AdaptThink: Reasoning Models Can Learn When to Think

Anonymous ACL submission

Abstract

Recently, large reasoning models have achieved impressive performance on various tasks by employing human-like deep thinking. However, the lengthy thinking process substantially increases inference overhead, making efficiency a critical bottleneck. In this work, we first demonstrate that *NoThinking*, which prompts the reasoning model to skip thinking and directly generate the final solution, is a better choice for relatively simple tasks in terms of both performance and efficiency. Motivated by this, we propose *AdaptThink*, a novel RL algorithm to teach reasoning models to choose the optimal thinking mode adaptively based on problem difficulty. Specifically, *AdaptThink* features two core components: (1) a constrained optimization objective that encourages the model to choose *NoThinking* while maintaining the overall performance; (2) an importance sampling strategy that balances *Thinking* and *NoThinking* samples during on-policy training, thereby enabling cold start and allowing the model to explore and exploit both thinking modes throughout the training process. Our experiments indicate that *AdaptThink* significantly reduces the inference costs while further enhancing performance. Notably, on three math datasets, *AdaptThink* reduces the average response length of DeepSeek-R1-Distill-Qwen-1.5B by 53% and improves its accuracy by 2.4%, highlighting the promise of adaptive thinking-mode selection for optimizing the balance between reasoning quality and efficiency.

1 Introduction

Recent advancements in large reasoning models, such as OpenAI o1(OpenAI, 2024) and DeepSeek-R1 (DeepSeek-AI, 2025), have demonstrated remarkable capabilities in tackling complex tasks. Given a problem, these models first engage in a long chain of thought—also referred to as *Thinking*—where they iteratively explore different approaches, accompanied by reflection, backtracking, and self-verification. Subsequently, they produce a final solution that contains only the correct steps and the answer to present to the user. While the long-thinking process markedly enhances the model's reasoning capacities, it also substantially increases inference overhead and latency (Qu et al., 2025; Sui et al., 2025). In particular, for some simple queries where users expect fast, near-instant responses, these models often generate excessive thinking with unnecessarily detailed steps or redundant attempts, resulting in a suboptimal user experience (Chen et al., 2024; Shen et al., 2025).

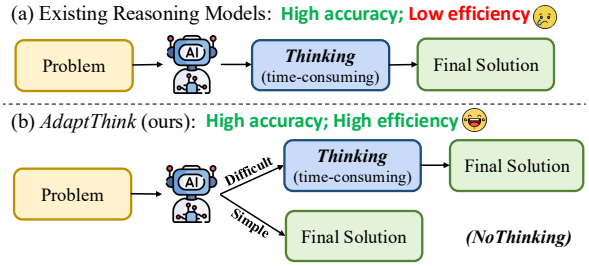

Figure 1: *AdaptThink* enables models to adaptively select between *Thinking* or *NoThinking* mode based on problem difficulty, thereby improving reasoning efficiency while further improving overall performance.

Existing efforts to improve reasoning efficiency primarily focus on reducing the length of model responses, either through incorporating length-based rewards in reinforcement learning (RL) (Arora and Zanette, 2025; Team et al., 2025), finetuning with preference pairs that penalizes longer responses (Chen et al., 2024; Shen et al., 2025; Luo et al., 2025a), or by merging reasoning and non-reasoning models (Wu et al., 2025). Nevertheless, these methods still apply thinking to all instances, regardless of whether thinking itself is necessary for every problem. In this work, we draw inspiration from the recently introduced *NoThinking* approach (Ma et al., 2025), which allows reasoning models to skip the thinking process and directly generate the final solution by prompting with a pseudo-thinking process. Specifically,

we further simplify the approach by prompting the model with an empty thinking segment (i.e., "`<think></think>`"). Our pilot study in Section 3 indicates that *NoThinking* achieves comparable or even better performance than *Thinking* on relatively simple problems (up to high-school competition level), while significantly reducing token usage; the benefits of *Thinking* only become pronounced when the problem is difficult enough.

In light of this observation, we are curious: *Can the reasoning model learn to select Thinking or NoThinking mode adaptively based on the difficulty of the input problem, thereby achieving more efficient reasoning without sacrificing or even improving performance?* To this end, we propose *AdaptThink*, a novel RL algorithm to teach reasoning models when to think. Specifically, *AdaptThink* features two core components: (1) a constrained optimization objective that encourages the model to choose *NoThinking* while ensuring overall performance does not degrade; (2) an importance sampling strategy that balances *Thinking* and *NoThinking* samples during on-policy training, thereby overcoming the challenge of cold start and allowing the model to explore and exploit both thinking modes throughout the whole training process.

Our experiments demonstrate that *AdaptThink* effectively enables reasoning models to adaptively select the optimal thinking mode based on problem difficulty, leading to substantial reductions in inference cost compared to prior approaches, while consistently enhancing model accuracy. For instance, on GSM8K, MATH500, and AIME2024, *AdaptThink* reduces the average response length of DeepSeek-R1-Distill-Qwen-1.5B by 50.9%, 63.5%, and 44.7%, and improving its accuracy by 4.1%, 1.4%, and 1.6%, respectively. The remarkable results substantiate the potential of difficulty-adaptive thinking-mode selection as a promising paradigm for advancing the trade-off between reasoning performance and efficiency.

In summary, our key contributions are as follows: (1) We simplify the *NoThinking* approach and demonstrate its advantages over *Thinking* for simpler tasks in terms of both performance and efficiency; (2) We propose *AdaptThink*, a novel RL algorithm that empowers reasoning models to adaptively select the optimal thinking mode adaptively based on problem difficulty, thereby substantially reducing inference costs and further improving performance; (3) We conduct extensive experiments to validate the efficacy of *AdaptThink*.

2 Related Work

Large Reasoning Models. Recent frontier large reasoning models (LRMs), such as OpenAI o1 (OpenAI, 2024), DeepSeek-R1 (DeepSeek-AI, 2025), and QwQ (Qwen Team, 2025), have developed the ability to employ human-like deep thinking in problem solving by generating a long chain of thought before arriving at a final solution. Such advanced ability is typically acquired through large-scale RL with verified rewards or fine-tuning on distilled reasoning traces. Despite promising performance, the lengthy thinking process introduces substantial inference costs and latency. Consequently, a variety of approaches have been proposed for more efficient reasoning.

Efficient Reasoning for LRMs. Most existing methods to improve the efficiency of LRMs focus on reducing the token usage in model responses. Some methods incorporate length-based rewards into RL to incentivize more concise responses (Arora and Zanette, 2025; Team et al., 2025) or enable precise control over response length (Aggarwal and Welleck, 2025). Other approaches finetune models with length-related preference pairs, which are obtained from best-of-N sampling (Luo et al., 2025a; Shen et al., 2025) or through postprocessing (Chen et al., 2024). Additionally, several works pursue training-free methods to decrease response length, employing techniques such as model merging (Team et al., 2025; Wu et al., 2025) or prompting (Han et al., 2024; Muennighoff et al., 2025; Fu et al., 2025; Xu et al., 2025). Nevertheless, these methods still utilize long thinking for all problems, while the recent *NoThinking* approach (Ma et al., 2025) allows reasoning models to bypass long thinking and directly output the final solution via prompting, achieving performance comparable to *Thinking* in low-token-budget settings. In this work, we further demonstrate that even with a sufficient token budget, *NoThinking* can outperform *Thinking* on simple problems while using significantly fewer tokens. This observation motivates us to propose *AdaptThink* to teach reasoning models to adaptively select the optimal thinking mode based on problem difficulty, which is a new direction for efficient reasoning.

3 Motivation

3.1 Preliminary

Consider a reasoning model parameterized by θ and denoted by π_θ. Given a prompt $x =$

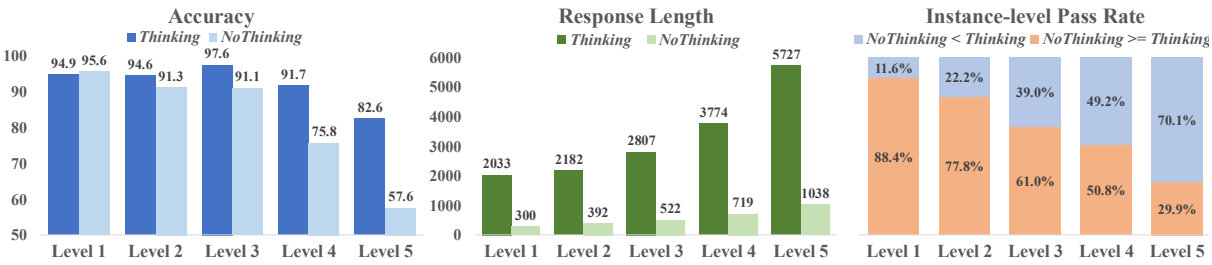

Figure 2: Comparison of DeepSeek-R1-Distill-Qwen-7B using *Thinking* and *NoThinking* mode across different difficulty levels of MATH500 dataset.

$[x_1, \ldots, x_n, \texttt{<think>}]$, where $[x_1, \ldots, x_n]$ represents the problem and <think> is the special token to start the thinking process, the model generates a response $y = [y_1, \ldots, y_l, \texttt{</think>}, y_{l+2}, \ldots, y_m]$. Here, $[y_1, \ldots, y_l]$ corresponds to the thinking, which is a long chain of thought consisting of constant exploration, reflection, and self-verification. The token </think> marks the end of thinking. The remaining sequence, $[y_{l+2}, \ldots, y_m]$, denotes the final solution, which only includes the correct steps to solve the problem and the final answer. From the perspective of probability theory, the response y is a sample drawn from the conditional probability distribution $\pi_\theta(\cdot|x)$. Since y is generated in an auto-regressive way, the conditional probability $\pi_\theta(y|x)$ can be decomposed as:

$$\pi_\theta(y|x) = \prod_{t=1}^{m} \pi_\theta(y_t|x, y_{<t}) \tag{1}$$

3.2 *NoThinking* is Better for Simple Problems

Current reasoning models, such as OpenAI o1 and DeepSeek-R1, apply long thinking across all problems (denoted as *Thinking* mode). Though enhancing models' reasoning capabilities, the lengthy thinking process often leads to unnecessary computation overhead, especially for some simple problems that can also be solved by non-reasoning models (e.g., GPT-4o and Qwen-2.5-Instruct) without thinking. Recently, Ma et al. (2025) proposed *NoThinking* method, which enables reasoning models to bypass long thinking and directly generate the final solution by prompting with a fake thinking process *"Okay, I think I have finished thinking.*</think>*"*, and found it is still effective in low-token-budget settings. In this work, we further simplify *NoThinking* by providing the models with an empty thinking (i.e., enforcing the first generated token $y_1 =$ </think>). Then, we conduct a pilot study to compare *Thinking* and *NoThinking* from the perspective of problem difficulty, with a sufficient token budget (16K).

Specifically, we utilize MATH500 (Lightman et al., 2024) dataset for the pilot study since its have categorized problems into five difficulty levels. For each problem, we employ DeepSeek-R1-Distill-Qwen-7B to generate 16 responses using *Thinking* and *NoThinking*, respectively. Then we analyze the accuracy, response length, and instance-level pass rate across the five difficulty levels. As illustrated in Figure 2, although the model is trained using long-thinking data, *NoThinking* still achieves accuracy comparable to *Thinking* on relatively simple problems (Level 1 to 3), and even slightly outperforms *Thinking* on the easiest Level-1 problems. Meanwhile, the average length of *NoThinking* responses is significantly shorter than *Thinking* ones. Additionally, compared to *Nothinking*, *Thinking* only improves the instance-level pass rate for less than half of the problems from Level 1 to 4. Overall, these findings indicates that *Thinking* only brings notable benefits for challenging problems, whereas *NoThinking* can be a better choice for simpler questions in terms of both accuracy and efficiency. This motivates us to explore efficient reasoning from a new perspective: teaching the reasoning model to adaptively select *Thinking* or *NoThinking* mode based on problem difficulty, thereby reducing inference costs while maintaining or even improving the overall performance. To this end, we propose *AdaptThink*, a novel RL algorithm that teaches reasoning models when to think.

4 AdaptThink

Our *AdaptThink* algorithm consists of two important components: (1) a constrained optimization objective that incentivizes the model to select *NoThinking* mode, while ensuring the overall performance does not decrease; (2) an importance sampling strategy that balances *Thinking* and *NoThinking* samples during on-policy training, thereby enabling cold start and also allowing the model to explore and exploit both thinking modes through-

out the entire training process. We will introduce these two components in detail as follows.

4.1 Constrained Optimization Objective

Considering that *NoThinking* mode offers a significant advantage over *Thinking* in reasoning efficiency, an ideal selection policy should prefer to choose *NoThinking* as long as the overall performance is not diminished. In other words, we should maximize the probability of generating *NoThinking* responses while ensuring the model's accuracy does not decline.

Formally, consider a reasoning model π_θ and a dataset $\mathcal{D}$. Let $\pi_{\theta_{\text{ref}}}$ denote the reference model, which is the initial π_θ and remains unchanged during training. Let $R(x, y, y^*)$ be the reward function (i.e., accuracy in math solving), where x, y, and $\hat{y}$ denote the prompt, model response, and golden answer, respectively. The function returns 1 if y is both correct and properly formatted; otherwise, it returns 0. For simplicity, we omit $\hat{y}$ and denote the function as $R(x, y)$. Let $\mathbb{1}(y_1 = \texttt{</think>})$ be the indicator function, which returns 1 if the first token of y is `</think>` (i.e., y is a *NoThinking* response), otherwise returns 0. Then our optimization objective can be formulated as:

$$\begin{aligned} &\max \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x)} \mathbb{1}(y_1 = \texttt{</think>}) \\ &s.t.\ \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x)} R(x, y) \geq \\ &\qquad \mathbb{E}_{x\sim\mathcal{D}, y'\sim\pi_{\theta_{\text{ref}}}(\cdot|x)} R(x, y'). \end{aligned} \tag{2}$$

To solve this constrained optimization problem, we incorporate the constraint into the objective as a penalty term, with a penalty weight $\lambda \geq 0$:

$$\begin{aligned} &\max \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x), y'\sim\pi_{\theta_{\text{ref}}}(\cdot|x)} \mathbb{1}(y_1 = \texttt{</think>}) \\ &\quad + \lambda\big(R(x, y) - R(x, y')\big). \end{aligned} \tag{3}$$

By dividing the both side by λ, letting $\delta = \frac{1}{\lambda}$, and reorganizing the terms about $\pi_{\theta_{\text{ref}}}$, we have:

$$\begin{aligned} &\max \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x)} \mathbb{1}(y_1 = \texttt{</think>}) \cdot \delta \\ &\quad + R(x, y) - \mathbb{E}_{y'\sim\pi_{\theta_{\text{ref}}}(\cdot|x)} R(x, y'). \end{aligned} \tag{4}$$

In practice, $\mathbb{E}_{y'\sim\pi_{\theta_{\text{ref}}}(\cdot|x)} R(x, y')$ can be approximated by pre-sampling before training. Specifically, we sample K responses from $\pi_{\theta_{\text{ref}}}(\cdot|x)$ for each x, and calculate their mean reward:

$$\bar{R}_{\text{ref}}(x) = \frac{1}{K} \sum_{i=1}^{K} R(x, y'^i),\ y'^i \sim \pi_{\theta_{\text{ref}}}(\cdot|x). \tag{5}$$

Then the optimization objective becomes:

$$\begin{aligned} &\max \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x)} \mathbb{1}(y_1 = \texttt{</think>}) \cdot \delta \\ &\quad + R(x, y) - \bar{R}_{\text{ref}}(x). \end{aligned} \tag{6}$$

Since $\mathbb{1}(y_1 = \texttt{</think>})$ and $R(x, y)$ are not differentiable, we employ policy gradient method to solve this optimization. Specifically, let $\pi_{\theta_{\text{old}}}$ be a distribution equal to π_θ without gradient update, and define the advantage function: $A(x, y) = \mathbb{1}(y_1 = \texttt{</think>}) \cdot \delta + R(x, y) - \bar{R}_{\text{ref}}(x)$. Then the objective can be converted into a PPO-style (Schulman et al., 2017) loss without KL penalty:

$$\begin{aligned} \mathcal{L}(\theta) = -\mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_{\theta_{\text{old}}}(\cdot|x)} \Big[&\min \Big(\tfrac{\pi_\theta(y|x)}{\pi_{\theta_{\text{old}}}(y|x)} A(x, y), \\ &\text{clip}\big(\tfrac{\pi_\theta(y|x)}{\pi_{\theta_{\text{old}}}(y|x)}, 1-\epsilon, 1+\epsilon\big) A(x, y) \Big) \Big]. \end{aligned} \tag{7}$$

Here, $\text{clip}(\cdot)$ denotes the clipping function, which improves the stability of training.

4.2 Importance Sampling

At each step of optimizing $\mathcal{L}(\theta)$ using on-policy training, we sample a batch $\mathcal{D}_b$ from the dataset $\mathcal{D}$, and then sample K responses $\{y^i\}_{i=1}^K$ from $\pi_{\theta_{\text{old}}}(\cdot|x)$ for each $x \in \mathcal{D}_b$ to estimate $\mathcal{L}(\theta)$. However, since the initial π_θ naturally apply *Thinking* across all problems, it is impossible to get *Nothinking* samples from $\pi_{\theta_{\text{old}}}$ from the training starts (i.e., $\pi_{\theta_{\text{old}}}(y_1 = \texttt{</think>}|x) \approx 0$). As a result, the model can only learn from *Thinking* samples and will never generate *NoThinking* responses.

To solve this cold-start challenge, we employ the technique of importance sampling. Specifically, we define a new distribution $\pi_{\text{IS}}(\cdot|x)$:

$$\pi_{\text{IS}}(y_t = a|x, y_{<t}) = \begin{cases} 0.5, \text{ if } t = 1, a = \texttt{</think>}; \\ 0.5, \text{ if } t = 1, a = w_{\text{start}}; \\ \pi_{\theta_{\text{old}}}(y_t = a|x, y_{<t}), \text{ if } t > 1. \end{cases} \tag{8}$$

Here, w_{start} is a common word to start long thinking, such as *"Alright"*. During training, we sample responses $\{y^i\}_{i=1}^K$ from $\pi_{\text{IS}}(\cdot|x)$ instead of $\pi_{\theta_{\text{old}}}(\cdot|x)$, so that half of the samples in a batch are in *Thinking* mode and the other half are *NoThinking*. This allows the model to learn from both modes from the beginning of training, and finally adaptively be able to select the appropriate mode. Accordingly, our final loss function of *AdaptThink* becomes:

$$\begin{aligned} \mathcal{L}_{\text{AT}}(\theta) = -\mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_{\text{IS}}(\cdot|x)} \Big[&\min \Big(\tfrac{\pi_\theta(y|x)}{\pi_{\text{IS}}(y|x)} A(x, y), \\ &\text{clip}\big(\tfrac{\pi_\theta(y|x)}{\pi_{\text{IS}}(y|x)}, 1-\epsilon, 1+\epsilon\big) A(x, y) \Big) \Big]. \end{aligned} \tag{9}$$

Algorithm 1 *AdaptThink*

Input: policy model π_θ; dataset $\mathcal{D}$; hyperparameters K, δ, ϵ
Initialize: reference model $\pi_{\theta_{\text{ref}}} \leftarrow \pi_\theta$

Sample K responses $\{y'^i\}_{i=1}^K \sim \pi_{\theta_{\text{ref}}}(\cdot|x)$ and calculate $\bar{R}_{\text{ref}}(x)$ for each $x \in \mathcal{D}$ (Equation 5)
for step $= 1, \ldots, M$ **do**
  Update the old policy model $\pi_{\theta_{\text{old}}} \leftarrow \pi_\theta$ and importance sampling distribution π_{IS} (Equation 8)
  Sample a batch $\mathcal{D}_b$ from $\mathcal{D}$
  Sample K responses $\{y^i\}_{i=1}^K \sim \pi_{\text{IS}}(\cdot|x)$ for each $x \in \mathcal{D}_b$ and estimate $\mathcal{L}_{\text{AT}}(\theta)$ (Equation 9. Half of y^i are *Thinking* responses and the other half are *NoThinking* responses.)
  Update the policy model π_θ by minimizing $\mathcal{L}_{\text{AT}}(\theta)$
end for

Output: π_θ

In addition to enabling cold start, importance sampling also preserves the opportunities for exploration and exploitation across both *Thinking* and *NoThinking* modes during the entire training process. This prevents π_θ from collapsing into one thinking mode forever and completely ignoring the other, even if the latter mode may demonstrate a greater advantage in the future. Finally, we summarize our *AdaptThink* algorithm in Algorithm 1.

4.3 A New Perspective to Understand the Loss

In this subsection, we provide another perspective to understand our loss function $\mathcal{L}_{\text{AT}}(\theta)$ by comparing the advantage $A(x, y)$ of *Thinking* and *NoThinking* samples from $\pi_{\text{IS}}(\cdot|x)$. Given a prompt x, we denote the average pass rate of *Thinking* and *NoThinking* samples as $\bar{R}_{\text{think}}(x)$ and $\bar{R}_{\text{nothink}}(x)$, respectively. Then their average advantages are:

$$\bar{A}_{\text{think}}(x) = \bar{R}_{\text{think}}(x) - \bar{R}_{\text{ref}}(x),$$
$$\bar{A}_{\text{nothink}}(x) = \delta + \bar{R}_{\text{nothink}}(x) - \bar{R}_{\text{ref}}(x). \quad (10)$$

Note that the probability of choosing *NoThinking* (i.e., $\pi_\theta(y_1 = \texttt{</think>}|x)$) and *Thinking* (i.e., $\pi_\theta(y_1 = w^*|x)$) are competitive. Therefore, when optimizing $\mathcal{L}_{\text{AT}}(\theta)$, $\pi_\theta(y_1 = \texttt{</think>}|x)$ will improve only if $\bar{A}_{\text{nothink}}(x) > 0$ and $\bar{A}_{\text{nothink}}(x) > \bar{A}_{\text{think}}(x)$, which give us:

$$\bar{R}_{\text{nothink}}(x) + \delta > \bar{R}_{\text{ref}}(x),$$
$$\bar{R}_{\text{nothink}}(x) + \delta > \bar{R}_{\text{think}}(x). \quad (11)$$

In other words, only when the problem is simple enough such that the accuracy gap between *NoThinking* and *Thinking*, as well as the reference model, is smaller than δ, $\mathcal{L}_{\text{AT}}(\theta)$ will favor *NoThinking* and encourage π_θ to directly generate the final solution. For more challenging problems where *NoThinking* lags far behind the other two, $\mathcal{L}_{\text{AT}}(\theta)$ will prioritize performance and guide π_θ to engage in *Thinking* more frequently. Therefore, $\mathcal{L}_{\text{AT}}(\theta)$ aligns well with our expectation for difficulty-adaptive thinking in Section 3.2.

5 Experiments

5.1 Setup

Models. We select DeepSeek-R1-Distill-Qwen-1.5B and DeepSeek-R1-Distill-Qwen-7B, two popular reasoning models that demonstrate impressive performance on math problem solving, as the initial policy models.

Dataset and Metrics. The training dataset we use is DeepScaleR (Luo et al., 2025b) dataset, which consists of 40K math problems drawn from AIME 1983-2023, AMC, Omni-Math (Gao et al., 2024), and STILL (Min et al., 2024). For evaluation, we use three math datasets with increasing difficulty: GSM8K (Cobbe et al., 2021) test set (1319 grade school math problems), MATH500 (Lightman et al., 2024) (500 high-school competition math problems), and AIME 2024 (30 Olympiad-level math problems). For evaluation metrics, we consider both accuracy and response length. We also report the average accuracy variation and the average length reduction rate across all the test datasets. Considering the limited size of AIME 2024, we repeatedly sample 16 responses for each case and report the average results. For all models, we set the evaluation context size to 16K, and set the temperature to 0.6 as suggested in DeepSeek's model cards.

Implementation Details. We build our code based on VeRL (Sheng et al., 2024) framework. The training context size, batch size, and the learning rate are set to 16K, 128, and 2e-6, respectively. The hyperparameters K, δ, and ϵ in *AdaptThink* are set to 16, 0.05, and 0.2, respectively. The comparison of using different δ is shown in Section 5.4. We train the models for 1 epoch, which is 314 steps in total. For the 1.5B model, we use one 8×H800 node and cost about 32 hours. For the 7B model, we use four 8×H800 nodes and cost about 28 hours. Finally, we select the checkpoints on 300 and 150

Method	GSM8K			MATH 500			AIME 2024			Average	
	Acc	Length	Ratio_{NT}	Acc	Length	Ratio_{NT}	Acc	Length	Ratio_{NT}	ΔAcc	ΔLength
DeepSeek-R1-Distill-Qwen-1.5B											
$\text{Original}_{Thinking}$	79.0	978	0.0%	80.6	4887	0.0%	29.4	12073	0.0%	-	-
$\text{Original}_{NoThinking}$	69.8	280	100.0%	67.2	658	100.0%	14.0	2190	100.0%	-12.7	-79.9%
$\text{DPO}_{Shortest}$	78.3	804	0.0%	82.4	3708	0.0%	30.7	10794	0.0%	+0.8	-17.5%
OverThink	77.2	709	0.0%	81.2	4131	0.0%	28.3	11269	0.0%	-0.8	-16.5%
DAST	77.2	586	0.0%	83.0	2428	0.0%	26.9	7745	0.0%	-0.6	-42.1%
O1-Pruner	74.8	**458**	0.0%	82.2	3212	0.0%	28.9	10361	0.0%	-1.0	-33.9%
TLMRE	80.7	863	0.0%	**85.0**	3007	0.0%	29.2	8982	0.0%	+2.0	-25.3%
ModelMerging	79.7	603	0.0%	63	2723	0.0%	18.1	10337	0.0%	-9.4	-32.3%
$\text{RFT}_{MixThinking}$	76	1077	8.8%	72.4	4341	33.4%	25.2	11157	21.0%	-5.1	-2.9%
AdaptThink	**83.1**	480	86.9%	82.0	**1782**	76.8%	**31.0**	**6679**	40.4%	**+2.4**	**-53.0%**
DeepSeek-R1-Distill-Qwen-7B											
$\text{Original}_{Thinking}$	87.9	682	0.0%	90.2	3674	0.0%	53.5	10306	0.0%	-	-
$\text{Original}_{NoThinking}$	85.1	283	100.0%	80.6	697	100.0%	24.2	1929	100.0%	-13.9	-73.6%
$\text{DPO}_{Shortest}$	85.7	402	0.0%	91.6	2499	0.0%	52.5	8699	0.0%	-0.6	-29.5%
OverThink	86.3	426	0.0%	89.4	2435	0.0%	53.1	8744	0.0%	-0.9	-28.8%
DAST	86.7	459	0.0%	89.6	2162	0.0%	45.6	**7578**	0.0%	-3.2	-33.4%
O1-Pruner	87.6	428	0.0%	86.6	2534	0.0%	49.2	9719	0.0%	-2.7	-24.7%
TLMRE	88.9	756	0.0%	91.8	2899	0.0%	54.0	8633	0.0%	+1.0	-8.8%
ModelMerging	88.4	531	0.0%	72.6	2280	0.0%	36.9	8624	0.0%	-11.2	-25.5%
$\text{RFT}_{MixThinking}$	86.2	365	66.5%	84.8	2411	64.8%	49.4	9969	10.0%	-3.7	-28.0%
AdaptThink	**91.0**	**309**	99.6%	**92.0**	**1875**	76.6%	**55.6**	8599	6.3%	**+2.3**	**-40.1%**

Table 1: Accuracy (Acc), response length (Length), and the ratio of *NoThinking* responses (Ratio_{NT}) of different methods on three math benchmarks. The best and second results are bolded and underlined, respectively.

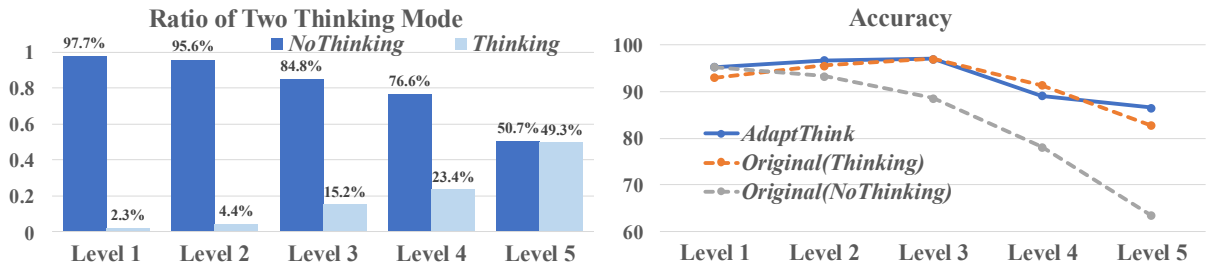

Figure 3: Left: The ratio that *AdaptThink*-7B choose *Thinking* or *NoThinking* across different MATH levels. Right: Comparison of accuracy between *AdaptThink*-7B and DeepSeek-R1-Distill-Qwen-7B using *Thinking* and *NoThinking* across different MATH levels.

steps for the 1.5B and 7B models, respectively, where the models' accuracy and response lengths achieve a good balance.

5.2 Baselines

We compare *AdaptThink* with the following representative methods for efficient reasoning:

- **$\text{DPO}_{Shortest}$** constructs preference data by sampling multiple responses for each problem in the training dataset and pairing the shortest correct response and the longest responses, then uses DPO (Rafailov et al., 2023) to finetune the model.
- **OverThink** (Chen et al., 2024) first constructs preference data by taking the original long-thinking response for each training problem as the negative example and retaining the first two attempts that arrive at the correct answer in the thinking as the positive example, and then uses SimPO (Meng et al., 2024) to alleviate models' overthinking behaviors.
- **DAST** (Shen et al., 2025) first constructs preference data by ranking pre-sampled responses with a length-based reward function, and then employs SimPO to finetune the model.
- **O1-Pruner** (Luo et al., 2025a) first estimates the reference model's performance through pre-sampling and then uses off-policy RL-style fine-tuning to encourage the model to generate shorter reasoning processes under accuracy constraints.
- **TLMRE** (Arora and Zanette, 2025) incorporates a length-based penalty in on-policy RL to incentivize the model to produce shorter responses.
- **ModelMerging** (Wu et al., 2025) reduces the response length of a reasoning model by weightedly averaging its weights with a non-reasoning model (i.e., Qwen-2.5-Math-1.5B/7B).

Method	GSM8K			MATH500			AIME 2024			Average	
	Acc	Length	Ratio$_{NT}$	Acc	Length	Ratio$_{NT}$	Acc	Length	Ratio$_{NT}$	ΔAcc	ΔLength
DeepSeek-R1-Distill-Qwen-1.5B											
Original$_{Thinking}$	79.0	978	0.0%	80.6	4887	0.0%	29.4	12073	0.0%	-	-
Original$_{NoThinking}$	69.8	280	100.0%	67.2	658	100.0%	14.0	2190	100.0%	-12.7	-79.9%
AdaptThink-1.5B											
$\delta=0$	84.6	718	70.4%	86	2511	50.0%	34.8	9279	0.2%	+5.5	-32.8%
$\delta=0.01$	82.4	638	55.6%	82.4	2473	67.0%	32.5	9165	19.8%	+2.8	-36.1%
$\delta=0.02$	83.1	628	75.7%	83.4	2337	62.6%	31.3	8696	21.3%	+2.9	-38.6%
$\delta=0.05$	83.1	480	86.9%	82	1782	76.8%	31	6679	40.4%	+2.4	-53.0%
$\delta=0.075$	83.2	580	71.8%	80.2	1621	84.2%	29.2	6087	64.2%	+1.2	-52.4%
$\delta=0.1$	82.5	358	91.7%	78.2	1272	90.4%	26.7	5301	83.5%	-0.5	-64.5%

Table 2: Performance of *AdaptThink*-1.5B using different δ value.

- **RFT$_{MixThinking}$** (Reject Fine-tuning) first samples multiple responses for each training problem x using both *Thinking* and *NoThinking*, then selects (1) correct *NoThinking* responses if the instance-level pass rate $\bar{R}_{\text{nothink}}(x) \geq \bar{R}_{\text{think}}(x)$ and (2) correct *Thinking* responses if $\bar{R}_{\text{nothink}}(x) < \bar{R}_{\text{think}}(x)$, and uses these selected responses to finetune the model.

For fair comparison, we re-implement all these baselines using DeepScaleR dataset.

5.3 Main Results

Table 1 presents the evaluation results of different methods on GSM8K, MATH500, and AIME 2024. Compared to the original 1.5B and 7B models, *AdaptThink* reduces the average response length by 53.0% and 40.1%, respectively, while also improves the average accuracy by 2.4% and 2.3%, demonstrating that *AdaptThink* enables significantly more efficient reasoning without compromising and even enhancing model performance. Moreover, *AdaptThink* outperforms most baselines—all of which optimize response length within the *Thinking* mode—in terms of both accuracy and length reduction. It also achieves the best average results, highlighting the effictiveness of adaptive thinking-mode selection as a novel paradigm for achieving efficient reasoning.

For the methods that involve both *Thinking* and *NoThinking* modes, we additionally report the ratio of responses generated in *NoThinking* mode (i.e., Ratio$_{NT}$ in Table 1). As shown in the table, *AdaptThink* produces more *NoThinking* responses for easier test sets (i.e., GSM8K and MATH500) while employing *Thinking* mode more frequently for challenging test sets (i.e., AIME 2024). See Appendix A for detailed cases. A similar trend is also observed within the five difficulty levels of MATH500 (Figure 3 and 5), where *AdaptThink* predominantly selects the *NoThinking* mode for the easiest Level-1 problems, and progressively increases the use of *Thinking* as the difficulty level rises. Meanwhile, compared to the original models using only *Thinking* or *NoThinking* mode, *AdaptThink* consistently achieves higher accuracy across most difficulty levels. These findings suggest that *AdaptThink* has successfully taught the model to adaptively choose an appropriate thinking mode based on problem difficulty, achieving a better balance between efficiency and performance.

5.4 More Analyses

Effect of δ. To show the effect of δ in our advantage function $A(x, y)$, we implement *AdaptThink* with different δ values on the 1.5B model and summarize the evaluation results in Table 2. As δ increases, the proportion of *NoThinking* responses progressively rises, resulting in a corresponding reduction in the average response length. However, the gain in accuracy also gradually decreases at the same time. This indicates that δ serves as a control parameter for the trade-off between reasoning efficiency and accuracy improvement. Notably, even when $\delta=0$, the model chooses *NoThinking* for over 50% of problems in GSM8K and MATH500, implying that *NoThinking* may possess an inherent advantage over *Thinking* when addressing relatively straightforward problems. Furthermore, for most δ values, *AdaptThink* consistently achieves notable reduction in response length and also improves accuracy, which underscores its robustness.

Effect of importance sampling. To demonstrate the effect of importance sampling, we compare *AdaptThink* with naive GRPO that samples directly from $\pi_{\theta_{\text{old}}}(\cdot|x)$ during training. As shown in Figure 4, because $\pi_{\theta_{\text{old}}}(\cdot|x)$ is initially unable

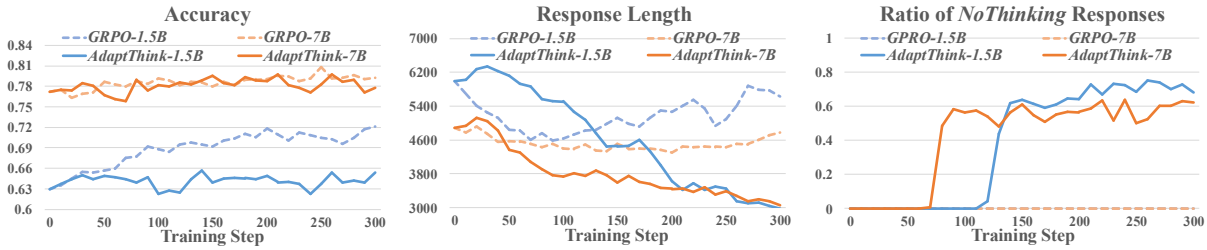

Figure 4: Comparison of average accuracy, average response length, and the ratio of *NoThinking* responses on the three math test sets between *AdaptThink* and naive GPRO at different training steps.

Method	Ratio_{IT}	Length
DeepSeek-R1-Distill-Qwen-1.5B		
Final Solutions from $\text{Original}_{Thinking}$	9.5%	1799
$\text{Original}_{NoThinking}$	8.2%	665
AdaptThink	7.9%	826
DeepSeek-R1-Distill-Qwen-7B		
Final Solutions from $\text{Original}_{Thinking}$	0.7%	321
$\text{Original}_{NoThinking}$	0.9%	341
AdaptThink	4.2%	426

Table 3: For the test cases where *AdaptThink* chooses *NoThinking*, we compare the implicit thinking ratio (Ratio_{IT}) and average length of three scenarios.

Method	MMLU				
	Acc	Length	Ratio_{NT}	ΔAcc	ΔLength
DeepSeek-R1-Distill-Qwen-1.5B					
$\text{Original}_{Thinking}$	35.7	1724	0.00%	-	-
$\text{Original}_{NoThinking}$	20.6	208	100.00%	-15.1	-87.9%
AdaptThink	42.2	1055	16.43%	+6.5	-38.8%
DeepSeek-R1-Distill-Qwen-7B					
$\text{Original}_{Thinking}$	63.4	1257	0.00%	-	-
$\text{Original}_{NoThinking}$	51.2	128	100.00%	-12.2	-89.8%
AdaptThink	63.6	856	16.41%	+0.2	-31.9%

Table 4: The performance of *AdaptThink* on the out-of-distribute test set MMLU.

to produce *NoThinking* samples, GRPO can only learn from *Thinking* samples and focus on improving accuracy throughout the training process. The response length of GRPO decreases only to around 4,600 (by eliminating overlong responses that would be truncated and receive no reward), after which it gradually increases. In contrast, our importance sampling strategy enables *AdaptThink* to learn from both *Thinking* and *NoThinking* samples at each training step. As the model gradually learn to generate more *NoThinking* responses for simple problems, the response length eventually decreases to below 3,000 tokens.

Implicit thinking ratio. A potential concern for *AdaptThink* is that RL may activate thinking features (e.g., reflection) within *NoThinking* mode (similar to DeepSeek-R1-Zero) and produce many implicit thinking responses. To allay this concern, we examine the test cases where *AdaptThink* chooses the *NoThinking* mode. For these cases, we collect (1) *NoThinking* responses from *AdaptThink*, (2) *NoThinking* responses from the original reasoning model, and (3) the final solution part of *Thinking* responses from the original model. We compare the ratio of implicit thinking responses (denoted by Ratio_{IT}) across these three scenarios by detecting whether some representative keywords for thinking (e.g, *"Wait"* and *"Alternatively"*) appear in the solutions. We also compare the average length of these solutions. As presented in Table 3, *AdaptThink* only slightly increases the implicit thinking ratio and response length for the 7B model. To entirely eliminate such behavior, one possible approach is to assign zero reward to implicit thinking samples during RL training.

Generalizability to OOD scenario. To assess the ability of *AdaptThink* to generalize in out-of-distribution scenarios, we conduct an evaluation on MMLU, which contains 14K multi-choice questions and covers 57 diverse domains, distinct from our training data in question format and subjects. As shown in Table 4, *AdaptThink* reduces average response length by more than 30% by producing *NoThinking* responses for about 16% of the problems (see Figure 8 for cases), while achieving higher accuracy than the original models.

6 Conclusion

In this work, we first demonstrate the advantages of *NoThinking* over *Thinking* in both performance and efficiency for relatively simple tasks. Motivated by this, we propose *AdaptThink*, a novel RL algorithm to enable reasoning models to adaptively select the optimal thinking mode based on problem difficulty. Experiments show that *AdaptThink* significantly reduces inference costs and further improves model performance, highlighting the promise of adaptive thinking-mode selection for advancing the trade-off between reasoning quality and efficiency.

7 Limitation

We discuss several limitations of our work in this section: (1) Due to limited computational resources, we only conduct our experiments on 1.5B and 7B models. Nevertheless, these experiments still demonstrate the efficacy of *AdaptThink* across different model sizes. (2) Similar to most previous open-sourced works, we only train our models using mathematical datasets because they are easy to obtain and can offer accurate, verifiable rewards. Though our evaluation on MMLU shows *AdaptThink* models can well generalize to OOD scenarios, we believe they can achieve better results if more training datasets with verifiable rewards for general domains are available.

8 Ethical Considerations

All the models and datasets used in this work are publicly published with permissible licenses.

References

Pranjal Aggarwal and Sean Welleck. 2025. L1: controlling how long A reasoning model thinks with reinforcement learning. *CoRR*, abs/2503.04697.

Daman Arora and Andrea Zanette. 2025. Training language models to reason efficiently. *CoRR*, abs/2502.04463.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2024. Do NOT think that much for 2+3=? on the overthinking of o1-like llms. *CoRR*, abs/2412.21187.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948.

Yichao Fu, Junda Chen, Yonghao Zhuang, Zheyu Fu, Ion Stoica, and Hao Zhang. 2025. Reasoning without self-doubt: More efficient chain-of-thought through certainty probing. In *ICLR 2025 Workshop on Foundation Models in the Wild*.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, Zhengyang Tang, Benyou Wang, Daoguang Zan, Shanghaoran Quan, Ge Zhang, Lei Sha, Yichang Zhang, Xuancheng Ren, Tianyu Liu, and Baobao Chang. 2024. Omni-math: A universal olympiad level mathematic benchmark for large language models. *CoRR*, abs/2410.07985.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. 2024. Token-budget-aware LLM reasoning. *CoRR*, abs/2412.18547.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Haotian Luo, Li Shen, Haiying He, Yibo Wang, Shiwei Liu, Wei Li, Naiqiang Tan, Xiaochun Cao, and Dacheng Tao. 2025a. O1-pruner: Length-harmonizing fine-tuning for o1-like reasoning pruning. *CoRR*, abs/2501.12570.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. 2025b. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. Notion Blog.

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

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. Simpo: Simple preference optimization with a reference-free reward. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Yingqian Min, Zhipeng Chen, Jinhao Jiang, Jie Chen, Jia Deng, Yiwen Hu, Yiru Tang, Jiapeng Wang, Xiaoxue Cheng, Huatong Song, Wayne Xin Zhao, Zheng Liu, Zhongyuan Wang, and Ji-Rong Wen. 2024. Imitate, explore, and self-improve: A reproduction report on slow-thinking reasoning systems. *CoRR*, abs/2412.09413.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel J. Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling. *CoRR*, abs/2501.19393.

OpenAI. 2024. Learning to reason with llms. https://openai.com/index/learning-to-reason-with-llms/. Accessed: 2025-05-07.

Xiaoye Qu, Yafu Li, Zhaochen Su, Weigao Sun, Jianhao Yan, Dongrui Liu, Ganqu Cui, Daizong Liu, Shuxian Liang, Junxian He, Peng Li, Wei Wei, Jing Shao, Chaochao Lu, Yue Zhang, Xian-Sheng Hua, Bowen

Zhou, and Yu Cheng. 2025. A survey of efficient reasoning for large reasoning models: Language, multimodality, and beyond. *CoRR*, abs/2503.21614.

Qwen Team. 2025. Qwq-32b-preview. https://qwenlm.github.io/blog/qwq-32b-preview/. Accessed: 15 March 2025.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347.

Yi Shen, Jian Zhang, Jieyun Huang, Shuming Shi, Wenjing Zhang, Jiangze Yan, Ning Wang, Kai Wang, and Shiguo Lian. 2025. DAST: difficulty-adaptive slow-thinking for large reasoning models. *CoRR*, abs/2503.04472.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Hanjie Chen, and Xia Ben Hu. 2025. Stop overthinking: A survey on efficient reasoning for large language models. *CoRR*, abs/2503.16419.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Fengxiang Tang, Flood Sung, Guangda Wei, Guokun Lai, and 75 others. 2025. Kimi k1.5: Scaling reinforcement learning with llms. *CoRR*, abs/2501.12599.

Han Wu, Yuxuan Yao, Shuqi Liu, Zehua Liu, Xiaojin Fu, Xiongwei Han, Xing Li, Hui-Ling Zhen, Tao Zhong, and Mingxuan Yuan. 2025. Unlocking efficient long-to-short LLM reasoning with model merging. *CoRR*, abs/2503.20641.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. 2025. Chain of draft: Thinking faster by writing less. *CoRR*, abs/2502.18600.

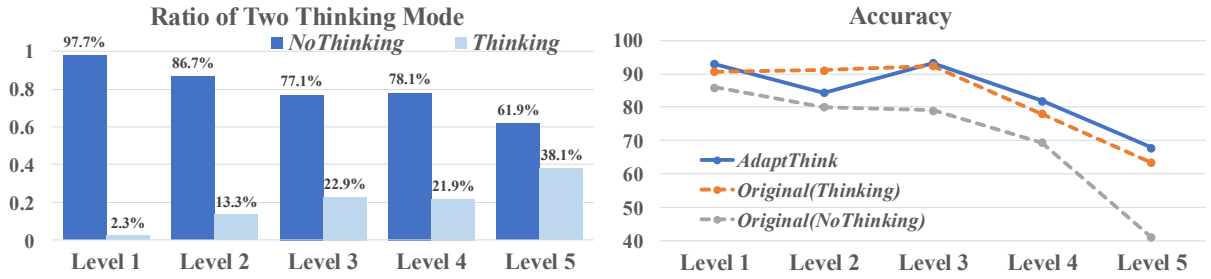

Figure 5: Left: The ratio that *AdaptiveThink*-1.5B choose *Thinking* or *NoThinking* across different MATH levels. Right: Comparison of accuracy between *AdaptiveThink*-1.5B and the DeepSeek-R1-Distill-Qwen-1.5B using *Thinking* and *NoThinking* across different MATH levels.

A Case Study

We show several cases of *AdaptThink* in Figure 6, 7, 8, and 9 to illustrate its ability of adaptive thinking-mode selection. As demonstrated in Figure 6, 7, and 8, when faced with relatively simple problems, DeepSeek-R1-Distill-Qwen-7B costs a large number of tokens in thinking, which often contains many unnecessary steps and redundant attempts. In contrast, our *AdaptThink*-7B model adaptively chooses *NoThinking* mode and directly produces a concise final solution, even if the input problem differs from our training data in both the question format and the subject. On the hard hand, the case in Figure 9 indicates that, when the input problem is difficult, *AdaptThink*-7B is also able to automatically employ *Thinking* to ensure its accuracy.

Question (GSM8K): Two cars are driving on a highway. The first car is traveling at an average speed of 60 miles per hour when the second car passes it at an average speed of 70 miles per hour. If both cars continue on the highway at the same speed, how many miles will separate them after 2 hours?

DeepSeek-R1-Distill-Qwen-7B:
<think>Okay, so I have this problem here about two cars on a highway. Let me try to understand it step by step. The first car is going at an average speed of 60 miles per hour. Then, the second car passes the first car, and it's going at 70 miles per hour. The question is asking, if both cars keep going at the same speed, how many miles apart will they be after 2 hours. Hmm, okay. So, let me visualize this. There's a highway, and both cars are moving in the same direction. The first car is ahead, going 60 mph, and the second car catches up to it, going 70 mph. So, the second car is faster, right? It's going 10 mph faster than the first car. Wait, so the second car is approaching the first car at a rate of 10 miles per hour. That makes sense because 70 minus 60 is 10. So, the gap between them is closing at 10 mph. But the question is asking, after 2 hours, how far apart will they be. Hmm, wait a second. If the second car is catching up, the distance between them is decreasing. But the question says, how many miles will separate them after 2 hours. So, is it possible that the second car overtakes the first car before 2 hours, and then the distance becomes positive again? Or maybe they're both moving in the same direction, so the second car is just gaining on the first car, but after 2 hours, the distance between them is how much? Wait, **[...](omit 2679 tokens)** so, I think I'm confident that the answer is 20 miles.</think> **Final Answer:** The number of miles that will separate them after 2 hours is $\boxed{20}$. The first car is traveling at an average speed of 60 miles per hour. The second car passes it at an average speed of 70 miles per hour. Both cars continue on the highway at the same speed. We need to determine how many miles will separate them after 2 hours. 1. The relative speed between the second car and the first car is the difference in their speeds: $70\,\text{mph} - 60\,\text{mph} = 10\,\text{mph}$. 2. The separation between the two cars after time t hours is given by the relative speed multiplied by time: Separation $= 10\,\text{mph} \times t$ hours. 3. After 2 hours, the separation will be: Separation $= 10\,\text{mph} \times 2\,\text{hours} = 20\,\text{miles}$. Thus, the number of miles that will separate them after 2 hours is $\boxed{20}$. **(3303 tokens)**

AdaptThink-7B (ours):
<think></think>To determine how many miles will separate the two cars after 2 hours, let's break down the problem step by step. 1. **Understand the Speeds:** The first car is traveling at 60 miles per hour (mph). The second car is traveling at 70 mph. 2. **Determine the Relative Speed:** Since both cars are moving in the same direction, the relative speed between them is the difference in their speeds. Relative speed $=$ Speed of second car $-$ Speed of first car $= 70\ \text{mph} - 60\ \text{mph} = 10\ \text{mph}$. 3. **Calculate the Distance Separated After 2 Hours:** Distance $=$ Speed $\times$ Time. Here, the relative speed is $10\ \text{mph}$, and the time is 2 hours. So, Distance $= 10\ \text{mph} \times 2\ \text{hours} = 20\ \text{miles}$. Therefore, after 2 hours, the two cars will be $\boxed{20}$ miles apart. **(222 tokens)**

Figure 6: For a simple math problem from GSM8K, DeepSeek-R1-Distill-Qwen-7B costs about 3000 tokens in thinking, which contains many unnecessary steps and redundant attempts. In contrast, our *AdaptThink*-7B adaptively chooses *NoThinking* mode and directly produces a concise final solution.

Question (MATH500): In how many ways can 8 people sit around a round table if 3 of the people – Pierre, Rosa, and Thomas – all want to sit together? (Two seatings are considered the same if one is a rotation of the other.)

DeepSeek-R1-Distill-Qwen-7B:
<think>Alright, so I have this problem where I need to figure out how many ways 8 people can sit around a round table, but there's a condition: Pierre, Rosa, and Thomas all want to sit together. Hmm, okay. Let me think about this step by step. First, I remember that when arranging people around a round table, the number of arrangements is different from arranging them in a straight line because rotations of the same arrangement are considered identical. For a round table with n people, the number of distinct seating arrangements is $(n-1)!$ instead of $n!$. That's because fixing one person's position accounts for the rotations, and then arranging the others relative to that fixed position. But in this case, there's a condition: Pierre, Rosa, and Thomas must all sit together. So, how does that affect the calculation? I think the way to handle this is to treat the group of Pierre, Rosa, and Thomas as a single entity or "block." If we consider them as one block, then we're effectively arranging this block along with the other 5 people. So, that makes a total of 6 "blocks" to arrange around the table. Wait, hold on. If we have 8 people and we're grouping 3 of them together, how many people are left? Let me subtract: $8 - 3 = 5$. So, yes, we have 5 individuals plus the block, totaling 6 units to arrange around the table. Now, since it's a round table, the number of ways to arrange these 6 units is $(6-1)! = 5! = 120$. That's because, again, fixing one position accounts for the rotational symmetry. But wait, **[...](omit 9138 tokens)** Therefore, I think the answer is 720. </think> **Final Answer:** The number of ways is $\boxed{720}$. To determine the number of ways 8 people can sit around a round table with the condition that Pierre, Rosa, and Thomas must sit together, we can treat them as a single block. This reduces the problem to arranging 6 entities (the block and the other 5 people) around a circular table. Treat the group as a single block: This gives us 6 entities (the block and 5 individuals). Arrange these 6 entities around a circular table: The number of distinct arrangements of 6 entities around a circular table is $(6-1)! = 5!$. Arrange the people within the block: The 3 people in the block can be arranged among themselves in $3!$ ways. Thus, the total number of arrangements is the product of these two results: $(6-1)! \times 3! = 5! \times 3! = 120 \times 6 = 720$ Therefore, the number of ways 8 people can sit around a round table with Pierre, Rosa, and Thomas sitting together is $\boxed{720}$. **(9812 tokens)**

AdaptThink-7B (ours):
<think></think>To determine the number of ways 8 people can sit around a round table with the condition that Pierre, Rosa, and Thomas must all sit together, we can treat the group of three as a single entity or "block." Here's the step-by-step solution: **1. Treat Pierre, Rosa, and Thomas as a single block:** This reduces the problem to arranging 6 entities around a round table: the block and the other 5 individuals. **2. Arrange the 6 entities around the table:** For circular arrangements, the number of ways to arrange n entities is $(n-1)!$. So, for 6 entities, it's $(6-1)! = 5! = 120$ ways. **3. Arrange Pierre, Rosa, and Thomas within their block:** The three people can be arranged among themselves in $3! = 6$ ways. **4. Multiply the two results together:** The total number of arrangements is the product of the arrangements of the blocks and the internal arrangements of the group: $5! \times 3! = 120 \times 6 = 720$. So, the number of ways 8 people can sit around a round table with Pierre, Rosa, and Thomas all sitting together is $\boxed{720}$. **(297 tokens)**

Figure 7: For a relatively simple math problem from MATH500, DeepSeek-R1-Distill-Qwen-7B costs about 9000 tokens in thinking. In contrast, our *AdaptThink*-7B adaptively chooses *NoThinking* mode and directly produces a concise final solution, using only 297 tokens.

Question (MMLU): A proposed mechanism for a reaction is as follows: $NO_2 + F_2 \rightarrow NO_2F + F$ (Slow step). $F + NO_2 \rightarrow NO_2F$ (Fast step). What is the order of the overall reaction? A) Zero order. B) First order. C) Second order. D) Third order. Output the correct option (one of ABCD) in "\boxed{}", such as "\boxed{A}".

DeepSeek-R1-Distill-Qwen-7B:
<think>Okay, so I have this chemistry problem here about reaction mechanisms and determining the overall order of the reaction. Hmm, let me try to figure this out step by step. The problem says the proposed mechanism is: 1. $NO_2 + F_2 \rightarrow NO_2F + F$ (slow step) 2. $F + NO_2 \rightarrow NO_2F$ (fast step) And the question is asking for the order of the overall reaction. The options are A. Zero order, B. First order, C. Second order, D. Third order. Alright, I remember that the overall order of a reaction is determined by the sum of the orders of each reactant in the rate law. But wait, the rate law isn't always given. In this case, the mechanism is provided, so I need to use that to figure out the rate law. First, in a reaction mechanism, the slow step is the rate-determining step, right? That means the rate law is based on that step. So I should focus on the first step: $NO_2 + F_2 \rightarrow NO_2F + F$. Looking at the slow step, the reactants are NO_2 and F_2. So the rate law would be $\text{rate} = k[NO_2][F_2]^m$, where m is the order with respect to F_2. But wait, in the slow step, F_2 is a reactant, but in the second step, there's an F that's produced. Hmm, I think that in the overall reaction, the intermediates (like F in this case) should be canceled out to get the rate law in terms of the initial reactants. Hold on, **[...](omit 3887 tokens)** Therefore, the answer is C. Second order. **Final Answer** $\boxed{C}$ </think> The proposed mechanism for the reaction is as follows: 1. $NO_2 + F_2 \rightarrow NO_2F + F$ (slow step) 2. $F + NO_2 \rightarrow NO_2F$ (fast step) The overall reaction is determined by combining the two steps: $NO_2 + F_2 + F + NO_2 \rightarrow NO_2F + F + NO_2F$. Simplifying, we get: $2\,NO_2 + F_2 \rightarrow 2\,NO_2F$. The rate law is determined by the slow step, which is the rate-determining step. The rate law for the slow step is: $\text{rate} = k[NO_2][F_2]$. Since F is an intermediate, we use the steady-state approximation to express [F] in terms of $[NO_2]$ and $[F_2]$. From the fast step, we find: $k_1[NO_2][F_2] = k_2[F][NO_2]$. Solving for [F]: $[F] = \frac{k_1}{k_2}[F_2]$. Substituting this back into the rate law confirms that the rate is still first order in $[NO_2]$ and first order in $[F_2]$. Therefore, the overall order of the reaction is: $1+1 = 2$. Thus, the order of the overall reaction is second order. $\boxed{C}$ **(4709 tokens)**

AdaptThink-7B (ours):
<think></think>To determine the order of the overall reaction, we analyze the given mechanism: **1. Slow Step (Rate-Determining Step):** $NO_2 + F_2 \rightarrow NO_2F + F$ The rate law for this step is: Rate $= k[NO_2][F_2]$.. **2. Fast Step:** $F + NO_2 \rightarrow NO_2F$. Since this step is fast, it does not affect the overall rate law. The overall rate law is determined by the slow step. Combining the rate laws from the slow step, the overall reaction order is the sum of the exponents in the rate law: the order with respect to NO_2 is 1, and the order with respect to F_2 is 1. Thus, the overall reaction order is $1 + 1 = 2$. The correct answer is $\boxed{C}$. **(263 tokens)**

Figure 8: Even for out-of-distribution problems from MMLU, *AdaptThink*-7B is still able to generate *NoThinking* response if the problem is relatively simple.

Question (AIME 2024): Find the largest possible real part of $(75+117i)z+\frac{96+144i}{z}$ where z is a complex number with $|z|=4$.

DeepSeek-R1-Distill-Qwen-7B:
<think>Alright, so I have this problem here: I need to find the largest possible real part of the expression $(75+117i)z+\frac{96+144i}{z}$ where z is a complex number with $|z|=4$. Hmm, okay, let me try to figure this out step by step. First, I remember that for complex numbers, the real part of an expression is the sum of the real parts of each term. So, I can probably break this down into two parts: the real part of $(75+117i)z$ and the real part of $\frac{96+144i}{z}$. Then, **[...](omit 3099 tokens)** I think that solidifies the answer. I don't see any mistakes in the reasoning. **Final Answer** $\boxed{540}$ </think> Given the problem of finding the largest possible real part of the expression $(75+117i)z+\frac{96+144i}{z}$ where z is a complex number with $|z|=4$, we start by expressing z as $z=4e^{i\theta}$. This allows us to rewrite the expression in terms of θ. First, we substitute $z=4e^{i\theta}$ and $\frac{1}{z}=\frac{1}{4}e^{-i\theta}$ into the expression: $(75+117i)\cdot 4e^{i\theta}+(96+144i)\cdot\frac{1}{4}e^{-i\theta}$. This simplifies to $(300+468i)e^{i\theta}+(24+36i)e^{-i\theta}$. Expanding each term using Euler's formula $e^{i\theta}=\cos\theta+i\sin\theta$ and $e^{-i\theta}=\cos\theta-i\sin\theta$, we separate the real and imaginary parts. After combining the real parts, we get $324\cos\theta-432\sin\theta$. To find the maximum value of this expression, we recognize it as a linear combination of sine and cosine, which can be written in the form $A\cos\theta+B\sin\theta$. The maximum value of such an expression is given by the amplitude $R=\sqrt{A^2+B^2}$. Calculating R: $R=\sqrt{324^2+(-432)^2}=\sqrt{104976+186624}=\sqrt{291600}=540$. Thus, the largest possible real part of the given expression is $\boxed{540}$. **(3771 tokens)**

AdaptThink-7B (ours):
<think>Okay, so I have this problem here: I need to find the largest possible real part of the expression $(75+117i)z+\frac{96+144i}{z}$, where z is a complex number with $|z|=4$. Hmm, okay. Let me try to figure this out step by step. First, I know that z is a complex number with magnitude 4. That means **[...](omit 3003 tokens)** Therefore, the largest possible real part is $\boxed{540}$. </think> Given the expression $(75+117i)z+\frac{96+144i}{z}$ where z is a complex number with $|z|=4$, we need to find the largest possible real part. First, we express z as $4e^{i\theta}$ where θ is the argument of z. Then, we rewrite the expression in terms of θ: $E=(75+117i)4e^{i\theta}+\frac{96+144i}{4}e^{-i\theta}$. Simplifying the constants, we get: $(75+117i)4=300+468i$ and $\frac{96+144i}{4}=24+36i$. Thus, the expression becomes: $E=(300+468i)e^{i\theta}+(24+36i)e^{-i\theta}$. Using Euler's formula, we expand and separate into real and imaginary parts: $(300+468i)(\cos\theta+i\sin\theta)=(300\cos\theta-468\sin\theta)+i(300\sin\theta+468\cos\theta)$ and $(24+36i)(\cos\theta-i\sin\theta)=(24\cos\theta+36\sin\theta)+i(-24\sin\theta+36\cos\theta)$. Adding the real parts together: $300\cos\theta-468\sin\theta+24\cos\theta+36\sin\theta=324\cos\theta-432\sin\theta$. The expression $324\cos\theta-432\sin\theta$ can be written as $R\cos(\theta+\phi)$ where $R=\sqrt{324^2+(-432)^2}$. Calculating R: $324^2=104976$, $432^2=186624$, $R=\sqrt{104976+186624}=\sqrt{291600}=540$. Thus, the largest possible real part is $\boxed{540}$. **(3743 tokens)**

Figure 9: For a chanlleging problems from AIME 2024, *AdaptThink*-7B is able to employs *Thinking* to solve it, instead of directly generates the final solution.