ADaPT: Token-Level Decoupling for Efficient Large Reasoning Models

Anonymous ACL submission

Abstract

Large reasoning models rely on long chain-of-thought to achieve strong performance, but applying such reasoning uniformly incurs high computational cost. Existing efficiency-oriented methods attempt to shorten or mix reasoning strategies, yet often degrade reasoning capability. We identify the root cause as sequence-level coupling between efficiency incentives and correctness optimization, which implicitly penalizes long but correct reasoning trajectories. To address this issue, we propose Adaptive Dual-Process Thinking (ADaPT), a token-level dual-process framework that explicitly decouples efficiency and correctness signals during training. ADaPT introduces a mode-selection token to control fast and slow reasoning, applying efficiency-related rewards exclusively to this token to avoid penalizing correct long reasoning while encouraging efficiency when appropriate. Moreover, ADaPT enables precise and continuous control over the efficiency–performance trade-off at inference time: by adjusting the generation probability of the mode-selection token, a single trained model can smoothly move along the efficiency–performance Pareto frontier. Extensive experiments demonstrate that ADaPT significantly reduces inference cost while maintaining strong reasoning performance across multiple benchmarks. Our code is available at `https://anonymous.4open.science/r/ADaPT-E171`.

1 Introduction

As the focus of language model scaling shifts from training time to test time (Chen et al., 2025b; Zhang et al., 2025b), large reasoning models (LRMs) have emerged as a distinct class designed for complex reasoning tasks. Representative examples include OpenAI-o1 (El-Kishky, 2024) and DeepSeek-R1 (Guo et al., 2025). These models achieve strong performance primarily by relying on long and structured chain-of-thought reasoning at inference time.

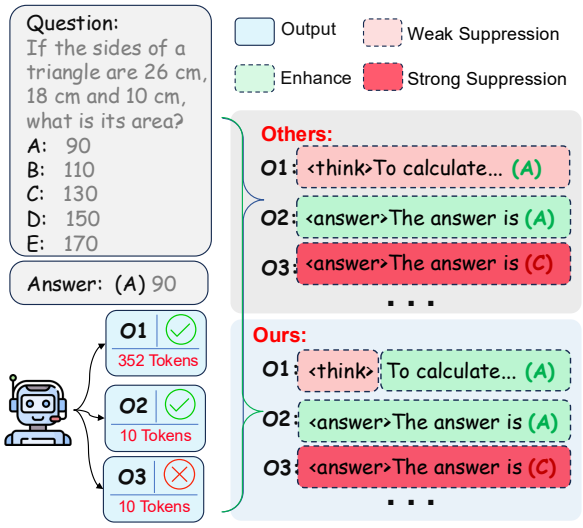

Figure 1: In commonly used methods, efficiency penalties are applied at the sequence level, which penalizes correct but long reasoning and degrades reasoning performance. In contrast, ADaPT applies efficiency penalties only to the `<think>` token that controls reasoning-mode selection, so correct long reasoning is not penalized simply for being long.

Despite these successes, current large reasoning models suffer from a fundamental inefficiency: they apply long chain-of-thought (CoT) reasoning uniformly across inputs, incurring substantial computational overhead (Liu et al., 2025; Hashemi et al., 2025; Han et al., 2025). For simple queries, such excessive reasoning not only increases token usage and inference cost, but can also harm accuracy due to overthinking (Chen et al., 2024; Luo et al., 2025). As a result, improving reasoning efficiency without degrading reasoning performance remains a central challenge for LRMs.

To address this efficiency challenge, existing approaches attempt to reduce reasoning cost through two main directions: length compression methods and hybrid reasoning methods (Li et al., 2025; Wu et al., 2025b; He et al., 2025; Team et al., 2025).

However, these approaches often **improve effi-**

ciency at the expense of reasoning performance, due to unresolved **conflicts between efficiency and correctness signals** during training. Length compression methods reward shorter outputs to improve efficiency, which creates a direct conflict with tasks that require sufficient reasoning depth. Once reasoning is shortened below a task-dependent threshold, performance inevitably degrades (Han et al., 2025; Lee et al., 2025). Hybrid reasoning methods aim to adapt reasoning length to task difficulty (He et al., 2025; Yang et al., 2025), but still rely on sequence-level efficiency rewards. When multiple outputs achieve comparable correctness, shorter answers receive both correctness and efficiency rewards, whereas longer but correct reasoning receives only the correctness reward and is therefore placed at a relative disadvantage (Figure 1). As a result, sequence-level efficiency training systematically suppresses correct long reasoning because it is less efficient, which ultimately leads to degraded deep reasoning capability.

To mitigate this signal conflict, we argue that **efficiency and correctness should be explicitly decoupled during training.** Based on this insight, we propose Adaptive Dual-Process Thinking (ADaPT), a token-level framework inspired by dual-process theory(Evans, 2003) that explicitly models fast and slow reasoning and decouples efficiency signals from answer correctness during training. As shown in Figure 1, ADaPT uses a mode-selection token to control reasoning modes, applying efficiency rewards only to this token while preserving correctness optimization.

By decoupling mode selection from answer correctness, ADaPT avoids penalizing correct long CoT reasoning simply for being long, thereby preserving deep reasoning capability while still encouraging efficient behavior when appropriate. Beyond mitigating performance degradation, ADaPT further enables precise control over the efficiency–performance trade-off at inference time. After training, the probability of triggering the `<think>` token can be smoothly adjusted, allowing a single trained model to move continuously along the efficiency–performance Pareto frontier.

ADaPT adopts a two-stage training pipeline. A supervised fine-tuning (SFT) stage equips the model with basic fast and slow reasoning behaviors, followed by a reinforcement learning stage based on a token-level variant of GRPO to optimize reasoning-mode selection.

The main contributions of this paper are as follows:

- We identify that performance degradation in efficient reasoning methods primarily stems from sequence-level coupling between efficiency and correctness signals.
- We propose ADaPT, a token-level dual-process framework that mitigates efficiency-induced performance degradation by explicitly decoupling these signals.
- ADaPT enables precise and continuous inference time control over the efficiency–performance trade-off, allowing a single model to trace out a Pareto frontier.
- Extensive experiments demonstrate that ADaPT significantly reduces inference cost while maintaining strong reasoning performance across multiple benchmarks.

2 Preliminary

Group Relative Policy Optimization (GRPO). GRPO (Shao et al., 2024) has demonstrated strong performance across various tasks and can achieve efficient scalability within the RLVR paradigm. It estimates advantage values by utilizing the reward scores of N solutions sampled within the same query, eliminating the need for an additional value model.

Formally, let $\pi_{\theta_{\text{old}}}$ and π_θ represent the policy models before and after the update, respectively. For a given problem x, a set of solution sequences $\{y^{(i)}\}_{i=1}^N$ sampled from $\pi_{\theta_{\text{old}}}$, and a reward function $R(\cdot)$, GRPO computes the advantage by normalizing rewards within the group:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\{y^{(i)}\}_{i=1}^N\sim\pi_{\theta_{\text{old}}}(\cdot|x)} \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|y^{(i)}|}\sum_{t=0}^{|y^{(i)}|-1} \min\left[\rho_t^{(i)}(\theta)\hat{A}_t^{(i)},\quad \tilde{\rho}_t^{(i)}(\theta)\hat{A}_t^{(i)}\right], \tag{1}$$

where $\rho_t^{(i)}(\theta)$ is the importance ratio, $\tilde{\rho}_t^{(i)}(\theta)$ is the clipped ratio, and $\hat{A}_t^{(i)}$ is the normalized advantage, defined as follows:

$$\rho_t^{(i)}(\theta) = \frac{\pi_\theta(y_t^{(i)} \mid x, y_{<t}^{(i)})}{\pi_{\theta_{\text{old}}}(y_t^{(i)} \mid x, y_{<t}^{(i)})}, \tag{2}$$

$$\tilde{\rho}_t^{(i)}(\theta) = \text{clip}\left(\rho_t^{(i)}(\theta), 1 \pm \epsilon_{\text{clip}}\right), \tag{3}$$

$$\hat{A}_t^{(i)} = \frac{R(x, y^{(i)}) - \mu_R}{\sigma_R}, \tag{4}$$

where μ_R and σ_R denote the mean and standard deviation of rewards $\{R(x, y^{(j)})\}_{j=1}^N$ within the group.

3 Method

We introduce *Adaptive Dual-Process Thinking* (ADaPT), a framework that enables large language models to adaptively choose between fast and slow reasoning while preserving deep reasoning capability. The core idea of ADaPT is to explicitly model reasoning-mode selection via a dedicated mode token and to decouple efficiency optimization from answer correctness during training. ADaPT is trained in two stages: (1) SFT stage, which exposes the model to both reasoning modes; (2) RL stage, which optimizes reasoning-mode selection using token-level rewards.

3.1 Efficiency-Performance Conflict Analysis

Before introducing ADaPT, we analyze why existing efficiency-oriented reasoning methods often suffer from systematic performance degradation. Despite differences in design, these methods share a common root cause: **efficiency and correctness are coupled at the sequence level**, leading to unavoidable training conflicts.

Why Existing Efficiency-Oriented Methods Fail. For length-compression methods, this conflict is explicit. Length-related rewards scalarize efficiency and correctness into a single sequence-level objective, directly incentivizing shorter reasoning. When tasks require sufficient reasoning depth, this creates a seesaw effect between efficiency and accuracy, suppressing necessary intermediate steps and degrading performance on complex problems (Han et al., 2025; Lee et al., 2025; Han et al., 2024). Hybrid reasoning methods instead rely on routing mechanisms to select among multiple reasoning modes. When trained with correctness-only sequence-level rewards, routing decisions cannot be reliably learned, causing models to default to uniformly slow reasoning to maximize accuracy (Wu et al., 2025b). To encourage mode diversity, prior work introduces additional efficiency-related signals such as length penalties (He et al., 2025; Wu et al., 2025b). However, these signals remain sequence-level, coupling routing decisions with answer generation and implicitly penalizing longer but correct reasoning, often leading to unstable or collapsed routing behaviors.

A Structural View from RL. From a reinforcement learning perspective, the fundamental issue with sequence-level efficiency-performance rewards is a **mismatch between what the reward measures and where the relevant decision is made**. A key observation is that efficiency depends only on the reasoning mode, not on the specific reasoning content generated within that mode. Once a mode is selected by the first action, subsequent actions can affect correctness but no longer influence efficiency.

In a standard MDP formulation(Feng et al., 2012), the return is defined as the cumulative sum of future rewards along the trajectory. The first action a_1, which corresponds to selecting the reasoning mode at the beginning of generation, is the only point at which the agent must trade off efficiency and expected task success, and thus its return should reflect both objectives. In contrast, once the agent transitions to the subsequent state s_2, where the reasoning mode has already been fixed, the remaining return depends solely on whether the final answer is correct. Subsequent actions should therefore optimize correctness only, since they cannot affect the efficiency outcome determined at the entry point. **This structural asymmetry implies that the sequence-level return should be decomposed accordingly.**

By contrast, sequence-level rewards that couple efficiency and correctness distribute efficiency pressure uniformly across all tokens, even though later actions have no control over the mode decision. This misalignment assigns efficiency-related signals to subsequent actions that cannot influence the reasoning mode, inducing a biased optimization objective that ultimately degrades performance under each fixed mode. **Decoupling efficiency into an early mode-selection reward** aligns the return with the decision structure and avoids this signal conflict.

3.2 Stage 1: ADaPT-SFT

SFT serves as a cold start to familiarize the model with predefined reasoning behaviors and their corresponding output formats. Specifically, we define two reasoning modes: (1) `<think>` mode, which generates a full chain-of-thought followed by the final answer; (2) `<answer>` mode, which produces a short rationale or directly outputs the final answer.

During SFT, training examples are annotated with either `<think>` or `<answer>` formats based on task difficulty. This allows the model to learn the basic behaviors associated with slow, explicit reasoning and fast, concise answering, providing a foundation for subsequent adaptive mode selection.

3.3 Stage 2: ADaPT-GRPO

After SFT, the model can generate multiple reasoning formats but cannot reliably select them based on task difficulty. ADaPT-GRPO addresses this by: (1) Decoupling mode selection from answer correctness via token-level rewards. (2) Stabilizing mode learning with a balanced dual-start rollout. This enables adaptive reasoning-mode selection without degrading long CoT reasoning capability.

3.3.1 Token-Level Mode Reward.

Standard GRPO optimizes correctness with sequence-level rewards, but applying mode preferences at the same level penalizes long yet correct reasoning. We therefore introduce a dedicated mode reward that decouples reasoning-mode selection from task correctness.

Mode Reward Design. In ADaPT-GRPO, we introduce a token-level mode reward applied to the `<think>` token:

$$r_{\text{think}} = -\big[\alpha(\text{Acc}_\text{a} - \gamma) + (1-\alpha)(\text{Acc}_\text{a} - \text{Acc}_\text{t})\big], \quad (5)$$

where Acc_a and Acc_t denote the accuracies of the `<answer>` and `<think>` modes, respectively, estimated from rollout data.

This reward consists of two complementary components. The first term enforces an absolute quality threshold on fast reasoning: fast mode is considered reliable only when $\text{Acc}_\text{a} > \gamma$, otherwise invoking `<think>` is encouraged. The second term captures the relative advantage of slow reasoning, increasing the incentive to trigger `<think>` when it provides clear performance gains over fast reasoning. The hyperparameter α balances these absolute and relative criteria.

During training, the mode reward explicitly ties the probability of emitting `<think>` to the reliability of fast reasoning, as measured by Acc_a. As a result, the learned policy internalizes a calibrated preference over invoking slow reasoning. At inference time, this enables smooth control of reasoning depth by adjusting the generation threshold of the `<think>` token, without modifying model parameters.

Combine Rewards and Return Definition. In ADaPT-GRPO, we combine a token-level mode reward with the conventional sequence-level task reward. Specifically, for each prompt with N sampled outputs $\{y^{(i)}\}_{i=1}^{N}$, the token-level reward is defined as:

$$r_t = \begin{cases} r_{\text{think}}, & t = 0,\ y_0 = \langle\text{think}\rangle, \\ r_{\text{seq}}, & t = T-1, \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

The return G_t at each token is computed by summing future rewards:

$$G_t = \sum_{k=t}^{T-1} r_k, \quad (7)$$

which yields

$$G_0 = r_{\text{think}} + R(x,y), \quad G_t = R(x,y), \quad t > 0. \quad (8)$$

Group-wise Global Normalization. Due to the finer granularity of token-level rewards, the original normalization method in Eq. 3 cannot be directly applied. We directly apply group-wise global normalization over all token-level returns within each prompt's rollout set, as in REINFORCE++ (Hu et al., 2025).

Specifically, for a prompt with rollout set $\mathcal{D}_{\text{group}} = \{G_t^{(i)} \mid i = 1,\ldots,N;\ t = 0,\ldots,T-1\}$, the normalized token-level advantage is computed as:

$$\tilde{A}_t^{(i)} = \frac{G_t^{(i)} - \mathbb{E}[G]}{\sqrt{\text{Var}[G]} + \epsilon_{\text{norm}}}, \quad G \in \mathcal{D}_{\text{group}}, \quad (9)$$

where $\mathbb{E}[\cdot]$ and $\text{Var}[\cdot]$ denote the return mean and variance over all tokens and rollouts in $\mathcal{D}_{\text{group}}$, and ϵ is a small constant for numerical stability.

3.3.2 Balanced Dual-Start Rollout.

Since ADaPT concentrates reasoning-mode selection into the first token, reliable learning at this position is critical. To both prevent mode collapse and ensure stable optimization of the mode-selection policy, we adopt a balanced dual-start rollout strategy. Each rollout batch of size n is evenly split into two groups: one forced to start with `<think>` and the other with `<answer>`, guaranteeing sufficient samples for both reasoning modes. The complete algorithmic description is deferred to Appendix B.

Since the first token is explicitly controlled and follows a uniform behavior policy, we set the old-policy probability at the first step to

$$\pi_{\theta_{\text{old}}}(a_t \mid s_t) = \frac{1}{2}, \quad t = 0, \quad (10)$$

when computing importance weights, thereby avoiding distribution mismatch.

However, applying standard PPO/GRPO clipping under this setting severely degrades learning at

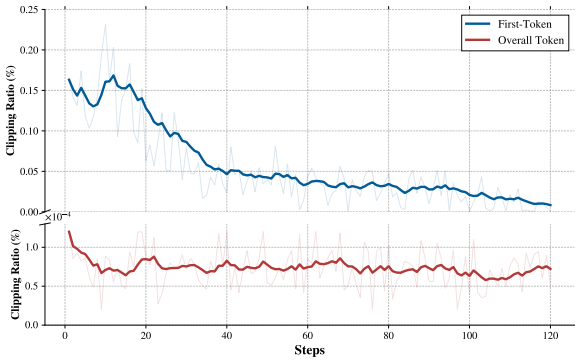

Figure 2: Clipping ratio during GRPO training. Clipping is rare for most tokens (bottom, $\times 10^{-4}$), but becomes much more frequent at the first token (top), which serves as the mode-selection decision point.

the first token, which serves as the mode-selection decision point. Due to the fixed behavior policy at $t = 0$, the importance ratio is frequently clipped, leading to a much higher clipping rate at the mode-selection token than at other positions (Figure 2).

This excessive clipping effectively nullifies the policy gradient at the first token, resulting in vanishing or zero gradients for mode selection. Consequently, the model receives little to no learning signal to optimize when to trigger <think> versus <answer>, severely impairing its ability to learn reliable reasoning-mode selection.

To address this issue, we apply a CISPO (Chen et al., 2025a) ratio constraint exclusively to the first token to ensure stable gradients, while all subsequent tokens are updated using the original GRPO/PPO objective. The resulting token-level policy gradient objective is

$$\mathcal{L}_{\text{PG}}^{\text{ADaPT}}(\theta) = \sum_{i=1}^{N} \sum_{t=0}^{|y^{(i)}|-1} L_t^{(i)\text{ADaPT}}(\theta), \tag{11}$$

$$L_t^{(i)\text{ADaPT}}(\theta) = \begin{cases} -\operatorname{detach}\big(\min(\rho_0^{(i)}, \epsilon_{\text{high}})\big) \\ \quad \times \tilde{A}_0^{(i)} \log \pi_\theta(y_0^{(i)}), & t = 0 \\ -\min\big(\rho_t^{(i)} \tilde{A}_t^{(i)}, \tilde{\rho}_t^{(i)} \tilde{A}_t^{(i)}\big), & t \geq 1. \end{cases} \tag{12}$$

where $\rho_t^{(i)}$ and $\tilde{\rho}_t^{(i)}$ follow the definitions in Eq. 2 and Eq. 3. Finally, model parameters are optimized by maximizing

$$\mathcal{J}_{\text{ADaPT-GRPO}}(\theta) = \mathbb{E}\big[\mathcal{L}_{\text{PG}}^{\text{ADaPT}}(\theta) - \beta D_{\text{KL}}[\pi_\theta \,\|\, \pi_{\text{ref}}]\big]. \tag{13}$$

4 Experiment

4.1 Setup

Datasets and Models. We conduct our experiments using the Qwen2.5 models (Yang et al., 2024), specifically the Qwen2.5-7B-Base and Qwen2.5-3B-Base variants. The training data for the first stage is constructed entirely from publicly available datasets. Specifically, we select a subset of the arm-team dataset[1] and utilize its long CoT and short CoT data to supervise the <think> and <answer> reasoning modes, respectively. For the <answer> mode, we additionally include direct-answer samples when available. In the second stage, we curate a total of 8.5k verifiable question–answer pairs from CSQA (Talmor et al., 2019), GSM8K (Cobbe et al., 2021), and MATH (Hendrycks et al., 2021). These tasks cover a wide range of difficulty levels, from relatively simple commonsense reasoning to complex mathematical problem solving. Further experimental settings and dataset details are provided in Appendix A.

Evaluations. We selected a diverse collection of datasets across commonsense reasoning, mathematics, and open-ended question answering. This selection includes both in-domain and out-of-domain test data, which are subsequently categorized into two difficulty groups. Easy Problems: CSQA, GSM8K, ARC (Clark et al., 2018). Hard Problems: MATH500, MMLU-Pro (Wang et al., 2024), Olympiad (He et al., 2024), and AIME24[2]. For AIME24, since the test set is relatively small, we report the avg@32 metric at a temperature of 0.6. For all other benchmarks, we report the pass@1 metric at a temperature of 0.

Baselines. We compare ADaPT with several representative baselines, including: **Base**, **SFT**, **SFT + GRPO**, and several recent efficiency-oriented RL methods, including **TLMRE** (Arora and Zanette, 2025), **ARM** (Wu et al., 2025b), and **R-4B** (Yang et al., 2025). Detailed descriptions of all baselines are provided in Appendix A.2.

4.2 Main Results

Table 1 summarizes the results of ADaPT on two Qwen2.5 model scales compared with several baselines, across Easy and Hard reasoning benchmarks. Easy tasks mainly involve basic reasoning and knowledge recall, while Hard tasks require multi-step reasoning and complex intermediate derivations.

[1] https://huggingface.co/datasets/arm-team/Stage1_SFT_aqua_rat

[2] https://huggingface.co/datasets/Maxwell-Jia/AIME_2024

Methods	Easy						Hard								AVG	
	CSQA		GSM8K		ARC		MATH500		MMLU-Pro		Olympiad		AIME24			
	ACC ↑	Length ↓	ACC ↑	Length ↓	ACC ↑	Length ↓	ACC ↑	Length ↓	ACC ↑	Length ↓	ACC ↑	Length ↓	ACC ↑	Length ↓	ACC ↑	Length ↓
Qwen2.5-7B																
Base	76.3	222	81.8	332	70.4	449	49.4	648	42.9	613	28.9	892	4.6	923	50.6	582
SFT	79.4	376	68.3	541	73.2	949	44.0	632	40.1	893	24.8	1069	1.3	1484	47.3	849
SFT+GRPO	84.3	391	91.1	801	87.8	405	77.4	1466	48.9	1007	39.5	2850	14.4	3857	63.3	1540
TLMRE	82.8	204	88.3	584	82.6	287	75.4	1073	47.4	672	36.5	2782	14.2	3491	61.0	1299
ARM	86.2	137	89.2	297	86.7	204	73.8	893	47.0	647	36.2	2362	13.1	3374	61.7	1131
R-4B	85.7	145	88.6	310	84.7	274	74.4	912	46.9	771	34.9	2017	11.7	2883	60.9	1044
ADaPT$_{answer}$	86.1	24	84.7	152	85.3	42	67.2	427	44.4	317	31.7	795	6.3	1201	57.9	423
ADaPT$_{think}$	83.5	385	91.4	803	87.7	327	78.2	1541	49.8	1083	40.2	3071	15.1	3772	63.7	1569
ADaPT	86.7	65	90.8	348	86.9	78	76.8	1003	48.5	680	35.8	1996	13.7	3044	62.7	1031
Qwen2.5-3B																
Base	66.7	176	64.1	258	28.6	474	40.2	593	18.2	673	18.4	794	1.7	1103	33.9	582
SFT	71.4	164	48.3	194	35.1	312	30.6	407	12.3	552	9.8	692	0.0	982	29.6	472
SFT+GRPO	79.3	418	88.9	873	72.1	577	66.4	1483	29.4	1114	31.4	2873	4.3	3433	53.1	1539
TLMRE	76.3	307	85.8	674	70.5	412	64.6	1178	28.4	1004	28.4	2085	3.5	3293	51.1	1279
ARM	78.4	141	84.2	286	70.3	203	63.0	973	28.6	673	28.9	2341	3.9	3094	51.0	1101
R-4B	76.9	226	83.6	427	68.3	184	62.8	1087	26.3	518	25.6	1864	2.5	2794	49.4	1014
ADaPT$_{answer}$	77.1	22	80.1	177	66.4	36	58.4	503	25.2	272	22.8	1033	2.0	1349	47.4	485
ADaPT$_{think}$	77.9	364	88.6	816	71.0	294	66.8	1492	27.5	1053	31.7	3289	4.6	3404	52.6	1530
ADaPT	79.5	72	86.3	317	71.5	74	64.6	1074	28.2	583	29.5	2103	3.8	2872	51.9	1013

Table 1: Performance comparison on Easy and Hard reasoning benchmarks. Results report accuracy (ACC) and average generation length (Length) for different methods on two Qwen2.5 model scales. $ADaPT_{\text{think}}$ and $ADaPT_{\text{answer}}$ are constrained variants of ADaPT that force the model to always use slow reasoning or fast reasoning, respectively.

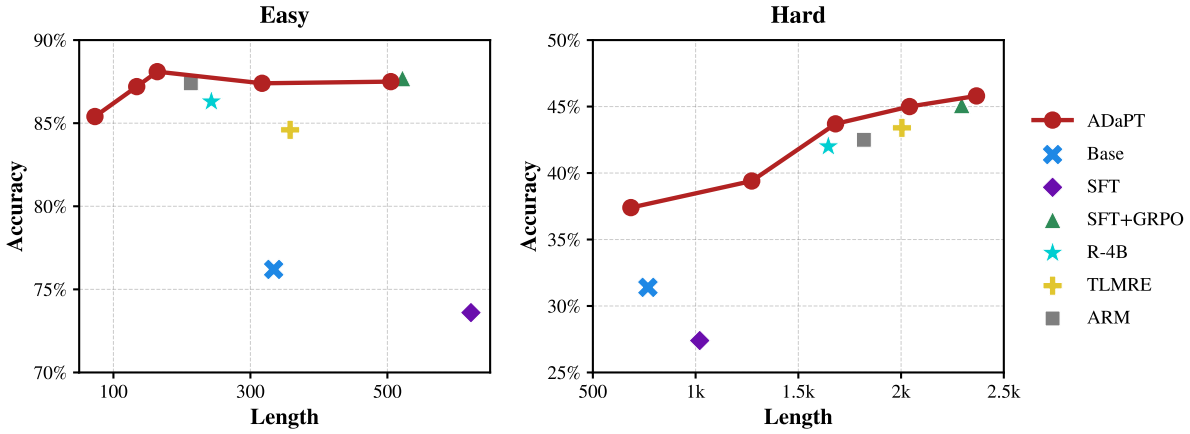

Figure 3: Accuracy and length trade-off of ADaPT on Easy (left) and Hard (right) tasks under different <think> token thresholds. By adjusting the threshold, ADaPT enables continuous Pareto control of reasoning efficiency, achieving Pareto optimal accuracy at different token budgets, while other methods remain inside this frontier.

SFT+GRPO substantially improves reasoning accuracy but incurs high reasoning cost. Compared with Base and SFT, SFT+GRPO achieves notable accuracy gains, especially on Hard tasks, confirming the effectiveness of reinforcement learning for complex reasoning. However, these gains are accompanied by a significant increase in generation length. For example, on Qwen2.5-7B, SFT+GRPO produces over 2k tokens on Olympiad and more than 3k tokens on AIME24, with similar trends observed on Qwen2.5-3B. This suggests that when training is dominated by correctness rewards, the model tends to adopt uniformly long reasoning trajectories to maximize accuracy.

ADaPT achieves a more favorable balance between performance and efficiency. Compared with SFT+GRPO, ADaPT significantly reduces generation length while incurring only marginal accuracy loss. On Qwen2.5-7B, ADaPT reduces the average generation length from 1540 to 1031 tokens, with accuracy decreasing slightly from 63.1 to 62.7. A similar pattern holds for Qwen2.5-3B. In contrast, existing efficiency-oriented baselines such as TLMRE, ARM, and R-4B often reduce reasoning length at the cost of noticeable performance degradation, particularly on Hard tasks. These results indicate that decoupling mode-selection rewards from correctness rewards enables the model

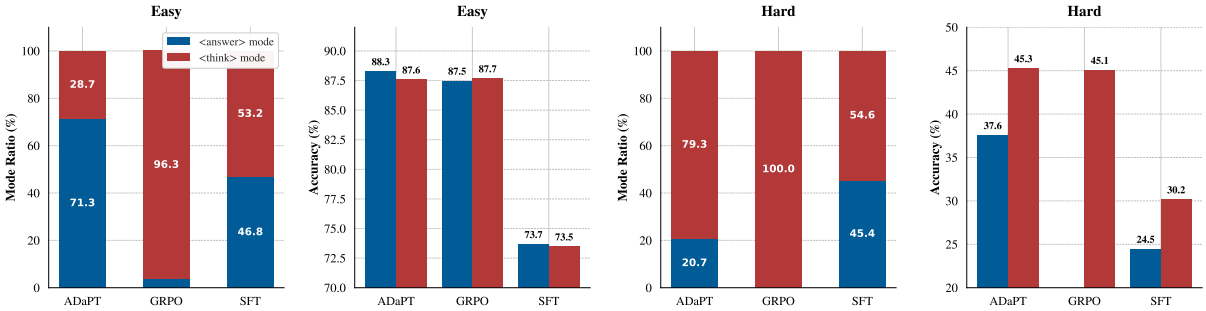

Figure 4: Distribution and accuracy of <answer> and <think> modes on Easy and Hard tasks. ADaPT adaptively selects fast reasoning for Easy tasks and increases slow reasoning only when required on Hard tasks, achieving a superior balance between efficiency and reasoning performance.

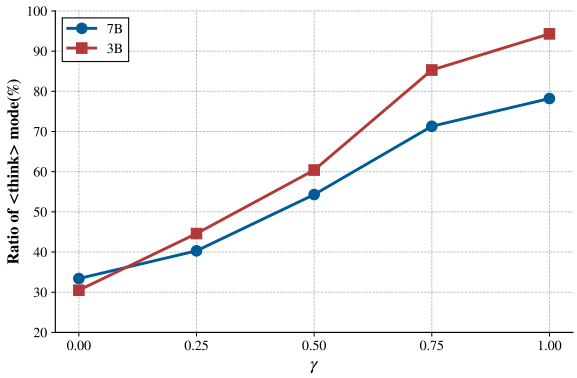

Figure 5: Effect of γ on the proportion of slow reasoning (<think>) usage for 7B and 3B models. Larger γ monotonically increases slow reasoning usage, with the 3B model showing a stronger shift due to lower fast reasoning reliability.

to eliminate unnecessary reasoning without sacrificing accuracy.

Different tasks exhibit distinct demands for fast and slow reasoning. On challenging benchmarks such as Olympiad and AIME24, forcing slow reasoning ($ADaPT_{\text{think}}$) substantially increases token usage but also yields clear accuracy gains. Conversely, on simpler tasks like CSQA, slow reasoning leads to longer outputs with lower accuracy, while fast reasoning ($ADaPT_{\text{answer}}$) achieves competitive performance with minimal cost. This highlights that indiscriminately increasing reasoning length can be harmful on simple problems.

ADaPT preserves strong long-chain reasoning capability. Across both model scales, $ADaPT_{\text{think}}$ matches or slightly outperforms SFT+GRPO on most benchmarks. For instance, on Qwen2.5-7B, $ADaPT_{\text{think}}$ achieves an average accuracy of 63.7, exceeding the 63.1 of SFT+GRPO. Although slow reasoning remains costly, these results demonstrate that token-level mode selection rewards do not weaken deep reasoning ability, and the model retains the capacity to perform complex multi-step reasoning when required.

4.3 Continuous Pareto Control of Reasoning Efficiency

A key property of ADaPT is that **reasoning depth can be continuously controlled by adjusting the generation probability threshold of the <think> token.** When the probability of generating <think> exceeds a given threshold, the model enters the slow-reasoning mode. We evaluate different threshold settings on Easy and Hard tasks and visualize the resulting accuracy–length trade-off in Figure 3.Across both task categories, increasing the threshold leads to more frequent slow reasoning, resulting in a monotonic increase in generation length and a corresponding improvement in accuracy. On Easy tasks, accuracy increases moderately before saturating, while on Hard tasks, higher reasoning depth yields consistent performance gains. These results show that ADaPT learns a continuous and controllable efficiency–capability trade-off rather than relying on a discrete switch between reasoning modes. Moreover, when projecting other baselines into the same accuracy–length space, most lie strictly inside the Pareto frontier defined by ADaPT. This indicates that ADaPT achieves higher accuracy at comparable or lower token budgets, confirming its systematic advantage in balancing reasoning efficiency and performance.

4.4 Adaptive Use of Fast and Slow Reasoning

Figure 4 demonstrates that **ADaPT learns to select reasoning modes in a manner that is both accurate and sensitive to task difficulty.** On Easy tasks, GRPO relies almost entirely on slow reason-

ing, reflecting training that is dominated by correctness rewards and therefore overuses long reasoning even when it is unnecessary. SFT exposes the model to both reasoning formats, but lacks a mechanism to select between them, resulting in mixed usage with lower accuracy. In contrast, ADaPT predominantly adopts fast reasoning on Easy tasks, achieving comparable accuracy with substantially lower inference cost. On Hard tasks, GRPO again defaults to slow reasoning for all inputs, whereas ADaPT selectively increases the use of slow reasoning while preserving a meaningful proportion of fast reasoning. Overall, ADaPT achieves strong accuracy under both reasoning modes and demonstrates effective and high performance reasoning mode selection across different task difficulties.

4.5 Ablation Study on γ

We ablate the parameter γ, which controls the tolerance of the mode reward to fast reasoning quality and regulates how easily the `<think>` mode is triggered. As γ increases, fast reasoning must satisfy a stricter reliability requirement, causing the model to invoke slow reasoning more frequently. Figure 5 shows a monotonic increase in `<think>` usage for both 7B and 3B models as γ varies from 0 to 1, demonstrating that γ enables smooth control over reasoning depth. The 3B model exhibits a stronger response and surpasses the 7B model for $\gamma \geq 0.25$, consistent with its lower fast-reasoning reliability and greater reliance on slow reasoning as a fallback.

5 Related Work

5.1 Length Compression

Recently, many studies have focused on improving the reasoning efficiency of LLMs. Some prompt-based approaches aim to simplify reasoning by modifying input prompts (Wu et al., 2025a; Muennighoff et al., 2025; Xu et al., 2025; Chen et al., 2024), for example by imposing explicit token budget constraints or instructing models to generate more concise reasoning chains (Renze and Guven, 2024; Han et al., 2024). Other methods emphasize early stopping strategies to reduce computational cost during reasoning (Han et al., 2025; Manvi et al., 2024; Li et al., 2024). In addition, several techniques reduce reasoning cost by explicitly controlling or pruning output length, typically through injecting multiple reasoning formats during pre-training or applying length penalties during reinforcement learning (Han et al., 2025; Team et al., 2025; Shen et al., 2025; Chang et al., 2025). Most of these approaches attempt to balance token budget and reasoning performance via explicit length constraints, often relying on accurate per-task token budget estimation or specialized training.

5.2 Hybrid Reasoning

While many works improve efficiency by uniformly compressing reasoning length, hybrid reasoning offers a more flexible alternative by allowing models to adopt different reasoning strategies depending on the task (Fang et al., 2025). This paradigm requires models to switch between reasoning modes, which can be guided by external signals (Ong et al., 2024; Du et al., 2025) or learned autonomously by the model itself (Fang et al., 2025; Wu et al., 2025b; He et al., 2025; Yang et al., 2025; Zhang et al., 2025a). Externally controlled approaches typically rely on routing mechanisms, such as complex control systems (Du et al., 2025) or cascaded frameworks where a smaller model estimates task difficulty before invoking a larger one (Ong et al., 2024). Other methods regulate reasoning behavior through explicit reasoning switches or prompts (Zhao et al., 2024). To enable autonomous reasoning-mode selection, many approaches rely on reinforcement learning. Some bias learning toward specific reasoning formats via sampling strategies (Wu et al., 2025b; Yang et al., 2025), while others design reward functions to encourage more efficient reasoning (He et al., 2025; Zhang et al., 2025a). However, when learning long and short reasoning simultaneously, these methods suffer from a fundamental limitation: sequence-level length rewards introduce unavoidable training conflicts between the two modes.

6 Conclusion

In this work, we show that the commonly observed trade-off between reasoning efficiency and capability arises from sequence-level coupling of efficiency and correctness signals during training. To address this issue, we propose ADaPT, a token-level framework that decouples reasoning-mode selection from answer correctness via a dedicated mode-selection token. Experiments demonstrate that ADaPT substantially reduces inference cost while preserving strong long-chain reasoning ability, and enables continuous control over the efficiency–accuracy trade-off at inference time.

7 Limitation

This work has several limitations that point to directions for future research. First, our evaluation focuses on a set of standard reasoning benchmarks, mainly covering mathematical and commonsense reasoning. While these benchmarks span different difficulty levels, they may not fully reflect more diverse real-world settings, such as long-context or interactive reasoning. Second, ADaPT adopts a binary distinction between fast and slow reasoning modes for clarity and controllability. Although effective, this design does not capture more fine-grained variations in reasoning behavior, which could be explored in future extensions. Finally, experiments are conducted on models up to the 7B scale. While the approach is generally applicable, its behavior on larger models and under different training regimes remains to be investigated.

References

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

Edward Y. Chang, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. 2025. Demystifying long chain-of-thought reasoning in llms. *ArXiv*, abs/2502.03373.

Aili Chen, Aonian Li, Bangwei Gong, Binyang Jiang, Bo Fei, Bo Yang, Boji Shan, Changqing Yu, Chao Wang, Cheng Zhu, Chengjun Xiao, Chengyu Du, Chi Zhang, Chu Qiao, Chunhao Zhang, Chunhui Du, Congchao Guo, Da Chen, Deming Ding, and 80 others. 2025a. Minimax-m1: Scaling test-time compute efficiently with lightning attention. *ArXiv*, abs/2506.13585.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wangxiang Che. 2025b. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *ArXiv*, abs/2503.09567.

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. *ArXiv*, abs/2412.21187.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Karl Cobbe, Vineet Kosaraju, Mo 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. *ArXiv*, abs/2110.14168.

Y. Du, C. Guo, W. Wang, and G. Tang. 2025. Cognitive decision routing in large language models: When to think fast, when to think slow. *ArXiv*, abs/2508.16636.

Ahmed El-Kishky. 2024. Openai o1 system card. *ArXiv*, abs/2412.16720.

Jonathan St B. T. Evans. 2003. In two minds: Dual-process accounts of reasoning. *Trends in Cognitive Sciences*, 7(10):454–459.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. 2025. Thinkless: Llm learns when to think. *ArXiv*, abs/2505.13379.

Zhengzhu Feng, Richard Dearden, Nicolas Meuleau, and Richard Washington. 2012. Dynamic programming for structured continuous markov decision problems. *arXiv preprint arXiv:1207.4115*.

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

Jinyi Han, Ying Huang, Ying Liao, Zishang Jiang, Xikun Lu, Haiquan Zhao, Xinyi Wang, Guanghao Zhou, Sihang Jiang, Jiaqing Liang, Weikang Zhou, Zeye Sun, Fei Yu, and Yanghua Xiao. 2025. Your models have thought enough: Training large reasoning models to stop overthinking. *ArXiv*, abs/2509.23392.

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

Masoud Hashemi, Oluwanifemi Bamgbose, Sathwik Tejaswi Madhusudhan, Jishnu Sethumadhavan Nair, Aman Tiwari, and Vikas Yadav. 2025. Dnr bench: Benchmarking over-reasoning in reasoning llms.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Annual Meeting of the Association for Computational Linguistics*.

Qianyu He, Siyu Yuan, Xuefeng Li, Mingxuan Wang, and Jiangjie Chen. 2025. Thinkdial: An open recipe for controlling reasoning effort in large language models. *ArXiv*, abs/2508.18773.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Xiaodong Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *ArXiv*, abs/2103.03874.

Jian Hu, Jason Klein Liu, Haotian Xu, and Wei Shen. 2025. Reinforce++: Stabilizing critic-free policy optimization with global advantage normalization.

Ayeong Lee, Ethan Che, and Tianyi Peng. 2025. How well do llms compress their own chain-of-thought? a token complexity approach. *ArXiv*, abs/2503.01141.

Wei Li, Yanbin Wei, Qiushi Huang, Jiangyue Yan, Yang Chen, James T. Kwok, and Yu Zhang. 2025. Dynamicmind: A tri-mode thinking system for large language models. *ArXiv*, abs/2506.05936.

Yiwei Li, Peiwen Yuan, Shaoxiong Feng, Boyuan Pan, Xinglin Wang, Bin Sun, Heda Wang, and Kan Li. 2024. Escape sky-high cost: Early-stopping self-consistency for multi-step reasoning. *ArXiv*, abs/2401.10480.

Yule Liu, Jingyi Zheng, Zhen Sun, Zifan Peng, Wenhan Dong, Zeyang Sha, Shiwen Cui, Weiqiang Wang, and Xinlei He. 2025. Thought manipulation: External thought can be efficient for large reasoning models. *ArXiv*, abs/2504.13626.

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

Rohin Manvi, Anikait Singh, and Stefano Ermon. 2024. Adaptive inference-time compute: Llms can predict if they can do better, even mid-generation. *ArXiv*, abs/2410.02725.

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

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph Gonzalez, Mohammed Waleed Kadous, and Ion Stoica. 2024. Routellm: Learning to route llms with preference data. *ArXiv*, abs/2406.18665.

Matthew Renze and Erhan Guven. 2024. The benefits of a concise chain of thought on problem-solving in large language models. In *2024 2nd International Conference on Foundation and Large Language Models (FLLM)*, pages 476–483.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junmei Song, Mingchuan Zhang, Y. K. Li, Yu Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *ArXiv*, abs/2402.03300.

Yi Shen, Jian Zhang, Jie fu Huang, Shuming Shi, Wenjing Zhang, Jiangze Yan, Ning Wang, Kai Wang, and Shiguo Lian. 2025. Dast: Difficulty-adaptive slow-thinking for large reasoning models. *ArXiv*, abs/2503.04472.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pages 1279–1297.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *ArXiv*, abs/1811.00937.

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, Feng Tang, Flood Sung, Guangda Wei, Guokun Lai, and 75 others. 2025. Kimi k1.5: Scaling reinforcement learning with llms. *ArXiv*, abs/2501.12599.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max W.F. Ku, Kai Wang, Alex Zhuang, Rongqi "Richard" Fan, Xiang Yue, and Wenhu Chen. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *ArXiv*, abs/2406.01574.

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

Siye Wu, Jian Xie, Yikai Zhang, Aili Chen, Kai Zhang, Yu Su, and Yanghua Xiao. 2025b. Arm: Adaptive reasoning model. *ArXiv*, abs/2505.20258.

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

Qi Yang, Bolin Ni, Shiming Xiang, Han Hu, Houwen Peng, and Jie Jiang. 2025. R-4b: Incentivizing general-purpose auto-thinking capability in mllms via bi-mode annealing and reinforce learning. *arXiv preprint arXiv:2508.21113*.

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, and 25 others. 2024. Qwen2.5 technical report. *ArXiv*, abs/2412.15115.

Jiajie Zhang, Nianyi Lin, Lei Hou, Ling Feng, and Juanzi Li. 2025a. Adaptthink: Reasoning models can learn when to think. *ArXiv*, abs/2505.13417.

Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Zhihan Guo, Yufei Wang, Irwin King, Xue Liu, and Chen Ma. 2025b. A survey on test-time scaling in large language models: What, how, where, and how well?

Zirui Zhao, Hanze Dong, Amrita Saha, Caiming Xiong, and Doyen Sahoo. 2024. Automatic curriculum expert iteration for reliable llm reasoning. *ArXiv*, abs/2410.07627.

Appendix

A Experiments Details

A.1 Hardware and Software Platform.

All experiments are conducted on workstations equipped with four NVIDIA A800 PCIe GPUs with 80GB memory each, running Ubuntu 20.04.6 LTS. Our implementation is based on the Verl (Sheng et al., 2025) framework. Rollouts are generated using temperature $= 0.6$. We set the maximum prompt length to 2,048 tokens and the maximum response length to 8192 tokens. Training is conducted for 120 steps with a batch size of 128 and a mini-batch size of 64. The actor is optimized using Adam, with learning rates of 1×10^{-6}. Our implementation uses grouped sampling with group size $G = 16$.

A.2 Details of Training Data

ADaPT-SFT Data Construction. For the SFT stage, we first use the base model to sample each question in the arm-team dataset ten times. If the model answers a question correctly with a probability greater than 0.5, we classify the question as easy; otherwise, it is classified as hard. For easy questions, we include both the standard chain-of-thought (CoT) data and the direct-answer data as training samples. For hard questions, we include only the long CoT data as training samples. This procedure establishes a difficulty-aware training curriculum and allows us to construct SFT data with explicit reasoning-depth supervision. Finally, we select 5k easy questions and 5k hard questions, resulting in a total of 15k training instances, since each easy question corresponds to two reasoning formats.

Baseline Details We compare ADaPT with several representative baselines: (1) **Base**: The foundational model without any fine-tuning. (2) **SFT**: The model trained only through the first-stage SFT. This baseline is used to investigate whether SFT alone can instill the capability of accurate format selection in the model. (3) **SFT + GRPO**: The model is first trained with Stage 1 SFT and then further trained using the standard GRPO algorithm. This explores the final distribution of the two modes under a free-exploration setting. (4) **TLMRE** (Arora and Zanette, 2025): An RL algorithm that incorporates a length penalty to encourage the model to generate shorter responses. (5) **ARM** (Wu et al., 2025b): A GRPO-based method designed to improve reasoning efficiency via four-mode routing. (6) **R-4B** (Yang et al., 2025): An RL algorithm that teaches efficient thinking to models through bi-mode annealing.

Algorithm 1 Balanced Dual-Start Rollout

Require: prompt x, policy π_θ, batch size N, mode tokens $\tau_{\text{t}} =$ <think>, $\tau_{\text{a}} =$ <answer>

Ensure: $\mathcal{D} = \{(x, y^{(i)})\}_{i=1}^{N}$ with $N/2$ samples per mode

$\mathcal{D} \leftarrow \emptyset$
for $i = 1$ **to** N **do**
  $y_0^{(i)} \leftarrow \mathbb{I}[i \le N/2]\tau_{\text{t}} + \mathbb{I}[i > N/2]\tau_{\text{a}}$
  $y_{1:T_i}^{(i)} \sim \prod_{t=1}^{T_i} \pi_\theta(\cdot \mid x, y_{0:t-1}^{(i)})$ ▷ until <eos> / max length
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(x, (y_0^{(i)}, y_{1:T_i}^{(i)}))\}$
end for
return $\mathcal{D}$

ADaPT-GRPO Data Example. For the GRPO data, we select samples from CSQA, GSM8K, and MATH, with a total of 8.5k instances. For each of these three datasets, we provide representative examples, as shown in Figure 6.

B Additional Implementation Details

This appendix provides the detailed rollout procedure for the balanced dual-start strategy introduced in Section 3.3.2, as summarized in Algorithm 1. The goal is to ensure sufficient and balanced sampling of both reasoning modes during training, while stabilizing learning at the mode-selection token.

C System Prompts and Case study

We present our system prompt and example cases for the two reasoning modes below.

GRPO Training data

/*CSQA*/

Question: Where are you likely to find a professional prostitute?

A. new york
B. whorehouse
C. street corner
D. corner of two streets
E. brother

Answer: B

/*GSM8K*/

Question: John buys 10 packs of magic cards. Each pack has 20 cards and 1/4 of those cards are uncommon. How many uncommon cards did he get?

Answer: 4

/*MATH*/

Question: To calculate 31^2, Emily mentally figures the value 30^2 and adds 61. Emily subtracts a number from 30^2 to calculate 29^2. What number does she subtract?

Answer: 59

Figure 6: Examples of GRPO Training Data

System Prompt

/*System Prompt*/

Please reason step by step. For difficult questions, output **<think>** and engage long thinking mode. For simple questions, output **<answer>** and engage short thinking mode. Provide your final answer within `\boxed{}`.

/*User Prompt*/

Let $a > 0$, and let $P(x)$ be a polynomial with integer coefficients such that

$$P(1) = P(3) = P(5) = P(7) = a$$

and

$$P(2) = P(4) = P(6) = P(8) = -a.$$

What is the smallest possible value of a?

Case Study 1: <answer> mode

/*User Prompt*/

A basket contains 25 oranges among which 1 is bad, 20% are unripe, 2 are sour and the rest are good. How many oranges are good?

/*Response*/

<answer>

There are 25 oranges in total. One is bad, 20% of them are unripe, which is 5 oranges, and 2 are sour. Subtracting these from the total gives 25 minus 1 minus 5 minus 2, which equals 17. Therefore, there are 17 good oranges. The answer is $\boxed{17}$.

/*User Prompt*/

What happens to water in a leaky faucet?

A. water spout
B. wet clothes
C. dribble
D. power turbine
E. thin soup

/*Response*/

<answer>

A leaky faucet allows water to drip continuously, which is a form of dribbling. The water does not form a spout, wet clothes, power a turbine, or become thin soup.
The answer is $\boxed{C}$.

/*User Prompt*/

Snow, rain, hail, and fog are all forms of

A. gas
B. water
C. wind
D. clouds

/*Response*/

<answer>

The answer is $\boxed{B}$.

Case Study 2: <think> mode

/*User Prompt*/

Define

$$p = \sum_{k=1}^{\infty} \frac{1}{k^2} \quad \text{and} \quad q = \sum_{k=1}^{\infty} \frac{1}{k^3}.$$

Find a way to write

$$\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^3}$$

in terms of p and q.

/*Response*/

<think>

Okay, so I have this problem here about these infinite series involving p and q. Let me try to understand what I need to do. So, the problem says that p is the sum of the reciprocals of the squares of the natural numbers, and q is the sum of the reciprocals of the cubes. Then I need to find a way to express the double sum $\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^3}$ in terms of p and q.

First, I remember that p is the Riemann zeta function evaluated at 2, so $p = \zeta(2) = \frac{\pi^2}{6}$. And q is $\zeta(3)$, which is a bit more complicated, but I don't think I need the exact value of q here. The main thing is to find a way to relate the double sum to these two sums.

So, the double sum is $\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^3}$. I think I should start by changing the order of summation. When you have a double sum, sometimes changing the order can help. So, instead of summing over j first and then k, I can sum over $j + k$ first. Let me see how that works.

Let me denote $n = j + k$. Then for each n, j can be from 1 to $n - 1$ and $k = n - j$. So the double sum becomes $\sum_{n=2}^{\infty} \sum_{j=1}^{n-1} \frac{1}{n^3}$. Because when $n = 2$, j can be 1, when $n = 3$, j can be 1 or 2, and so on. So the inner sum is $\sum_{j=1}^{n-1} \frac{1}{n^3} = \frac{n-1}{n^3}$. Therefore, the double sum is $\sum_{n=2}^{\infty} \frac{n-1}{n^3}$.

Now, I can simplify $\frac{n-1}{n^3}$ as $\frac{1}{n^2} - \frac{1}{n^3}$. So the sum becomes $\sum_{n=2}^{\infty} \left(\frac{1}{n^2} - \frac{1}{n^3}\right)$.

I can split this into two separate sums: $\sum_{n=2}^{\infty} \frac{1}{n^2} - \sum_{n=2}^{\infty} \frac{1}{n^3}$. Now, I know that $\sum_{n=1}^{\infty} \frac{1}{n^2} = p = \frac{\pi^2}{6}$, so $\sum_{n=2}^{\infty} \frac{1}{n^2} = p - 1$. Similarly, $\sum_{n=1}^{\infty} \frac{1}{n^3} = q$, so $\sum_{n=2}^{\infty} \frac{1}{n^3} = q - 1$. Therefore, the sum becomes $(p - 1) - (q - 1) = p - 1 - q + 1 = p - q$.

So the answer should be $p - q$. Let me double-check to make sure I didn't make any mistakes. Let me go through the steps again.

Original double sum: $\sum_{j=1}^{\infty} \sum_{k=1}^{\infty} \frac{1}{(j+k)^3}$. By changing the order to sum over $n = j + k$, the inner sum is from $j = 1$ to $n - 1$, so $\sum_{j=1}^{n-1} \frac{1}{n^3} = \frac{n-1}{n^3} = \frac{1}{n^2} - \frac{1}{n^3}$. Then summing from $n = 2$ to infinity gives $\sum_{n=2}^{\infty} \frac{1}{n^2} - \sum_{n=2}^{\infty} \frac{1}{n^3} = (p - 1) - (q - 1) = p - q$. Yeah, that seems right.

So the answer is $\boxed{p - q}$.