Less is Not Worse: Effective Reasoning Without Complete Reasoning Chains

Jaehui Hwang **Sangdoo Yun** **Byeongho Heo** **Dongyoon Han**
NAVER AI Lab
{jaehui.hwang, sangdoo.yun, bh.heo, dongyoon.han}@navercorp.com

Abstract

Large language models (LLMs) often produce lengthy reasoning traces with substantial token redundancy. While reasoning processes are generally considered necessary, it has been underexplored whether LLMs truly require the complete trajectory. To investigate, we conduct (1) attention map analysis and (2) targeted lesion studies that remove token groups, both of which show that intermediate tokens contribute minimally to reasoning quality. Our analyses suggest that the most redundant segments typically appear in the middle of reasoning chains, whereas the earlier and later segments are crucial for accurate final outcomes. We argue that this approach avoids redundant intermediate information and exploits the LLM's capability to infer concise and coherent intermediate steps by using the known start and end points. Based on these observations, we propose `MidCut`, a method that removes redundant middle steps during both training and inference. We evaluate `MidCut` in two scenarios for LLM reasoning: (1) supervised fine-tuning (SFT) for reasoning and (2) decoding strategy for a test-time application.

1 Introduction

Recent advancements in large language models (LLMs) have significantly improved performance across a wide range of language-related tasks [1–6]. Despite these improvements, several tasks such as mathematical problem solving and logical reasoning require complex problem-solving processes, making it challenging for LLMs to achieve high accuracy. Large reasoning models (LRMs) have been introduced both by GPT-4o and various open-source initiatives [2, 4, 7] to handle the tasks. LRMs commonly incorporate chain-of-thought (CoT) [8] reasoning capabilities, in which intermediate reasoning steps are produced as a thinking trajectory during the generation process. Recent LRMs separate the internal thinking trajectory from the final answer that users face, enabling models to perform extensive reasoning while presenting only the concise conclusion to users.

Several studies have noted that the thinking trajectories produced by LRMs are typically lengthy and complex [7, 9–11]. Prior work further suggests that LLMs often already know the answer before generating a fully explicit reasoning chain [12]. The NoThinking [9] demonstrates that problems can sometimes be solved without any explicit reasoning process, although with substantially lower performance. These observations indicate that reasoning trajectories are important for solving complex problems, yet the fully explicit trajectory may not always be necessary. This raises a fundamental question: *do LLMs require the complete reasoning trajectory, and if not, which parts are non-essential?*

In this paper, we investigate redundancy in thinking trajectories through two systematic analyses: attention weight patterns and knockouts motivated by prior works [13–16]. We find that answer tokens place little attention on middle steps, and that removing these steps preserves answer quality, in contrast to removing the beginning or end. Motivated by these findings, we propose `MidCut`, a simple approach that trims intermediate steps of thinking trajectories. `MidCut`-SFT leverages trimmed

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Efficient Reasoning.

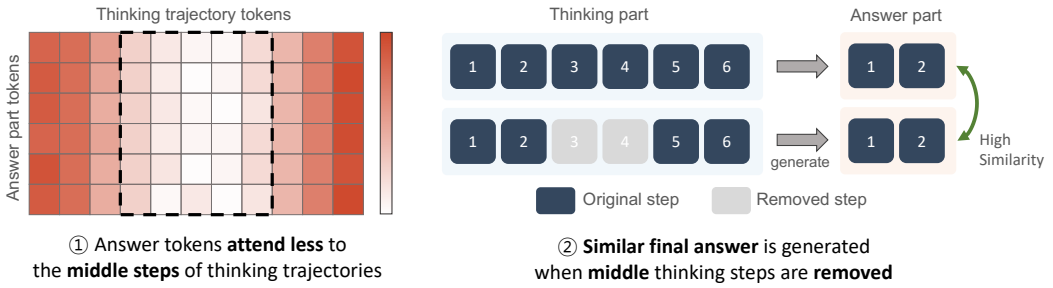

Figure 1: **Illustrative analysis of redundancy in thinking trajectories.** We illustrate our forthcoming analyses in Section 2: (left) attention weights across thinking trajectories show low focus on intermediate tokens; (right) removing them yields similar outputs. Both observations suggest redundancy.

trajectories for more efficient supervised fine-tuning (SFT), while `MidCut`-Decoding skips redundant steps at inference to accelerate reasoning generation. Experiments demonstrate that both strategies improve efficiency while consistently maintaining strong performance across diverse reasoning tasks.

2 Analysis of Thinking Trajectories

This section studies whether certain tokens within thinking trajectories[1] are redundant in real. We analyze thinking trajectories through two complementary approaches: an attention-based analysis of how models process reasoning steps and a knockout-based analysis of how thinking trajectories influence the quality of answer generation. An overview of these analyses is illustrated in Figure 1.

Attention weights analysis. Following prior works [13–16] that interpreted models using attention-based metrics, we analyze attention weights to investigate how different parts of thinking trajectories contribute to answer generation. We expect these patterns to reveal how models prioritize tokens during generation, offering insights into information flow within transformer architectures.

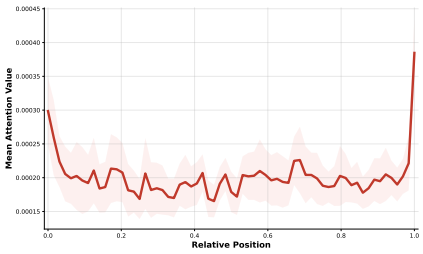

Figure 2: **Averaged 1D attention weights** across thinking trajectories. Intermediate tokens show low scores.

We employ `S1.1-32B` [17] using 30 questions from GPQA-D [18], excluding cases where the model fails to solve the problem or the response exceeds 16K tokens. For each sample, we extract attention weights from answer tokens to thinking tokens across all layers and heads, and then average over layers, heads, and answer tokens to obtain a single distribution over thinking trajectory positions. Since lengths of thinking trajectories vary across samples, we normalize token indices to relative positions before averaging across problems. Figure 2 shows that answer tokens attend strongly to the beginning and ending parts of the trajectory, whereas intermediate steps are weakly attended. This suggests that intermediate steps generally contribute less to answer generation, indicating that these tokens may be redundant within the overall thinking trajectory.

Attention knockout analysis. Based upon the observational insights from attention analysis, we further apply the attention knockout technique [16], which masks attention links and evaluates model performance to probe the importance of specific connections or tokens. We delete specific trajectory segments and generate answers under these conditions to verify again whether thinking trajectories contain redundancy with respect to answer generation. By observing how answer quality changes under such interventions, we can assess whether particular segments play a causal role in generating answers. Our analysis is conducted on 30 problems from the GPQA-D dataset using the `S1.1-32B` model. For each problem, a full thinking trajectory is first generated.

[1]Among two distinct components comprising reasoning outputs - the thinking trajectory (reasoning steps) and the answer part (final solution) - our analysis focuses on the former.

Then, we remove three different segments: the beginning segment (0–20%), an intermediate segment centered in the middle (40–60%), and the ending segment (80–100%). The model is subsequently prompted to generate answers from these truncated trajectories as well as from the full trajectory. Finally, textual similarity between answers from truncated and full trajectories is measured using Jaccard similarity [19] and ROUGE-L [20].

Table 1: **Accuracy and text similarity** after various knockouts of thinking trajectories. We systematically remove different parts and evaluate their impact on answer quality. "Jac. Sim." refers to Jaccard similarity.

Knockout	Jac. Sim.	ROUGE-L
Beginning (0-20%)	0.5174	0.5133
Intermediate (40-60%)	0.5555	0.5528
Ending (80-100%)	0.4906	0.4844

Table 1 shows the average results across multiple cases. In the results, removing the intermediate part consistently yields the highest similarity to answers generated with the full trajectory. In contrast, removing the beginning and ending parts results in lower similarity, suggesting that these segments contain critical information for formulating correct answers. This provides quantitative evidence for our hypothesis that intermediate steps are often redundant and do not substantially contribute to final answer quality.

3 Trimming Reasoning Chains to Enhance Training and Decoding

Based on the knowledge in Section 2, which reveals redundancy in the middle parts of thinking trajectories, we introduce `MidCut`, a method for removing intermediate steps from reasoning trajectories. We present two applications of our `MidCut` methodology: **`MidCut`-SFT** is a preprocessing method for thinking trajectories of LRMs on SFT datasets. This approach removes intermediate parts of the trajectory while preserving the beginning and ending segments, which are believed to contain the essential setup of the problem and formulation of the conclusion. **`MidCut`-Decoding** applies the same principle during the inference phase, truncating intermediate reasoning steps after the thinking phase to reduce computational overhead and generation latency without compromising the quality of final answers. In both applications, our method targets the redundant middle parts identified in our analysis while preserving the critical reasoning components at the trajectory boundaries.

3.1 `MidCut`-SFT: `MidCut` for Supervised Fine-tuning

Method. `MidCut`-SFT follows a simple yet effective approach: given a thinking trajectory from a large reasoning model, we remove the middle section. Specifically, we preserve the first and last segments of the trajectory based on predefined thresholds, either by step count or token count. Steps are defined by splitting trajectories at double newline characters (\n\n), represented by natural reasoning breakpoints.

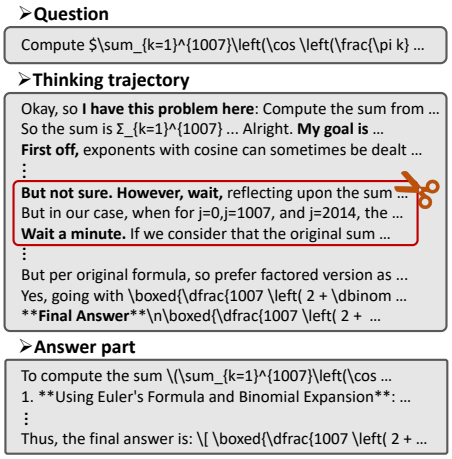

Figure 3: Example of `MidCut`-SFT applied to s1k-1.1 dataset.

We explore several variants of this filtering approach: (1) Step-level filtering: Remove middle steps while preserving the first and last n steps of the trajectory. (2) Token-level filtering: Remove middle tokens while preserving the first and last k tokens, regardless of step boundaries. (3) Similarity-based filtering: We compute Jaccard similarity between each step and the preceding 5 steps of the trajectory. Steps exceeding a predefined similarity threshold are filtered out to reduce redundancy, targeting repetitive reasoning patterns. In this paper, we set $n = 100$ for step-level filtering and $k = 8,000$ for token-level filtering.

Comparison methods. To evaluate the effectiveness of our approach, we compare it against several alternative trajectory reduction strategies: (1) LLM-based Compression: we use external LLMs (Claude Sonnet[2] [21]) to compress thinking trajectories while preserving essential reasoning content. (2) Single-end Preservation: instead of preserving both beginning and end parts, we retain only the

[2]We utilize `claude-sonnet-4-20250514` version of Claude Sonnet 4.

Table 2: **Performance of `MidCut`-SFT on three benchmarks (AIME24, GPQA-D, MATH) with training token usage reduction.** Bold indicates the best performance in each benchmark, and underline indicates performance better than baseline. "Token usage" denotes the proportion of training tokens used relative to the baseline (100%).

Method	AIME24	GPQA-D	MATH	Average	Token usage
Base	0.6444	0.6195	0.9413	0.7351	100%
LLM-based	0.3333	0.5791	0.8900	0.6008	45.66%
Single (first)	0.6222	**0.6380**	**0.9500**	0.7367	21.93%
Single (last)	0.6000	0.6128	0.9420	0.7183	29.66%
Random	0.5667	0.6162	0.9447	0.7092	43.85%
Ours					
Step-level	**0.6889**	0.6229	0.9440	**0.7519**	23.42%
Token-level	0.6111	0.6195	0.9487	0.7264	25.51%
Sim-based	0.6444	0.6279	0.9453	0.7392	9.06%

first $2*n$ steps or only the last $2*n$ steps, maintaining the same total preserved length as `MidCut`-SFT. (3) Random Step Selection: we randomly sample $2*n$ steps from the original trajectory, preserving the same amount of content but without structural consideration.

Experimental results. We fine-tune `Qwen2.5-32B-Instruct` on the s1k-1.1 dataset for 5 epochs and evaluate different trajectory reduction strategies on AIME24 [22], GPQA-D [18], and MATH500 (MATH) [23] datasets. Table 2 shows that our `MidCut`-SFT variants maintain competitive performance while the number of total tokens used for SFT is reduced. Step-level filtering achieves the highest average performance (0.7519), even outperforming full trajectories (0.7351), while using only 23.42% of the training tokens. Compared to alternatives, our dual-end preservation approach shows clear advantages over LLM-based compression (0.6008), single-end methods (0.7367 or 0.7183), and random selection (0.7092). While our variants like similarity-based filtering (0.7392) and token-level filtering (0.7264) also perform well, the simple step-level cutting proves to be most effective.

3.2 `MidCut`-Decoding: `MidCut` for Effective Decoding

Method. `MidCut`-Decoding applies the middle-cutting principle during inference to reduce computational overhead. After the model completes its thinking trajectory, we cut the middle part of the trajectory before producing the final answer. We implement two cutting ratios: removing 33% and 50% of the middle part while preserving the initial problem understanding and final reasoning conclusions.

Experimental results. Table 3 reports `MidCut`-Decoding results on the SFT-trained model with the step-level `MidCut`-SFT introduced in Section 3.1. Compared to using the full reasoning trajectory, trimming 33–50% of the intermediate steps yields almost identical average performance (0.6917 vs. 0.6933). In particular, the 33% cut shows small but consistent gains on GPQA-D and MATH. These findings indicate that `MidCut`-Decoding can substantially reduce inference costs for long reasoning traces while maintaining comparable task accuracy.

Table 3: **Performance of `MidCut`-Decoding.** "Accuracy" denotes the averaged accuracy across AIME24, GPQA-D, and MATH.

Method	Accuracy
Full trajectory	0.7519
`MidCut` 33%	0.7543
`MidCut` 50%	0.7519

4 Conclusion

We have studied the necessity of complete thinking trajectories and proposed `MidCut` for trajectory reduction. Through our analyses, we have shown that intermediate reasoning chains are often redundant. `MidCut`-SFT achieved the improved reasoning results further with efficiency, and `MidCut`-Decoding reduced inference costs without compromising quality. We believe that our simple yet effective method, which showcases how to handle lengthy reasoning trajectories without relying on any challenging restrictions, provides a promising research direction and paves the way for developing more efficient reasoning models in the near future.

References

[1] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 1

[2] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. 1

[3] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

[4] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. 1

[5] Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025.

[6] Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024. 1

[7] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. 1

[8] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022. 1

[9] Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. Reasoning models can be effective without thinking. *arXiv preprint arXiv:2504.09858*, 2025. 1

[10] Jhouben Cuesta-Ramirez, Samuel Beaussant, and Mehdi Mounsif. Large reasoning models are not thinking straight: on the unreliability of thinking trajectories. *arXiv preprint arXiv:2507.00711*, 2025.

[11] Yuyang Wu, Yifei Wang, Ziyu Ye, Tianqi Du, Stefanie Jegelka, and Yisen Wang. When more is less: Understanding chain-of-thought length in llms. *arXiv preprint arXiv:2502.07266*, 2025. 1

[12] Jack Lindsey, Wes Gurnee, Emmanuel Ameisen, Brian Chen, Adam Pearce, Nicholas L. Turner, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. On the biology of a large language model. *Transformer Circuits Thread*, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/biology.html. 1

[13] Emmanuel Ameisen, Jack Lindsey, Adam Pearce, Wes Gurnee, Nicholas L. Turner, Brian Chen, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. Circuit tracing: Revealing computational graphs in language models. *Transformer Circuits*

Thread, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/methods.html. 1, 2

[14] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*, 2019.

[15] Aman Madaan, Katherine Hermann, and Amir Yazdanbakhsh. What makes chain-of-thought prompting effective? a counterfactual study. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1448–1535, 2023.

[16] Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. Dissecting recall of factual associations in auto-regressive language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. 1, 2

[17] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candes, and Tatsunori Hashimoto. s1: Simple test-time scaling. In *Workshop on Reasoning and Planning for Large Language Models*, 2024. 2

[18] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98. 2, 4

[19] Christopher D Manning. *Introduction to information retrieval*. Syngress Publishing,, 2008. 3

[20] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, 2004. 3

[21] Anthropic. Claude 4. https://www.anthropic.com/claude/sonnet, 2025. 3

[22] OpenAI. Learning to reason with llms. OpenAI blog, 2024. URL https://openai.com/index/learning-to-reason-with-llms/. 4

[23] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023. 4