example, the Pre-judgement process correctly identifies the need for slow thinking (reasoning) to solve the problem. We then compared the results of direct thinking and reasoning. As shown, the direct reasoning approach made a computational error, leading to an incorrect answer, while the reasoning approach provided the correct solution. This example demonstrates that LRMs are capable of accurately assessing the need for reasoning intervention based on their own capabilities and the complexity of the task, further validating the effectiveness of Pre-judged Reasoning.

Variable Name	Prompt Template
Question	What is the sum of the digits in the terminating decimal representation of the fraction $\frac{4321}{5^7 \cdot 2^8}$?
Canonical Answer	14
Pre-judgement Output	{"require_slow_thinking": true}
Fast thinking Output	<p>To find the sum of the digits in the terminating decimal representation of the fraction $\frac{4321}{5^7 \cdot 2^8}$, follow these steps:</p> <p>First, recognize that $5^7 \cdot 2^8$ can be rewritten to have a common base for both factors:</p> $5^7 \cdot 2^8 = 2^7 \cdot (2 \cdot 5)^7 = 2^7 \cdot 10^7$ <p>So, the fraction becomes:</p> $\frac{4321}{2^7 \cdot 10^7}$ <p>To convert the fraction into a decimal, adjust it so that the denominator is a power of 10:</p> $\frac{4321}{2^7 \cdot 10^7} = \frac{4321 \div 2^7}{10^7}$ <p>Calculate $2^7 = 128$, so:</p> $\frac{4321}{128} = 33.6875$ <p>Now, place this result over 10^7 to form the decimal:</p> $\frac{33.6875}{10^7} = 0.0000336875$ <p>Identify all the digits in the decimal representation and sum them up:</p> $0 + 0 + 0 + 0 + 0 + 3 + 3 + 6 + 8 + 7 + 5 = 29$ <p>The sum of the digits is 29.</p>
Reasoning Output	<p>First, I need to simplify the fraction $\frac{4321}{5^7 \times 2^8}$.</p> <p>I notice that 5^7 and 2^7 can be paired to form 10^7, which is a power of ten. This will help in converting the fraction into a decimal.</p> <p>So, I rewrite the denominator as $10^7 \times 2$, since $2^8 = 2^7 \times 2$.</p> <p>Next, I simplify the fraction by dividing both the numerator and the denominator by 5^7 to get $\frac{4321}{2}$.</p> <p>Performing the division, $4321 \div 2 = 2160.5$.</p> <p>Now, I convert this decimal into its terminating form: 2160.50000000.</p> <p>Finally, I sum all the digits in this decimal representation:</p> $2 + 1 + 6 + 0 + 5 + 0 + 0 + 0 + 0 + 0 + 0 = 14.$

Table 5. Pre-judged Reasoning Example from DeepSeek-R1-14B

C. Frequency of Reasoning Triggered

Table 6 extends Table 2 with rows Reasoning Improvement Cases, Reasoning Triggered and Reasoning Beneficial.

Metrics	GSM8K		MATH500		AIME24	
	Deepseek-R1-14B	Qwen3-32B	Deepseek-R1-14B	Qwen3-32B	Deepseek-R1-14B	Qwen3-32B
Fast Thinking Accuracy / #Tokens	0.722 / 169	0.931 / 147	0.594 / 375	0.610 / 392	0.100 / 1054	0.167 / 1576
Reasoning Accuracy / #Tokens	0.408 / 474	0.933 / 754	0.720 / 1981	0.834 / 3605	0.167 / 6164	0.433 / 7040
Pre-judged Reasoning Accuracy / #Tokens	0.632 / 280 -40.9%	0.933 / 437 -42.0%	0.708 / 1455 -36.6%	0.826 / 3072 -14.8%	0.167 / 5589 -9.33%	0.433 / 7040 0%
Fallback Reasoning Accuracy / #Tokens	0.726 / 166 -65.0%	0.934 / 149 -80.2%	0.710 / 1555 -21.5%	0.804 / 2740 -24.0%	0.233 / 5018 -18.59%	0.400 / 7000 -8.57%
Reasoning Improvement Cases	36	38	63	112	2	8
Pre-judged Reasoning						
Reasoning Triggered	412	431	234	347	26	30
Reasoning Beneficial	36	21	26	30	2	8
Fallback Reasoning						
Reasoning Triggered	9	3	174	218	27	30
Reasoning Beneficial	1	2	58	99	2	8

Table 6. The *Reasoning Triggered* row indicates when reasoning is activated. The *Reasoning Beneficial* row represents cases where our method yields the correct answer, while Fast Thinking does not.