Done Is Better than Perfect: Unlocking Efficient Reasoning by Structured Multi-Turn Decomposition

Zihao Zeng [*1,2] **Xuyao Huang** [*1] **Boxiu Li** [1] **Hao Zhang** [3] **Zhijie Deng** [1]

Abstract

Large Reasoning Models (LRMs) have gained increasing attention over the past few months. Despite being effective, LRMs are criticized for the excessively lengthy Chain-of-Thought (CoT) to derive the final answer, suffering from high first-token and overall latency. Typically, the CoT of LRMs mixes multiple *thinking units*, some of which are split by markers like "aha", "wait", or "alternatively"; each unit attempts to produce a candidate answer to the original query. Hence, a natural idea to improve efficiency is to reduce the unit number. Yet, the fact that the thinking units in vanilla CoT cannot be explicitly managed renders doing so challenging. This paper introduces **Multi-Turn Decomposition (MinD)** to decode conventional CoT into a sequence of explicit, structured, and turn-wise interactions to bridge the gap. In MinD, the model provides a multi-turn response to the query, where each turn embraces a thinking unit and yields a corresponding answer. The subsequent turns can reflect, verify, revise, or explore alternative approaches to both the thinking and answer parts of earlier ones. This not only makes the answer delivered more swiftly, but also enables explicit controls over the iterative reasoning process (i.e., users may halt or continue at any turn). We follow a supervised fine-tuning (SFT) then reinforcement learning (RL) paradigm to realize MinD. We first rephrase the outputs of an LRM into multi-turn formats by prompting another LLM, and then tune the LRM with such data. Observing that the tuned model tends to consume even more tokens than the original one (probably due to that the multi-turn formats introduce additional answer tokens), we advocate leveraging RL algorithms like GRPO to prioritize correct outputs with fewer turns. Trained on the MATH dataset using R1-Distill models, MinD can achieve up to $\sim 70\%$ reduction in both output token usage and time to first token (TTFT), while maintaining competitive performance on reasoning benchmarks such as MATH-500, AIME24, AMC23, and GPQA-Diamond.

[*]Equal contribution [1]Shanghai Jiao Tong University [2]RealAI [3]University of California, San Diego. Correspondence to: Zhijie Deng <zhijied@sjtu.edu.cn>.

The second AI for MATH Workshop at the 42nd International Conference on Machine Learning, Vancouver, Canada.

1. Introduction

Large Reasoning Models (LRMs) have recently attracted significant attention due to their advancing reasoning capabilities, including OpenAI-o1 (Jaech et al., 2024), DeepSeek-R1 (Guo et al., 2025), and Kimi-1.5 (Team et al., 2025). These models have achieved remarkable performance on complex tasks, e.g., mathematical competitions, thanks to their ability to engage in a "think-then-answer" paradigm, where intermediate reasoning chains are generated to induce the final answer. The resultant Chain-of-Thought (CoT) activates contextually accurate responses through iterative exploration and verification of potential solutions.

Despite these advantages, LRMs often suffer from inefficiency issues as the CoT can become excessively lengthy, exhibiting substantially increased computational costs and latency compared to non-reasoning Large Language Models (LLMs). To mitigate these, several strategies have been proposed in recent works. For example, some approaches encourage models to generate answers more directly through strategically designed prompts (Jie et al., 2024), truncate the chain of thought to avoid unnecessary token generation (Fu et al., 2025; Team, 2025), or leverage speculative reasoning via model collaboration (Pan et al., 2025; She et al., 2025). Other approaches focus on reducing token redundancy by refining model reasoning paths through supervised fine-tuning (SFT) (Yang et al., 2025b), or by enhancing decision efficiency with improvements to Group Relative Policy Optimization (GRPO) algorithms (Yu et al., 2025; Liu et al., 2025).

The CoT reasoning process in LRMs is typically composed of multiple *thinking units*—discrete cognitive steps like initial attempts, follow-up validations, reflections, and strategic shifts. Each unit can contribute to generating a candidate

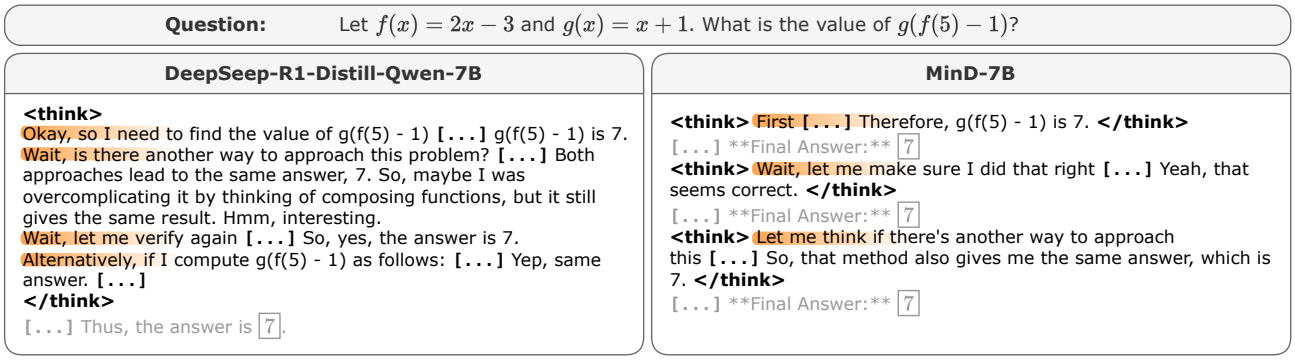

Figure 1. An illustration of responses from DeepSeek-R1-Distill-Qwen-7B and the transformed MinD-7B model on the same math problem. The original LRM follows a think-then-answer format, where the reasoning process consists of multiple thinking units (the start of each new unit is marked with an orange highlight). In contrast, MinD-7B adopts a multi-turn reasoning paradigm, where each turn contains a thinking unit followed by an answer. Also note that MinD-7B tends to use fewer thinking units due to the GRPO training (see Section 3.3).

answer, while current LRMs tend to employ redundant units to ensure the final answer is close to 'perfect' (see an empirical analysis of such redundancy in Figure 2 (right)). While reducing the number of thinking units could improve reasoning efficiency, the inability to explicitly manage these units in standard CoT makes this challenging. It highlights the need for more fine-grained approaches to improve reasoning efficiency.

Building on this insight, we introduce **Multi-Turn Decomposition (MinD)** to decode the "think-then-answer" CoT reasoning into a sequence of multi-turn interactions to enable the explicit control of the number of thinking units, where each turn contains a single thinking unit and an answer generated based on both the current and all preceding units. Refer to Figure 1 for an illustration of the paradigm shift. To implement MinD, we adopt a pipeline combining SFT and GRPO. We first convert conventional CoT traces into structured, multi-turn formats using GPT-4o (OpenAI et al., 2024) and then fine-tune the target model on such data. To further enhance efficiency, we apply GRPO to encourage the model to generate accurate responses within fewer reasoning turns, thereby reducing latency and computational costs.

To evaluate the effectiveness of MinD, we conduct extensive experiments across a range of reasoning benchmarks. On DeepSeek-R1-Distill-Qwen-1.5B, MinD reduces token usage by up to $\sim 70\%$ and accelerates time to first token (TTFT) by $4.2\times$ on MATH-500, while maintaining over 95% accuracy. Furthermore, MinD demonstrates strong out-of-distribution generalization on this model, with token reductions of 69% on AIME24 and 53% on GPQA-Diamond. These results highlight the efficiency and broad applicability of MinD in diverse reasoning scenarios.

2. Related Work

Efficient Reasoning Paradigms The evolution of reasoning frameworks for LLMs has progressed significantly since the introduction of CoT prompting (Wei et al., 2022). CoT has proven effective in enhancing LLMs' reasoning abilities by explicitly guiding models through intermediate reasoning steps (Guo et al., 2025), but this approach often leads to excessively lengthy outputs, resulting in high token consumption and increased latency (Chiang & yi Lee, 2024). These inefficiencies have motivated researchers to explore efficient reasoning paradigms to reduce intermediate tokens without compromising reasoning quality. For example, methods like token skipping (Xia et al., 2024) and length-harmonizing pruning (Luo et al., 2025) have demonstrated significant reductions in token counts while maintaining strong task performance (Fu et al., 2025). Another approach seeks to decouple the reasoning process from explicit token generation by leveraging continuous latent spaces. For instance, Token-Assorted Mixing (Su et al., 2025) and Hidden Thinking frameworks (Shen et al., 2025) aim to perform internal computations without generating extensive token sequences, achieving 3-5× faster processing speeds compared to conventional CoT (Hao et al., 2025). Additionally, several studies have explored integrating reasoning and non-reasoning models to enhance efficiency. For example, the C3OT system (Kang et al., 2025) employs a multi-stage verification pipeline to reduce token redundancy, while speculative reasoning approaches (Pan et al., 2025) dynamically adjust the reasoning depth based on task complexity, further reducing token usage. Hybrid architectures like Hawkeye (She et al., 2025) also leverage speculative decoding (Zhang et al., 2024) to balance accuracy and computational efficiency.

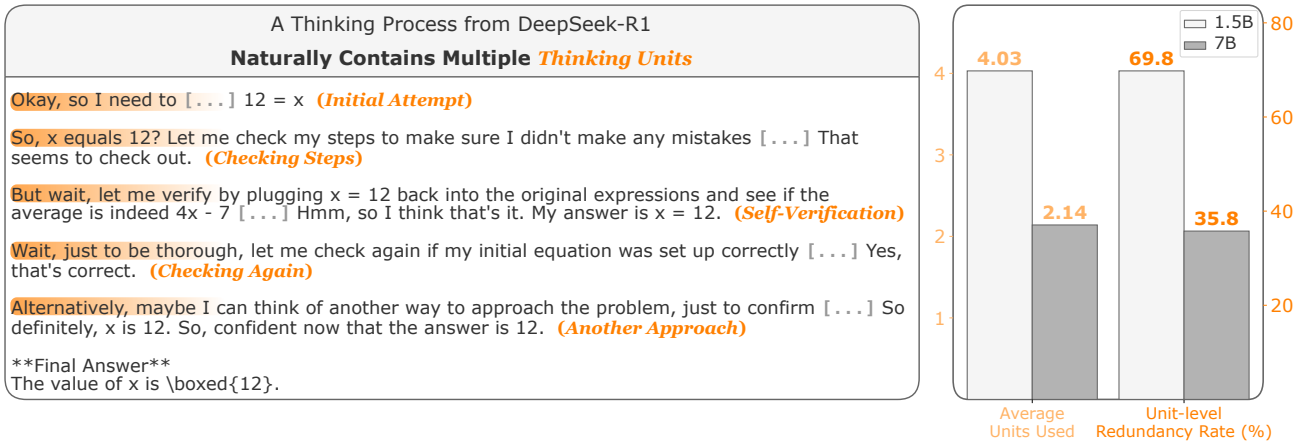

Figure 2. **Left:** An example of a standard CoT from DeepSeek-R1, naturally containing multiple discrete thinking units (the start of each new unit is marked with an orange highlight). **Right:** Empirical analysis of unit-level redundancy, which is calculated based on Equation (5), in R1-distilled models on the MATH-500 dataset, showing an average redundancy rate of 69.8% for the 1.5B model and 35.8% for the 7B model.

Reinforcement Learning for Reasoning Optimization Reinforcement learning (RL) has become an essential tool for optimizing LLM reasoning, providing precise control over decision-making processes. Group Relative Policy Optimization (GRPO) (Shao et al., 2024) is one of the most influential methods in this domain, aligning reward signals with step-wise reasoning validity rather than simply final answer correctness. Building on this foundation, frameworks like DAPO (Yu et al., 2025) and R1-Zero (Liu et al., 2025) incorporate dynamic reward shaping and entropy-controlled exploration to further refine model outputs. Recent advancements have also focused on integrating search-based techniques to enhance reasoning efficiency. For instance, Search-R1 (Jin et al., 2025) combines Monte Carlo Tree Search with policy gradients to optimize reasoning path selection, reducing unnecessary token usage. Similarly, length-aware control frameworks like L1-Controller (Aggarwal & Welleck, 2025) balance correctness and token efficiency through dual reward signals, achieving substantial latency reductions. Other approaches, such as R1-Searcher (Song et al., 2025), incorporate dynamic halting mechanisms to automatically terminate unproductive reasoning chains, significantly improving efficiency in open-domain tasks. ThinkPrune (Hou et al., 2025) adopts length clipping to the reward function, pruning outputs to reduce redundancy.

Training-Based Efficiency Enhancements Training strategies also significantly enhance reasoning efficiency. Supervised fine-tuning (SFT) methods like Thinking-Optimal Scaling (Yang et al., 2025b) align models with optimal solution trajectories, reducing token redundancy without compromising accuracy. Hybrid training regimes have also gained traction, combining imitation learning and reinforcement learning to refine reasoning efficiency. For example, the SpecReason framework (Pan et al., 2025) employs a two-stage process, beginning with teacher-student distillation for foundational policy approximation, followed by adversarial reward shaping for fine-grained optimization.

3. Method

In this section, we first introduce the standard Chain-of-Thought (CoT) reasoning of Large reasoning models (LRMs) and briefly review Group Relative Policy Optimization (GRPO) (DeepSeek-AI, 2025). We then present an empirical study showing how redundant reasoning steps commonly arise in LRMs. Finally, we outline MinD, which reformulates the standard CoT into a multi-turn structure, and discuss how to leverage GRPO to encourage concise and effective multi-turn reasoning.

3.1. Preliminary

CoT for LRMs LRMs commonly adopt a "think-then-answer" paradigm for complex problem solving. Given a query q, an LRM typically produces an output o of the form:

$$q \rightarrow o = \texttt{<think>}\, t \,\texttt{</think>}\, a\,, \tag{1}$$

where t denotes the internal thinking process, delimited by <think> and </think>, and a is the final answer. The thinking process t can be viewed as an exploration of the solution space and is naturally decomposed into multiple *thinking units*—self-contained logical steps that can induce a candidate answer to q, with an example from DeepSeek-R1 (Guo et al., 2025) depicted in Figure 2 (left). Formally, letting u_i denote a thinking unit, there is

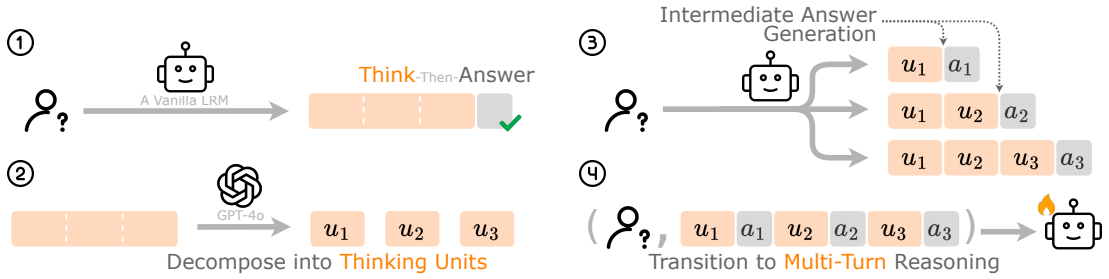

Figure 3. Transforming think-then-answer LRMs into a multi-turn reasoning paradigm, consisting of four steps: (1) Rejection sampling to filter out responses with correct final answers; (2) Unit segmentation using GPT-4o to divide CoTs into discrete reasoning units; (3) Intermediate answer completion to extract answers (a_k) for each prefix sub-trace ($t_{\leq k}$); and (4) SFT to align LRMs with the multi-turn format.

$t = (u_1, u_2, \ldots, u_n)$. These units may arise from (1) an initial attempt to solve the problem, (2) depth-wise exploration such as validation, backtracking, or correction along a single line of reasoning, or (3) breadth-wise search involving alternative methods or perspectives. Each unit can thus be interpreted as a path in the reasoning space, potentially building on previous steps, and may terminate with a provisional answer to the query. However, current LRMs tend to employ many thinking units before gaining the final answer to solve the problem as 'perfectly' as possible, causing significant inefficiency issues.

GRPO Let π_θ denote the current policy and $\pi_{\theta_{\text{old}}}$ the reference policy from the previous iteration. Given a query q, GRPO samples G completions $o_1, \ldots, o_G$ and optimizes the objective:

$$\mathbb{E}_{q,\,\{o_i\}_{i=1}^{G}}\Bigg[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{j=1}^{|o_i|} \min\Big(\rho_{i,j}A_i,\ \text{clip}\big(\rho_{i,j},\,1-\epsilon,\,1+\epsilon\big)\,A_i\Big)\Bigg] \tag{2}$$

where $\rho_{i,j} = \frac{\pi_\theta(o_{i,j}|q,o_{i,<j})}{\pi_{\theta_{\text{old}}}(o_{i,j}|q,o_{i,<j})}$ is the ratio between the new and old policies for token j in sequence o_i and $|o_i|$ is the sequence length. A_i is the group-standardized advantage:

$$A_i = \frac{R(o_i) - \text{mean}(\{R(o_1), \ldots, R(o_G)\})}{\text{std}(\{R(o_1), \ldots, R(o_G)\})}\,, \tag{3}$$

where R denotes the reward function, and $\text{mean}(\{r_1, \ldots, r_G\})$ and $\text{std}(\{r_1, \ldots, r_G\})$ represent the mean and standard deviation of group rewards, respectively. For clarity, we omit the KL regularization term, as it is not the focus of our analysis.

3.2. Unit-Level Redundancy in LRMs

Before devoting to reducing the number of thinking units of LRMs, we first systematically investigate the *unit-level redundancy*, which is intuitively high considering the repeated depth-wise validations or breadth-wise explorations of alternative solution paths, even after repeatedly arriving at essentially the same valid answer, in long CoTs. Concretely, we conducted a detailed analysis using DeepSeek-R1-Distill-Qwen-1.5B/7B (DeepSeek-AI, 2025). We extracted their CoT traces from the MATH (Lightman et al., 2023) and GSM8K (Cobbe et al., 2021) training sets (restricted to correctly answered examples), and segmented each trace into discrete thinking units using GPT-4o (OpenAI et al., 2024) (see Section B for details).

For each segmented trace $t = (u_1, u_2, \ldots, u_n)$, we constructed prefix sub-traces $t_{\leq k} = (u_1, \ldots, u_k)$ for $1 \leq k \leq n$. We then prompted the model to generate an intermediate answer a_k by appending a special stop token `</think>` after $t_{\leq k}$ given the current partial reasoning:

$$q \to o_k = \texttt{<think>}\, t_{\leq k}\, \texttt{</think>}\, a_k\,, \quad k = 1, \cdots, n\,. \tag{4}$$

To quantify unit-level redundancy, we define the minimal sufficient prefix $t_{\leq n^*}$ as the shortest prefix that leads to a correct final answer. The *unit-level redundancy rate* is then defined as:

$$\text{URR} = \frac{n - n^*}{n} \cdot \mathbb{1}_{a_n \text{ is correct}}\,, \tag{5}$$

where n is the total number of thinking units and n^* is the minimal number required for correctness. A higher URR indicates a greater proportion of unnecessary reasoning steps.

Our empirical results, summarized in Figure 2 (right), show that the average unit-level redundancy rates are 69.8% for the 1.5B model and 35.8% for the 7B model. This reveals that a large portion of reasoning in current LRMs is unnecessary for problem-solving.

3.3. Multi-Turn Decomposition (MinD)

Our basic notion is that the model should not be that cautious. Given that "done is better than perfect", we aim to

let the model yield a candidate answer as soon as possible. Besides, we would also like to penalize the unit-level redundancy. MinD realizes these through two key innovations.

Multi-Turn CoT Reformulation MinD first employs supervised fine-tuning (SFT) to shift the reasoning paradigm from "think-then-answer" (i.e., Equation (1)) to a structured multi-turn format:

$$\texttt{<think>}\ u_1\ \texttt{</think>}\ a_1\ \texttt{<think>}\ u_2\ \texttt{</think>}\ a_2 \cdots \texttt{<think>}\ u_n\ \texttt{</think>}\ a_n\ , \tag{6}$$

where the thinking units $(u_1, u_2, \ldots, u_n)$ in the original CoT t are distributed into a sequence of *reasoning turns*. Each turn also includes an intermediate answer a_k.

To construct the training data for multi-turn SFT, we first segment the original thinking process t into $(u_1, u_2, \ldots, u_n)$, and then generate an intermediate answer a_k after each u_k, as described in Section 3.2. The overall pipeline is illustrated in Figure 3.

After training, the learned multi-turn LRM enables flexible management of the thinking units (e.g., choose to continue or abort from the reasoning by manipulating the token `</think>`), but we empirically observe that when applying no control, the model tends to generate even more output tokens than the original one (see Table 4). This is because SFT primarily reshapes the reasoning format without directly addressing unit-level redundancy, and a_k incurs further token usage. To bridge the gap, we suggest leveraging GRPO to prioritize efficient reasoning traces.

Reducing Reasoning Turns via GRPO We define a reward function R comprises three components for GRPO:

$$R = \mathcal{R}_{\text{format}} + \mathcal{R}_{\text{accuracy}} + \mathcal{R}_{\text{unit}}\ . \tag{7}$$

In detail, they are: (1) Format Consistency Reward $\mathcal{R}_{\text{format}}$, which ensures that the generated output adheres to the multi-turn structure described in Equation (6). (2) Answer Accuracy Reward $\mathcal{R}_{\text{accuracy}}$, which rewards the model for producing a correct final answer, as determined by matching a_n to the ground truth. (3) Unit Compactness Reward $\mathcal{R}_{\text{unit}}$, which penalizes cases where a single reasoning unit contains multiple exploratory trajectories and thus encourages a clear separation between reasoning turns. See Section 4.3 for further analysis of this component. The specific weights for each reward component are detailed in Table 1.

Note that we do not introduce an explicit reward term regarding the number of turns, because GRPO inherently introduces an implicit bias toward generating shorter CoTs that yield correct answers. As shown in Equation (2), for a fixed advantage A_i, the per-token normalization $1/|o_i|$ results in larger per-token updates for shorter outputs (Lin et al., 2025; Yu et al., 2025; Liu et al., 2025), thereby encouraging the model to produce more concise and efficient completions. This effect is particularly pronounced in LRMs, which exhibit strong reasoning abilities and can generate diverse correct completions during training. Thus, the GRPO framework naturally encourages shorter reasoning chains, as empirically shown in Figure 5 by the marked reduction in reasoning turns after training.

4. Experiments

In this section, we evaluate the efficiency of MinD across several benchmarks. Section 4.1 describes the experimental setup. Section 4.2 presents the main results, focusing on token reduction, accuracy, and latency. Ablation studies and additional discussion are provided in Section 4.3.

4.1. Setup

Table 1. Reward function value settings.

	$\mathcal{R}_{\text{format}}$	$\mathcal{R}_{\text{accuracy}}$	$\mathcal{R}_{\text{unit}}$
Compliance	+1	+2	0
Non-Compliance	-1	-2	-0.3

Table 2. Training data sizes.

	1.5B	7B
SFT	3610	3532
GRPO	7500	7500

Training Details The training process for MinD consists of two key phases, as described in Section 3.3. The first SFT phase is conducted using the LLaMA-Factory repository (Zheng et al., 2024). We perform full-parameter fine-tuning for 2 epochs with a learning rate of 5e-5. The second GRPO phase leverages the veRL repository (Sheng et al., 2024). During this phase, we train for 1 epoch with an actor learning rate of 1e-6. For each training step, 10 roll-out completions are generated for each sample, with all other hyperparameters set to the default values provided by veRL. The reward function described in Section 3.3 is adopted with the weight configurations listed in Table 1.

Models & Datasets Our experiments are based on DeepSeek-R1-Distill-Qwen-1.5B/7B (DeepSeek-AI, 2025). For SFT, the training data consists of questions from the GSM8K (Cobbe et al., 2021) and MATH (Lightman et al., 2023) training sets. Model-generated responses are filtered via rejection sampling to retain only correct answers, then pre-processed as shown in Figure 3. For

Table 3. Performance comparison of various baselines and our proposed method, MinD, across four reasoning benchmarks: MATH-500, AIME24, AMC23, and GPQA-Diamond. The table reports both accuracy (Acc.; higher is better) and average output token usage (Tokens; lower is better) for each model. Results are shown for both 1.5B and 7B parameter configurations, covering the original LRM (DeepSeek-R1-Distill-Qwen-1.5B and 7B), ThinkPrune (Hou et al., 2025), Dynasor (Fu et al., 2025), DEER (Yang et al., 2025a), and our method, MinD. Note that for MinD, GRPO is performed only on the MATH training set, making MATH-500 in-domain and the others out-of-domain. As shown in the table, MinD consistently achieves competitive or superior accuracy while significantly reducing token usage, demonstrating its effectiveness for efficient and generalizable reasoning.

	MATH-500		AIME24		AMC23		GPQA-Diamond	
	Acc.↑	Tokens↓	Acc.↑	Tokens↓	Acc.↑	Tokens↓	Acc.↑	Tokens↓
1.5B								
Original LRM	85.4	5389	26.7	15177	67.5	9956	32.3	9842
ThinkPrune (Hou et al., 2025)	83.2 -2.6%	1938 -64%	27.1 +1.5%	5631 -63%	73.2 +8.4%	3039 -70%	-	-
DEER (Yang et al., 2025a)	73.2 -14.3%	1118 -79%	20.0 -25.1%	3302 -78%	47.5 -29.6%	2384 -76%	5.6 -82.7%	4128 -58%
MinD	82.8 -3.0%	1719 -68%	30.0 +12.4%	4856 -68%	77.5 +14.8%	2384 -76%	31.3 -3.1%	4690 -52%
7B								
Original LRM	93.0	3928	50.0	14107	90.0	6076	50.5	8390
Dynasor (Fu et al., 2025)	88.5 -4.8%	2591 -34%	47.7 -4.6%	8760 -38%	87.1 -3.2%	4913 -19%	-	-
DEER (Yang et al., 2025a)	87.4 -6.0%	975 -75%	33.3 -33.4%	3235 -77%	82.5 -8.3%	1622 -73%	27.3 -45.9%	2265 -73%
MinD	91.6 -1.5%	2859 -27%	46.7 -6.6%	7258 -49%	95.0 +5.6%	3777 -38%	53.0 +5.0%	6845 -18%

GRPO, we use the MATH training set exclusively, with sample sizes detailed in Table 2. We evaluate on both in-distribution (MATH-500 (Lightman et al., 2023)) and out-of-distribution benchmarks, including AIME24 (of America, 2024), AMC23 (of Science, 2023), and GPQA-Diamond (Rein et al., 2023), to assess generalization.

Baselines To assess the efficiency of our method, we compare against the following baselines: Original LRM: The base models used in this work, DeepSeek-R1-Distill-Qwen-1.5B and 7B. ThinkPrune (Hou et al., 2025): Adds length clipping to the GRPO reward and is trained on the AIME-AMC subset, progressively pruning outputs at the token level to reduce response length. DEER (Yang et al., 2025a): A training-free approach that detects "action transition points" (e.g., "Wait," "Alternatively," "Hmm") to trigger answer generation, and halts decoding when the mean token probability surpasses a confidence threshold. Dynasor (Fu et al., 2025): Periodically inserts probes (e.g., every 32, 64, or 128 tokens) to extract intermediate answers and assess their consistency, enabling early termination of generation.

Evaluation Metrics We evaluate MinD using three primary metrics: accuracy, average output token usage, and time-to-first-token (TTFT). TTFT measures the time it takes for the model to generate the first answer token of the response, from when the prompt was sent—a key determinant of user experience. The evaluations are conducted using the Open-R1 evaluation scripts (Face, 2025), with a maximum sequence length of 32,768 tokens, a temperature setting of 0.6, and a top-p value of 0.95, running on four NVIDIA A100 GPUs.

4.2. Main Results

Reducing Output Tokens for Efficient Reasoning After training the 1.5B and 7B multi-turn reasoning models as described in Section 4.1, we evaluated their token efficiency across a range of reasoning benchmarks. The results, summarized in Table 3, show that MinD consistently reduces output token usage while maintaining strong performance. On in-domain MATH-500, MinD lowers the average token usage to 1719 for the 1.5B model—a 68% reduction from the Original LRM (5389 tokens)—while achieving 82.8% accuracy. Although ThinkPrune attains similar accuracy (83.2%), it requires more tokens (1938). DEER achieves the lowest token usage (1118), but with a substantial accuracy drop to 73.2%. For the 7B model, MinD reduces average token usage by 27% compared to the Original LRM (2859 vs. 3928), with a high accuracy of 91.6%, outperforming both Dynasor and DEER in the balance of accuracy and efficiency. MinD's efficiency generalizes well to out-of-domain benchmarks. For example, on AMC23 (1.5B), MinD reaches 77.5% accuracy with 2384 tokens, substantially outperforming ThinkPrune and DEER in both accuracy and token reduction. Similar trends are observed on AIME24 and GPQA-Diamond. These results demonstrate that MinD effectively eliminates unnecessary reasoning steps, producing concise outputs without compromising performance.

Table 4. Comparison of different training strategies on DeepSeek-R1-Distill-Qwen-1.5B. Original LRM refers to the pretrained baseline. SFT-Only applies only the supervised fine-tuning step from MinD. Non-Multi-Turn applies GRPO without explicit multi-turn segmentation. MinD denotes our full method with both multi-turn segmentation and GRPO. Acc.↑ indicates accuracy (higher is better), and Tokens↓ indicates average output length (lower is better).

	Original LRM		SFT-Only		Non-Multi-Turn		MinD	
	Acc.↑	Tokens↓	Acc.↑	Tokens↓	Acc.↑	Tokens↓	Acc.↑	Tokens↓
MATH-500	85.4	5389	82.8	5655	82.0	1866	82.8	1719
AIME24	26.7	15177	26.7	20675	20.0	7654	30.0	4856
AMC23	67.5	9956	77.5	8409	65.0	3415	77.5	2384
GPQA-Diamond	32.3	9842	28.3	12501	28.8	3397	37.4	4345

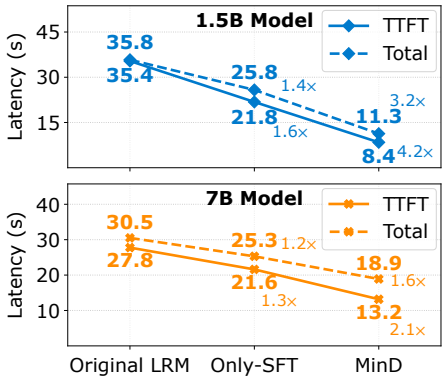

Figure 4. TTFT (time to first token) and total latency of two DeepSeek-R1-distilled models on MATH-500. MinD achieves up to 4.2× (1.5B) and 2.1× (7B) speedups over the original LRMs in TTFT, and 3.2× (1.5B) and 1.6× (7B) in total latency.

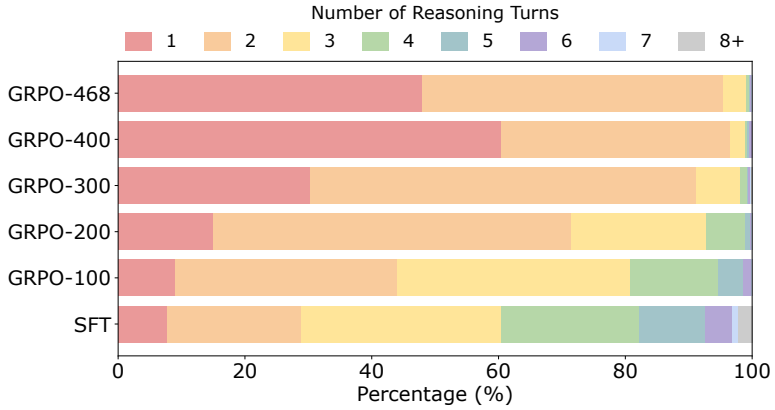

Figure 5. The distribution of reasoning turns for MinD at different training stages (1.5B model) on the MATH-500 dataset. Each bar represents a model checkpoint, including the SFT model and successive GRPO training steps. As GRPO training progresses, the number of reasoning turns per output decreases and becomes increasingly concentrated at 1 or 2 turns (highlighted in red and orange), demonstrating the effectiveness of GRPO in mitigating reasoning redundancy.

Reducing TTFT and Total Latency The TTFT and total response latency for the original R1-distilled LRMs and our MinD models are summarized in Figure 4. As shown, MinD significantly reduces both TTFT and total latency across both model sizes. For the 1.5B configuration, the original 1.5B model requires 35.4s TTFT, which drops to 21.8s after SFT and further to 8.4s with MinD, resulting in a 4.2× speedup. The total latency is similarly reduced from 35.8s (original) to 25.8s (SFT) and 11.3s (MinD), a 2.1× improvement. For the 7B model, TTFT decreases from 27.8s (original) to 21.6s (SFT) and 13.2s (MinD), achieving a 2.1× speedup. The total latency is reduced from 30.5s to 25.3s and 18.9s, for a 1.6× speedup. These results show that MinD shortens both the time to first answer token and the overall response latency.

4.3. Discussion & Ablation

GRPO is Crucial for Efficient Reasoning As discussed in Section 3.3, SFT alone does not guarantee efficient reasoning. To demonstrate this, we compare the performance of models after SFT and after the full MinD pipeline, as shown in Table 4. The results reveal that SFT-only training often increases average output token usage relative to the original LRM. In contrast, applying GRPO further leads to substantial reductions in token usage while preserving accuracy, underscoring the essential role of GRPO in enabling concise and effective reasoning.

Role of $\mathcal{R}_{\text{unit}}$ in Maintaining Multi-Turn Reasoning As discussed in Section 3.3 and detailed in Table 1, our GRPO framework introduces a Unit Compactness Reward, $\mathcal{R}_{\text{unit}}$, to enforce that each reasoning turn contains only a single,

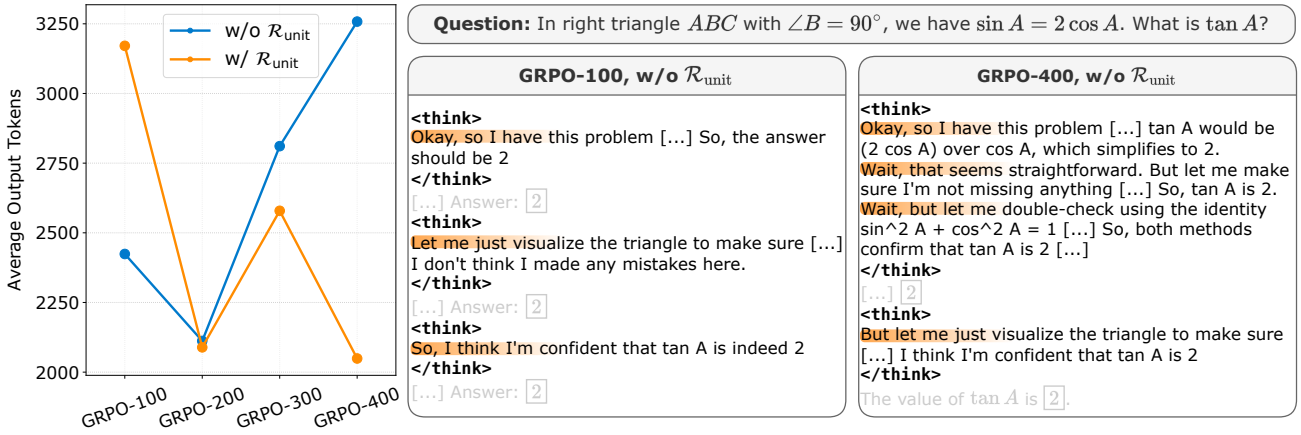

Figure 6. **Left:** Comparison of GRPO training with and without $\mathcal{R}_{\text{unit}}$ on MATH-500 for different 1.5B model checkpoints, showing Average Output Tokens for each. Removing $\mathcal{R}_{\text{unit}}$ leads to instability and collapse in output length. **Right:** An illustrative case comparing the outputs of GRPO-100-step and GRPO-400-step checkpoints trained without $\mathcal{R}_{\text{unit}}$. While the earlier checkpoint (GRPO-100) maintains clear multi-turn reasoning, the later checkpoint (GRPO-400) exhibits several thinking units within a single turn (the start of each new unit is marked with an orange highlight), demonstrating that omitting $\mathcal{R}_{\text{unit}}$ results in blurred step boundaries and loss of controllable, structured reasoning.

coherent exploratory trajectory. This mechanism is essential for preventing the model from degenerating into the original monolithic think-then-answer style—a common outcome under GRPO's token-level averaging (Section 3.3), which tends to favor shorter correct outputs. Without a specific penalty for multi-trajectory turns, the model may skip intermediate answers, collapsing the multi-turn reasoning structure into a single-block CoT. To counteract this, $\mathcal{R}_{\text{unit}}$ penalizes reasoning turns that contain multiple exploratory trajectories, detected by linguistic cues such as phrases like "double-check." This strategy encourages each turn to contain only one exploratory trajectory—especially in the critical first turn—without requiring external supervision, and thus maintains the multi-turn paradigm throughout training. The impact of $\mathcal{R}_{\text{unit}}$ is demonstrated in Figure 6, which shows how its absence leads to a collapse in output structure and length.

MinD Effectively Alleviates Redundancy To demonstrate the effectiveness of GRPO in reducing redundancy, we plotted the distribution of reasoning turns for SFT and GRPO models on the MATH-500 dataset, as shown in Figure 5. The figure clearly illustrates that GRPO significantly reduces the number of reasoning turns, indicating a more compact and efficient reasoning process compared to the purely SFT-trained models. Additionally, from the data in Table 3, GRPO reduces the average output tokens on MATH-500 by 68.1% for the 1.5B model and 27.2% for the 7B model, compared to their respective original LRMs. This aligns well, though not directly, with the redundancy rates of 69.8% and 35.8% for these models, as reported in Figure 2 (Right). While not directly comparable, these figures collectively show that MinD via GRPO significantly reduces redundancy, yielding more concise and efficient outputs.

The Importance of Multi-Turn Structure To evaluate the impact of the multi-turn design, we performed SFT using responses from the original distilled-1.5B model, without applying any multi-turn segmentation (i.e., using the same question set as in step (1) of Figure 3), followed by GRPO with only the format and outcome rewards. As shown in Table 4, the Non-Multi-Turn model achieves comparable results to MinD on in-distribution MATH-500, but exhibits a notable drop in accuracy and only marginal reductions in token usage on out-of-distribution benchmarks. We hypothesize that the standard CoT format limits models' flexibility in adjusting reasoning steps, hindering the learning of controllable and generalizable processes.

Additional discussion can be found in Section A.

5. Conclusion And Limitation

In this paper, we introduced Multi-Turn Decomposition (MinD), an efficient method for improving the reasoning efficiency of large language models. MinD significantly reduces token usage and latency while preserving strong performance across diverse tasks. Our results show that structured reasoning mitigates slow responses and high computational costs in large language models. Future work includes scaling MinD to larger models, extending to more domains, and adapting turn counts to difficulty or user needs.

References

Aggarwal, P. and Welleck, S. L1: Controlling how long a reasoning model thinks with reinforcement learning, 2025. URL https://arxiv.org/abs/2503.04697.

Chiang, C.-H. and yi Lee, H. Over-reasoning and redundant calculation of large language models, 2024. URL https://arxiv.org/abs/2401.11467.

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.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Face, H. Open r1: A fully open reproduction of deepseek-r1, January 2025. URL https://github.com/huggingface/open-r1.

Fu, Y., Chen, J., Zhuang, Y., Fu, Z., Stoica, I., and Zhang, H. Reasoning without self-doubt: More efficient chain-of-thought through certainty probing. In *ICLR 2025 Workshop on Foundation Models in the Wild*, 2025. URL https://openreview.net/forum?id=wpK4IMJfdX.

Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Hao, S., Sukhbaatar, S., Su, D., Li, X., Hu, Z., Weston, J. E., and Tian, Y. Training large language model to reason in a continuous latent space, 2025. URL https://openreview.net/forum?id=tG4SgayTtk.

Hou, B., Zhang, Y., Ji, J., Liu, Y., Qian, K., Andreas, J., and Chang, S. Thinkprune: Pruning long chain-of-thought of llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2504.01296.

Jaech, A., Kalai, A., Lerer, A., Richardson, A., El-Kishky, A., Low, A., Helyar, A., Madry, A., Beutel, A., Carney, A., et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Jie, R., Meng, X., Shang, L., Jiang, X., and Liu, Q. Prompt-based length controlled generation with multiple control types, 2024. URL https://arxiv.org/abs/2406.10278.

Jin, B., Zeng, H., Yue, Z., Yoon, J., Arik, S., Wang, D., Zamani, H., and Han, J. Search-r1: Training llms to reason and leverage search engines with reinforcement learning, 2025. URL https://arxiv.org/abs/2503.09516.

Kang, Y., Sun, X., Chen, L., and Zou, W. C3ot: Generating shorter chain-of-thought without compromising effectiveness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 24312–24320, 2025. doi: 10.1609/aaai.v39i23.34608.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Lin, Z., Lin, M., Xie, Y., and Ji, R. Cppo: Accelerating the training of group relative policy optimization-based reasoning models. *arXiv preprint arXiv:2503.22342*, 2025.

Liu, Z., Chen, C., Li, W., Qi, P., Pang, T., Du, C., Lee, W. S., and Lin, M. Understanding r1-zero-like training: A critical perspective, 2025. URL https://arxiv.org/abs/2503.20783.

Luo, H., Shen, L., He, H., Wang, Y., Liu, S., Li, W., Tan, N., Cao, X., and Tao, D. O1-pruner: Length-harmonizing fine-tuning for o1-like reasoning pruning, 2025. URL https://arxiv.org/abs/2501.12570.

of America, M. A. American invitational mathematics examination - aime 2024, 2024. URL https://maa.org/math-competitions/american-invitational-mathematics-examination-aim

of Science, A. A. Australian mathematics competition - amc 2023, 2023. URL https://www.amt.edu.au/news/amc-2023.

OpenAI, Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., Avila, R., Babuschkin, I., Balaji, S., Balcom, V., Baltescu, P., Bao, H., Bavarian, M., Belgum, J., Bello, I., Berdine, J., Bernadett-Shapiro, G., Berner, C., Bogdonoff, L., Boiko, O., Boyd, M., Brakman, A.-L., Brockman, G., Brooks, T., Brundage, M., Button, K., Cai, T., Campbell, R., Cann, A., Carey, B., Carlson, C., Carmichael, R., Chan, B., Chang, C., Chantzis, F., Chen, D., Chen, S., Chen, R., Chen, J., Chen, M., Chess, B., Cho, C., Chu, C., Chung, H. W., Cummings, D., Currier, J., Dai, Y., Decareaux, C., Degry, T., Deutsch, N., Deville, D., Dhar, A., Dohan, D., Dowling, S., Dunning, S., Ecoffet, A., Eleti, A., Eloundou, T., Farhi, D., Fedus, L., Felix, N., Fishman, S. P., Forte, J., Fulford, I., Gao, L., Georges, E., Gibson, C., Goel, V., Gogineni, T., Goh, G., Gontijo-Lopes, R., Gordon, J., Grafstein, M., Gray, S.,

Greene, R., Gross, J., Gu, S. S., Guo, Y., Hallacy, C., Han, J., Harris, J., He, Y., Heaton, M., Heidecke, J., Hesse, C., Hickey, A., Hickey, W., Hoeschele, P., Houghton, B., Hsu, K., Hu, S., Hu, X., Huizinga, J., Jain, S., Jain, S., Jang, J., Jiang, A., Jiang, R., Jin, H., Jin, D., Jomoto, S., Jonn, B., Jun, H., Kaftan, T., Łukasz Kaiser, Kamali, A., Kanitscheider, I., Keskar, N. S., Khan, T., Kilpatrick, L., Kim, J. W., Kim, C., Kim, Y., Kirchner, J. H., Kiros, J., Knight, M., Kokotajlo, D., Łukasz Kondraciuk, Kondrich, A., Konstantinidis, A., Kosic, K., Krueger, G., Kuo, V., Lampe, M., Lan, I., Lee, T., Leike, J., Leung, J., Levy, D., Li, C. M., Lim, R., Lin, M., Lin, S., Litwin, M., Lopez, T., Lowe, R., Lue, P., Makanju, A., Malfacini, K., Manning, S., Markov, T., Markovski, Y., Martin, B., Mayer, K., Mayne, A., McGrew, B., McKinney, S. M., McLeavey, C., McMillan, P., McNeil, J., Medina, D., Mehta, A., Menick, J., Metz, L., Mishchenko, A., Mishkin, P., Monaco, V., Morikawa, E., Mossing, D., Mu, T., Murati, M., Murk, O., Mély, D., Nair, A., Nakano, R., Nayak, R., Neelakantan, A., Ngo, R., Noh, H., Ouyang, L., O'Keefe, C., Pachocki, J., Paino, A., Palermo, J., Pantuliano, A., Parascandolo, G., Parish, J., Parparita, E., Passos, A., Pavlov, M., Peng, A., Perelman, A., de Avila Belbute Peres, F., Petrov, M., de Oliveira Pinto, H. P., Michael, Pokorny, Pokrass, M., Pong, V. H., Powell, T., Power, A., Power, B., Proehl, E., Puri, R., Radford, A., Rae, J., Ramesh, A., Raymond, C., Real, F., Rimbach, K., Ross, C., Rotsted, B., Roussez, H., Ryder, N., Saltarelli, M., Sanders, T., Santurkar, S., Sastry, G., Schmidt, H., Schnurr, D., Schulman, J., Selsam, D., Sheppard, K., Sherbakov, T., Shieh, J., Shoker, S., Shyam, P., Sidor, S., Sigler, E., Simens, M., Sitkin, J., Slama, K., Sohl, I., Sokolowsky, B., Song, Y., Staudacher, N., Such, F. P., Summers, N., Sutskever, I., Tang, J., Tezak, N., Thompson, M. B., Tillet, P., Tootoonchian, A., Tseng, E., Tuggle, P., Turley, N., Tworek, J., Uribe, J. F. C., Vallone, A., Vijayvergiya, A., Voss, C., Wainwright, C., Wang, J. J., Wang, A., Wang, B., Ward, J., Wei, J., Weinmann, C., Welihinda, A., Welinder, P., Weng, J., Weng, L., Wiethoff, M., Willner, D., Winter, C., Wolrich, S., Wong, H., Workman, L., Wu, S., Wu, J., Wu, M., Xiao, K., Xu, T., Yoo, S., Yu, K., Yuan, Q., Zaremba, W., Zellers, R., Zhang, C., Zhang, M., Zhao, S., Zheng, T., Zhuang, J., Zhuk, W., and Zoph, B. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

Pan, R., Dai, Y., Zhang, Z., Oliaro, G., Jia, Z., and Netravali, R. Specreason: Fast and accurate inference-time compute via speculative reasoning. *arXiv preprint arXiv:2504.07891*, 2025.

Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., and Bowman, S. R. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y. K., Wu, Y., and Guo, D. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

She, J., Li, Z., Huang, Z., Li, Q., Xu, P., Li, H., and Ho, Q. Hawkeye:efficient reasoning with model collaboration, 2025. URL https://arxiv.org/abs/2504.00424.

Shen, X., Wang, Y., Shi, X., Wang, Y., Zhao, P., and Gu, J. Efficient reasoning with hidden thinking, 2025. URL https://arxiv.org/abs/2501.19201.

Sheng, G., Zhang, C., Ye, Z., Wu, X., Zhang, W., Zhang, R., Peng, Y., Lin, H., and Wu, C. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Song, H., Jiang, J., Min, Y., Chen, J., Chen, Z., Zhao, W. X., Fang, L., and Wen, J.-R. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*, 2025.

Su, D., Zhu, H., Xu, Y., Jiao, J., Tian, Y., and Zheng, Q. Token assorted: Mixing latent and text tokens for improved language model reasoning, 2025. URL https://arxiv.org/abs/2502.03275.

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. *arXiv preprint arXiv:2501.12599*, 2025.

Team, Q. Qwen3, April 2025. URL https://qwenlm.github.io/blog/qwen3/.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837, 2022.

Xia, H., Wang, W., Yu, H., Wang, X., Lin, X., and Zhou, M. Tokenskip: Controllable chain-of-thought compression in llms. *arXiv preprint arXiv:2502.12067*, 2024.

Yang, C., Si, Q., Duan, Y., Zhu, Z., Zhu, C., Lin, Z., Cao, L., and Wang, W. Dynamic early exit in reasoning models, 2025a. URL https://arxiv.org/abs/2504.15895.

Yang, W., Ma, S., Lin, Y., and Wei, F. Towards thinking-optimal scaling of test-time compute for llm reasoning, 2025b. URL https://arxiv.org/abs/2502.18080.

Yu, Q., Zhang, Z., Zhu, R., Yuan, Y., Zuo, X., Yue, Y., Fan, T., Liu, G., Liu, L., Liu, X., Lin, H., Lin, Z., Ma, B., Sheng, G., Tong, Y., Zhang, C., Zhang, M., Zhang, W., Zhu, H., Zhu, J., Chen, J., Chen, J., Wang, C., Yu, H., Dai, W., Song, Y., Wei, X., Zhou, H., Liu, J., Ma, W.-Y., Zhang, Y.-Q., Yan, L., Qiao, M., Wu, Y., and Wang, M. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Zhang, J., Wang, J., Li, H., Shou, L., Chen, K., Chen, G., and Mehrotra, S. Draft & verify: Lossless large language model acceleration via self-speculative decoding. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11263–11282, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.607. URL https://aclanthology.org/2024.acl-long.607/.

Zheng, Y., Zhang, R., Zhang, J., Ye, Y., Luo, Z., Feng, Z., and Ma, Y. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguistics. URL http://arxiv.org/abs/2403.13372.

A. Word Frequency Analysis of Thinking Units

In this section, we collect and compare the number of distinct words representing thinking units in DeepSeek-R1-Distill-1.5B, including the Original LRM, Non-Multi-Turn (GRPO applied without explicit multi-turn segmentation) , and MinD. Although these words do not precisely correspond to the number of actual thinking units, they serve as a meaningful proxy and offer indicative insights into their distribution(see Table 5 for details).

Table 5. The frequency of words representing thinking units in outputs generated by Original LRM, Non-Multi-Turn and MinD across MATH-500, AIME24 and AMC23.

	Wait	Alternatively	double-check	check	verify
MATH-500					
Original LRM	13993	2206	368	1272	**124**
Non-Multi-Turn	1822	333	41	**347**	193
MinD	**1651**	**237**	**10**	434	249
AIME24					
Original LRM	3742	415	20	215	17
Non-Multi-Turn	317	67	**0**	45	19
MinD	**211**	**45**	**0**	**34**	**8**
AMC23					
Original LRM	2302	385	35	205	45
Non-Multi-Turn	246	38	3	**42**	**17**
MinD	**215**	**30**	**0**	50	22

B. Prompting for MinD

In this section, we present the complete prompt formats used in the MinD process (see Figure 3 for details).

Q&A Template

```
{Question}
Please reason step by step, and put your final answer within \\boxed{}.
```

Decomposing into Thinking Units

```
You will be provided with a math problem and a solution generated by a
reasoning model.  The model's response may contain multiple Reasoning
Rounds.  One Reasoning Round is a part of the full model generation and
is defined as a complete reasoning process or verification process that
explicitly contains the final answer.  Your task is to carefully analyze
the response and segment it into individual Reasoning Rounds.  Specifically,
insert ``[split]'' between every two consecutive Reasoning Rounds.
---
Problem:  {question}
Solution:  {prediction}
---
Please give the solution with ``[split]'' tags without any redundant words.
```