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 throughout 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. It returns 0/1 if y is incorrect/correct. 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

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

able to select the appropriate mode. Accordingly, our final loss function of *AdaptThink* becomes:

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

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:

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

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:

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

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 learn-

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.

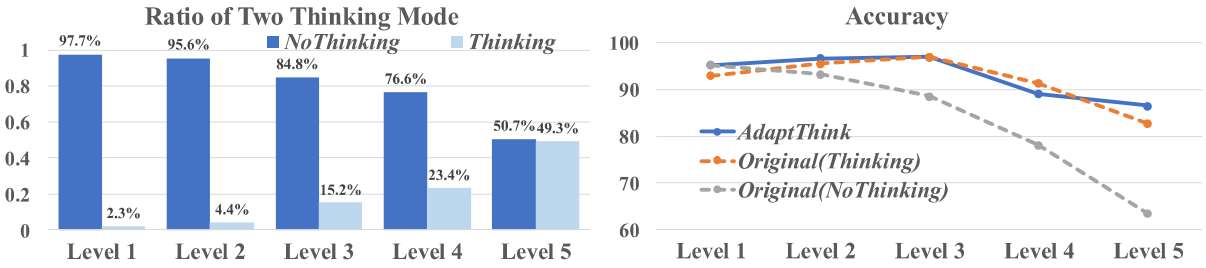

Figure 3: Left: The ratio that *AdaptThink*-7B choose *Thinking* or *NoThinking* across different MATH levels. Right: Comparison of accuracy between *AdaptThink*-7B and DeepSeek-R1-Distill-Qwen-7B using *Thinking* and *NoThinking* across different MATH levels.

ing 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 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