Optimizing Anytime Reasoning via Budget Relative Policy Optimization

Penghui Qi[12], **Zichen Liu**[12], **Tianyu Pang**[1], **Chao Du**[1], **Wee Sun Lee**[2], **Min Lin**[1]
[1]Sea AI Lab [2]National University of Singapore
https://github.com/sail-sg/AnytimeReasoner

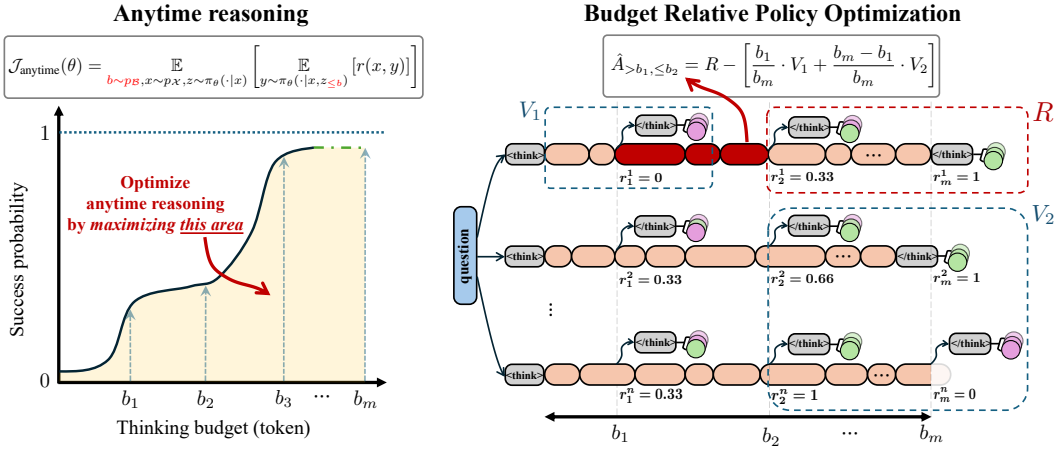


Figure 1: **Left**: We optimize *anytime reasoning* by sampling thinking budgets from a prior distribution $p_{\mathcal{B}}$ and maximizing the rewards at sampled budgets to push up the area under the curve. This objective naturally introduces *verifiable dense rewards* into the thinking process. **Right**: Budget Relative Policy Optimization (BRPO) leverages these dense rewards to improve advantage estimation via the Monte Carlo return (R) and an interpolated baseline that combines current progress (V_1) and the average return within the rollout group (V_2).

Abstract

Scaling test-time compute is crucial for enhancing the reasoning capabilities of large language models (LLMs). Existing approaches typically employ reinforcement learning (RL) to maximize a verifiable reward obtained at the end of reasoning traces. However, such methods optimize only the final performance under a large and fixed token budget, which hinders efficiency and flexibility in both training and deployment. In this work, we present **AnytimeReasoner**, a novel framework for optimizing reasoning performance under varying thinking budget constraints. To achieve this, we truncate the complete thinking process to fit within sampled token budgets from a prior distribution, compelling the model to summarize the optimal answer for each truncated thinking for verification. This introduces **verifiable dense rewards** into the reasoning process, facilitating more effective credit assignment in RL optimization. We then optimize the thinking and summary policies in a decoupled manner to maximize the cumulative reward. Additionally, we introduce a novel variance reduction technique, **B**udget **R**elative **P**olicy **O**ptimization (**BRPO**), to enhance the robustness and efficiency of the learning process when reinforcing the thinking policy. Empirical results in mathematical reasoning tasks demonstrate that our method consistently outperforms GRPO across all thinking budgets under various prior distributions, enhancing both training and token efficiency.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

1 Introduction

OpenAI o1 [OpenAI, 2024] and DeepSeek-R1 [Guo et al., 2025] have shown that scaling test-time compute via RL is crucial for LLM reasoning. This involves an extensive thinking process using the chain of thought (CoT) [Wei et al., 2022] before producing an answer. RL is then employed to maximize the outcome reward provided by a rule-based verifier to check the correctness of the generated answer. While RL for LLM reasoning is an active area of research, most existing work focuses on optimizing final performance based on the complete thinking process. This approach can be inefficient in both training and deployment, as long CoTs are costly, especially for online services.

In our work, we focus on **optimizing anytime reasoning for LLMs via RL**. This is conceptually similar to the *anytime algorithms* introduced in Dean and Boddy [1988], Zilberstein and Russell [1995], where the system can be interrupted at any point during computation, providing the best possible solution so far and is expected to improve the solution quality when more resources are allocated. Concretely in LLM reasoning, we assume the thinking process can be interrupted at any time, and the model should be able to summarize the best solution from incomplete thinking. This capability can significantly extend the serving capacity for online services with limited computing resources. When there are too many requests to handle, the service can choose to interrupt in-progress requests once the thinking length is able to give sufficient accuracy, reserving longer thinking with better accuracy when resources are available. Moreover, users may want to control the thinking budget as in Gemini 2.5[Comanici et al., 2025], but the optimal budget is often agnostic. Compared to budget-aware reasoning[Han et al., 2024], our design supports an economical strategy by incrementally increasing the budget, as it allows for continued thinking and reuses the computation already spent.

To achieve optimal performance for anytime reasoning, we propose **sampling the thinking budget from a prior distribution** while learning, rather than using a fixed, large budget as in prior work [Liu et al., 2025, Zeng et al., 2025, Luo et al., 2025]. This approach makes the model performance robust to potential interruptions in the thinking process, while incentivizing it to reach correct answers more efficiently. By achieving a balance between token efficiency and thorough exploration [Qu et al., 2025], these models are also able to obtain better performance when given larger budgets.

We investigate how to efficiently train LLMs with RL under sampled thinking budgets. By forcing the model to summarize the answers at predefined thinking budgets (drawn from the support of the prior distribution), we introduce **verifiable dense rewards** into the reasoning process. These rewards provide richer signals and better credit assignment during training [Qu et al., 2025, Cui et al., 2025a]. We also propose **a novel variance reduction technique termed Budget Relative Policy Optimization (BRPO) that advances beyond GRPO** [Shao et al., 2024] to improve training stability and efficiency under this dense reward framework. As illustrate in Figure 1 (right), we leverage rewards at previous budgets to compute the advantage function, combining with the average return of a group of reasoning trajectories. Empirically, we observe that generating a high-quality summary is critical for both final and anytime performance. Thus, we **decouple the optimization of the thinking and summary policies**, always sampling from a uniform distribution to derive a better summary policy, thereby improving training efficiency.

We term our overall framework as *AnytimeReasoner*. Experimental results demonstrate that **AnytimeReasoner consistently surpasses GRPO in both final and anytime performance**. We conduct extensive ablation studies to evaluate the impact of each component. By independently incorporating decoupled optimization, variance reduction, and budget sampling into GRPO, we observe significant performance enhancements, underscoring the effectiveness of our methods. Notably, even when merely using the maximum token budget (without budget sampling), our method still outperforms GRPO in both standard and anytime reasoning, highlighting the robustness of our approach.

2 Methodology

In a training paradigm similar to R1-Zero [Guo et al., 2025], the model is tasked with generating a comprehensive CoT within a designated "thinking box" upon receiving a question. Subsequently, the model summarizes the answer based on this thinking process. A rule-based reward is then calculated according to the summarized answer. The RL objective is to maximize the expected reward:

$$\mathcal{J}(\theta) = \mathbb{E}\underbrace{x \sim p_{\mathcal{X}}}_{\text{question}}\mathbb{E}\underbrace{z \sim \pi_\theta(\cdot|x)}_{\text{thinking process}}\mathbb{E}\underbrace{y \sim \pi_\theta(\cdot|x,z)}_{\text{answer}}[r(x,y)] \tag{1}$$

where x represents the question, z denotes the thinking process, y is the summarized answer, and $r(x, y)$ is the reward function.

In previous studies [Zeng et al., 2025, Liu et al., 2025, Luo et al., 2025], the generation of thinking process and summary are typically sampled together. If the thinking process exceeds the predefined generation limit, the response is considered a negative sample. We contend that this approach is impractical, particularly in online services where a valid summary should be provided even if the thinking process is incomplete. We propose decoupling the generation of the thinking process and its summary, allocating separate token budgets for each. When the thinking process is halted due to budget constraints, we insert ellipses followed by a *</think>* to prompt the model to produce a summary (see Appendix A), similar to Muennighoff et al. [2025] and Qu et al. [2025].

To differentiate between the thinking and summary policies, we denote the thinking policy as π_θ and the summary policy as π_ϕ. By defining $r_\phi(x, z) = \mathop{\mathbb{E}}_{y\sim\pi_\phi(\cdot|x,z)}[r(x, y)]$, the objective can be expressed as:

$$\mathcal{J}(\theta, \phi) = \mathop{\mathbb{E}}_{x\sim p_\mathcal{X}, z\sim\pi_\theta(\cdot|x)}[r_\phi(x, z)]\,. \tag{2}$$

Given that $|y| \ll |z|$, multiple summaries can be sampled to better estimate the expected reward for each thinking process, while incurring only a small computational overhead.

2.1 Optimizing Anytime Reasoning

Test-time scaling [OpenAI, 2024] is crucial for enhancing the reasoning capabilities of LLMs. This concept operates on the premise that increased computational effort during the reasoning process generally leads to better performance. However, in typical RL training setups like R1-Zero-like [Guo et al., 2025], the performance on anytime reasoning is not guaranteed. The reward evaluation is based on the entire thinking process, lacking insight into whether incremental thinking consistently improves performance [Qu et al., 2025].

To optimize anytime reasoning, we propose sampling the thinking budget from a prior distribution rather than using a fixed token budget. Let b represent the token budget for thinking, sampled from a prior distribution $p_\mathcal{B}$ over a set of increasing budgets $\{b_1, \ldots, b_m\}$ ($P_j = p_\mathcal{B}(b = b_j)$ for simplicity). The anytime reasoning objective is:

$$\mathcal{J}_{\text{anytime}}(\theta, \phi) = \mathop{\mathbb{E}}_{b\sim p_\mathcal{B}, x\sim p_\mathcal{X}, z\sim\pi_\theta(\cdot|x)}[r_\phi(x, z_{\le b})] = \mathop{\mathbb{E}}_{x\sim p_\mathcal{X}, z\sim\pi_\theta(\cdot|x)}\left[\sum_{j=1}^{m} P_j r_\phi(x, z_{\le b_j})\right], \tag{3}$$

where $z_{\le b}$ is the truncated thinking process at length of the token budget b,

$$z_{\le b} = \begin{cases} z, & \text{if } b \ge |z| \\ \text{truncate}(z, b), & \text{if } b < |z| \end{cases}.$$

Instead of focusing solely on the final score based on the entire thinking process as in standard reasoning task, we maximize the expected score over all possible budgets with distribution $p_\mathcal{B}$. As illustrated in Figure 1, this is akin to maximizing the area under the score curve when $p_\mathcal{B}$ is a uniform distribution across every token budget. However, evaluating for all token budgets is impractical and unnecessary, so we evaluate the score only at a small predefined budget support (with $m \le 8$ in our experiments).

It is important to note that this approach transforms the problem into a dense reward framework, introducing verifiable dense rewards for each thinking budget. This facilitates better credit assignment during RL training and enhances the identification of each component's contribution to a successful reasoning process. As illustrated in Figure 2, the dense rewards for budgets prior to reaching a correct answer are low. However, the cumulative return is relatively higher if the reasoning process ultimately arrives at a correct answer. In contrast, the cumulative return after the first correct answer is relatively low, localizing and highlighting the tokens that contributed to the initial correct answer. This approach is distinct from typical sparse reward RL training for standard reasoning tasks, where all tokens receive the same return. Such sparse reward structures typically lead to unstable and inefficient RL training, while our dense reward approach provides more informative learning signals throughout the entire reasoning process.

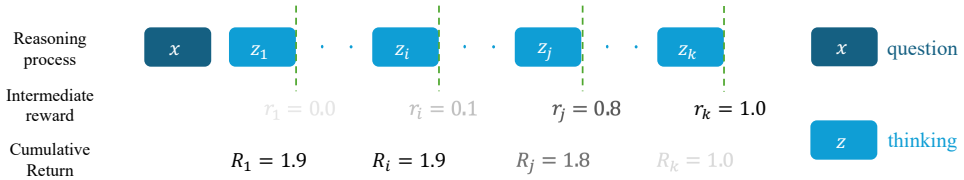


Figure 2: By introducing dense rewards, we achieve better credit assignment during RL training. We assume a uniform distribution over thinking budgets and omit the probability for simplicity.

Relation to Standard Reasoning Tasks A larger thinking budget is supposed to yield better performance in expectation. Since $z_{\leq b}$ is always a prefix of z, the optimal summary policy π_{ϕ^*} should satisfy:

$$\mathop{\mathbb{E}}_{z\sim\pi_\theta(\cdot|x)}\left[r_{\phi^*}(x, z_{\leq b})\right] \leq \mathop{\mathbb{E}}_{z\sim\pi_\theta(\cdot|x)}\left[r_{\phi^*}(x, z)\right], \tag{4}$$

for any b and x. Then we have:

$$\mathcal{J}_{\text{anytime}}(\theta, \phi^*) \leq \mathcal{J}(\theta, \phi^*) \tag{5}$$

This justifies the anytime reasoning objective as a lower bound of the standard reasoning objective. Therefore, maximizing performance in anytime reasoning should also enhance performance in standard reasoning tasks. In an extreme case where $P_m = 1$ (training only with full reasoning length), $\mathcal{J}_{\text{anytime}}$ falls back to the standard reasoning objective $\mathcal{J}$. For detailed proof, refer to Appendix C.

2.2 Budget Relative Policy Optimization

By defining $j_t = \arg\min_j b_j \geq t$, which represents the nearest token budget after t, the gradient for the thinking policy can be computed as follows:

$$\nabla_\theta \mathcal{J}_{\text{anytime}}(\theta, \phi) = \mathop{\mathbb{E}}_{x\sim p_{\mathcal{X}}, z\sim\pi_\theta(\cdot|x)}\left[\sum_{t=1}^{|z|} \nabla_\theta \log \pi_\theta(z_t|x, z_{<t}) \left(R(x, z, j_t) - V(x, z_{<t})\right)\right], \tag{6}$$

where

$$R(x, z, j_t) = \sum_{j=j_t}^{m} P_j r_\phi(x, z_{\leq b_j}),$$

and $V(x, z_{<t})$ is the variance reduction term, which should be a function correlated to $R(x, z, j_t)$ but invariant with respect to z_t.

Typically, we set $V(x, z_{<t}) = \mathop{\mathbb{E}}_{z_{\geq t}\sim\pi_\theta(\cdot|x, z_{<t})}[R(x, [z_{<t}, z_{\geq t}], j_t)]$, representing the expected future return [Sutton and Barto, 2018]. In traditional RL, GAE[Schulman et al., 2015] is often used by estimating this value with a critic model. However, training a critic model for LLM can be both costly and noisy [Guo et al., 2025]. An alternative is sampling-based approach, as in VinePPO [Kazemnejad et al., 2024] and Remax [Li et al., 2023], but this requires significant additional computation across all thinking budgets. Group-based methods, such as GRPO [Shao et al., 2024] and RLOO [Ahmadian et al., 2024], treat generation as a bandit and use the average score of multiple responses for variance reduction. However, they are unsuitable in our scenario due to the presence of dense rewards.

In LLM generation, newly sampled tokens (actions) are consistently appended to the existing context (states). This implies that the current context ($z_{<t}$) always serves as a prefix for any future context ($[z_{<t}, z_{\geq t}]$). This unique property distinguishes it from traditional RL but is often overlooked. Assuming a perfect summary policy that consistently extracts the best answer from the thinking process, the reward should increase monotonically with the number of generated tokens, satisfying $r_\phi(x, z_{<t}) \leq r_\phi(x, [z_{<t}, z_{\geq t}])$. Consequently, the current reward $r_\phi(x, z_{<t})$ is correlated with any future reward $r_\phi(x, [z_{<t}, z_{\geq t}])$, particularly when t is large enough to yield a correct answer or when $|z_{<t}| \gg |z_{\geq t}|$. This correlation justifies its use as a suitable baseline for variance reduction.

Building on this insight, we introduce **B**udget **R**elative **P**olicy **O**ptimization (**BRPO**) for efficient variance reduction. Specifically, we employ the following variance reduction term:

$$V_1 = \frac{\sum_{j=1}^{j_t-1} \lambda^{j_t-j} r_\phi(x, z_{\leq b_j})}{\sum_{j=1}^{j_t-1} \lambda^{j_t-j}} \sum_{j=j_t}^{m} P_j, \tag{7}$$

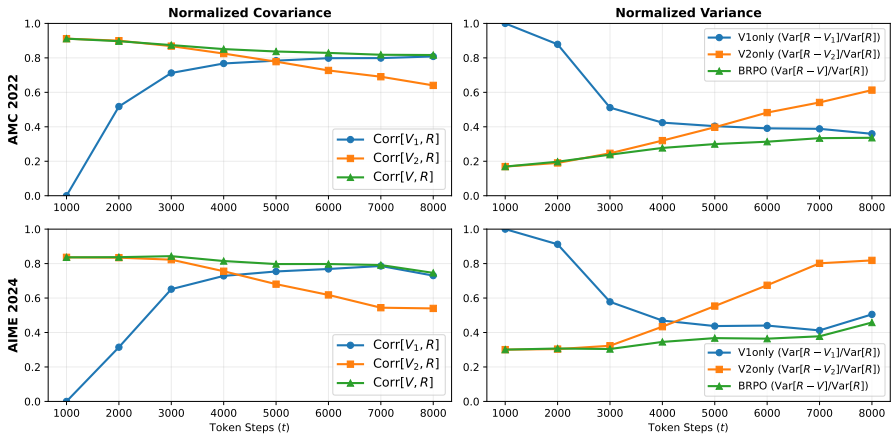


Figure 3: **Left**: The correlation coefficient of V_1 and V_2 with $R(x, z, j_t)$. **Right**: The normalized variance of our BRPO. We evaluate the R1-Distill-1.5B model under the scenario where $\lambda = 0.5$, and $p_{\mathcal{B}}$ is a uniform distribution over $\{1000, 2000, ..., 8000\}$.

where the evaluated scores at previous budgets, weighted by a discount factor λ, serve as the reward baseline (highlighted in red), and are multiplied by the sum of probabilities after j_t to align with the scale of $R(x, z, j_t)$.

As illustrated in Figure 3, when t is small, the effectiveness of V_1 may diminish because a short thinking process $z_{<t}$ provides limited information. In such cases, we apply a variant of GRPO as a complement. We sample a set of thinking processes $\{z^1, z^2, \ldots, z^G\}$ and compute:

$$V_2 = \frac{1}{G} \sum_{i=1}^{G} R(x, z^i, j_t), \tag{8}$$

which represents the expected return after j_t given the question x. Note that the correlation between V_2 and $R(x, z, j_t)$ decreases as t increases, as shown in Figure 3, due to differing prefixes ($z_{<t}$) in these thinking processes.

By combining V_1 and V_2, the overall variance reduction term is:

$$V(x, z_{<t}) = \frac{j_t - 1}{m} V_1 + \frac{m - j_t + 1}{m} V_2. \tag{9}$$

As demonstrated in Figure 3, our BRPO significantly outperforms GRPO in reducing variance, especially when the thinking is long.

2.3 Decoupled Optimization for Thinking and Summary

In a rigorous derivation, the optimization of thinking and summary policies should share the same prior budget distribution $p_{\mathcal{B}}$. However, an optimal summary policy is crucial when the thinking process is incomplete, and its effectiveness is significantly influenced by $p_{\mathcal{B}}$. An imbalanced prior distribution can lead to suboptimal summary policy. To achieve a robust anytime reasoning performance, we decouple the optimization of thinking and summary policies by using a different budget distribution, $p'_{\mathcal{B}}$, for the summary policy. The decoupled gradient of the summary policy with respect to the anytime reasoning objective 3 can be computed as follows:

$$\nabla_\phi \mathcal{J}_{\text{anytime}}(\theta, \phi) = \mathop{\mathbb{E}}_{x \sim p_{\mathcal{X}}, z \sim \pi_\theta(\cdot|x)} \left[\sum_{j=1}^{m} P'_j \mathop{\mathbb{E}}_{y \sim \pi_\phi(\cdot|x, z_{\leq b_j})} \left[\nabla_\phi \log(\pi_\phi(y|x, z_{\leq b_j})) r(x, y) \right] \right]. \tag{10}$$

In our experiments, we set $p'_{\mathcal{B}}$ as a uniform distribution over the budget support $\{b_1, \ldots, b_m\}$. We employ a distinct approach to optimize the summary policy. Specifically, for each question x and thinking process $z_{\leq b_j}$, we sample a group of summaries and use GRPO to stabilize the optimization.

Typically, a shared model ($\phi = \theta$) is used for both thinking and summary policies. In such cases, the overall gradient is:

$$\nabla_\theta \mathcal{J}_{\text{anytime}}(\theta) = \nabla_\theta \mathcal{J}_{\text{anytime}}(\theta, \phi)\big|_{\phi=\theta} + \nabla_\phi \mathcal{J}_{\text{anytime}}(\theta, \phi)\big|_{\phi=\theta}.$$

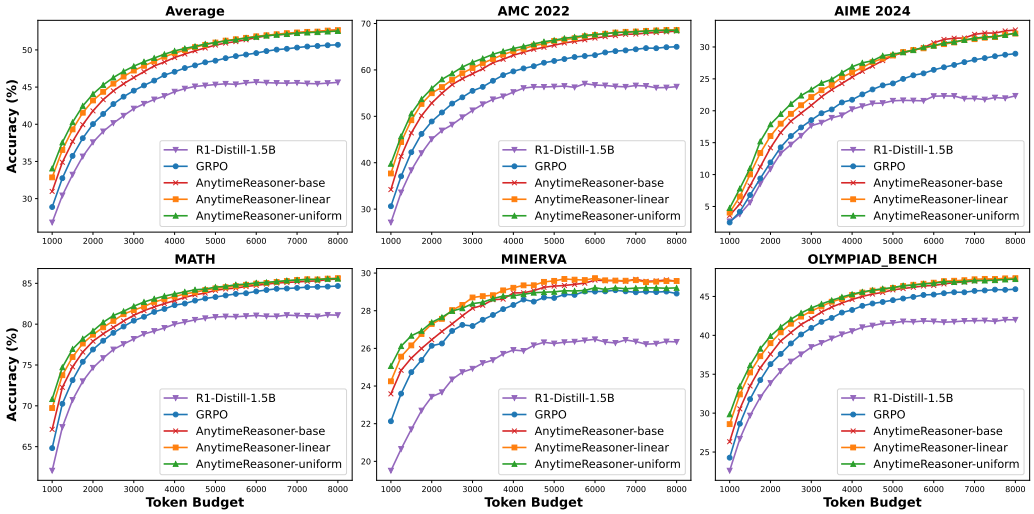


Figure 4: The comparison of anytime reasoning performance between GRPO and our *AnytimeReasoner* with various prior budget distributions. Notably, the accuracies at the maximum token budget (8000) reflect the performance in the standard reasoning task.

3 Experiments

We implement our algorithms based on the Verl framework [Sheng et al., 2024], incorporating several key modifications as detailed in Appendix B. We employ Proximal Policy Optimization (PPO) [Schulman et al., 2017] to optimize both thinking and summary policies. For the thinking policy, we use BRPO to compute the advantage function, as detailed in Section 2.2. During training, we allocate four token budgets ($m = 4$) for thinking: {2000, 4000, 6000, 8000}. For each question, we sample a group of 8 complete thinking processes (stopped either by *</think>* or when exceeding 8000 tokens). We sample 4 answers to calculate the average score at each thinking budget, which is used to compute the advantage function as in Dr. GRPO [Liu et al., 2025]. The summary length is restricted to 128 tokens. We extract the first answer and use a rule-based verifier to determine the 0/1 outcome reward. As detailed in Section 2.3, we employ different prior distributions for the thinking and summary policies. Unless otherwise specified, the prior distribution $p'_{\mathcal{B}}$ for the summary policy is set to a uniform distribution.

We fine-tuned DeepSeek-R1-Distill-Qwen-1.5B [Guo et al., 2025] on 40,315 math problems from DeepScaleR [Luo et al., 2025] for a single epoch, using a batch size of 64 questions per policy iteration. Our experiments were conducted on 8 NVIDIA A100 80G GPUs, with each experiment taking approximately 30 hours to complete (less than 10% overhead in total compared to GRPO). During training, we evaluate the average scores of AIME2024 and AMC2022 every 20 steps and report their performance curves, sampling 32 responses for each question. After training, we assess the final model using five benchmarks: AIME2024 [Li et al., 2024a], AMC2022 [Li et al., 2024a], MATH500 [Hendrycks et al., 2021], Minerva Math [Lewkowycz et al., 2022], and Olympiad Bench [He et al., 2024], with 32 uniform token budgets ranging from 0 to 8000. We compare our methods with GRPO [Shao et al., 2024], incorporating the corrections introduced in Dr. GRPO [Liu et al., 2025].

3.1 Main Results

We consider the following prior distributions $p_{\mathcal{B}}$ when optimizing the thinking policy by equation 3:

- *Base*: We only optimize the final performance as in standard reasoning task, namely $P_m = 1$.
- *Uniform*: We set $p_{\mathcal{B}}$ as a uniform distribution.
- *Linear*: We assign probability proportional to the budget length, such that $p_{\mathcal{B}}(b) \propto b$.

We evaluate the final models after training and plot the score curves under varying thinking budgets in Figure 4. For each question in AMC and AIME, we sample 320 thinking processes to compute the average score. For other datasets, we sample 80 thinking processes per question.

As shown in Figure 4, all variants of our method consistently outperform GRPO by a large margin across varying prior distributions. With small budgets, *AnytimeReasoner-uniform* excels by prioritizing optimization of these budgets. When the thinking budget is large, *AnytimeReasoner* with different prior distributions tends to converge to similar performance, demonstrating the robustness of our approach. Notably, even for *AnytimeReasoner-base*, where we optimize performance only under the maximum thinking budget as in the GRPO baseline, we still achieve significant better performance at all thinking budgets. This improvement is due to the decoupled optimization and our variance reduction technique (discussed further in Section 3.2.3). More details and additional evaluations on longer context can be found in Appendix D.

3.2 Ablations

To further investigate which aspects of our framework contribute to performance improvements, we conduct detailed ablations considering three factors: verifiable dense rewards (Section 3.2.1), decoupled optimization (Section 3.2.2), and variance reduction (Section 3.2.3). We report three metrics during training. *Anytime Accuracy*: the average accuracy over thinking budgets at {2000, 4000, 6000, 8000}. *Final Accuracy*: the accuracy at the maximum budget (8000). *Average Thinking Length*: the average thinking length under the maximum budget (8000).

3.2.1 Verifiable Dense Rewards

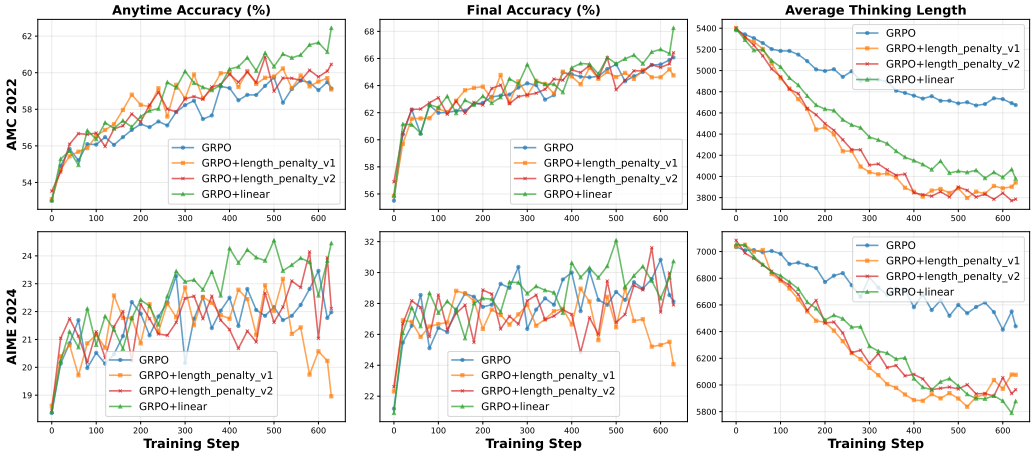


Figure 5: Ablation on verifiable dense rewards. For *GRPO+length_penalty_v1*, we follow Aggarwal and Welleck [2025], assigning reward $1 - \frac{0.2|z|}{b_m}$ for the correct answer and 0 for wrong answer. For *GRPO+length_penalty_v2*, we follow Arora and Zanette [2025] with α as 0.2.

We investigate the effectiveness of verifiable dense rewards by modifying the objective of the thinking policy in equation 3 with a *linear* prior distribution, while keeping the summary policy training consistent with GRPO. Specifically, we use V_2 as the variance reduction term to align with GRPO and eliminate the influence of enhanced variance reduction. To demonstrate our method's superior token efficiency, we compare it against reward shaping, which uses a length penalty on correct answers to encourage concise reasoning [Aggarwal and Welleck, 2025, Arora and Zanette, 2025].

As illustrated in Figure 5, incorporating dense rewards improves both the anytime and final performance. Notably, since our objective diverges from directly optimizing final performance as in the GRPO baseline, the observed improvements can be attributed to enhanced credit assignment facilitated by dense rewards. Another prominent observation is that the average thinking length is clearly shorter than the GRPO baseline under the maximum budget. This is because the thinking policy is encouraged to arrive at a correct answer as quickly as possible, making the model favor

shorter, correct responses. Although reward shaping with length penalty can also reduce the thinking length, it sacrifices the performance and is unstable during training.

3.2.2 Decoupled Optimization

Figure 6: Ablation on decoupled optimization for summary policy.

To study the impact of decoupled optimization for thinking and summary policies (detailed in Section 2.3), we modify the training of summary policy in GRPO to align with *AnytimeReasoner*, while keeping the thinking policy training unchanged. Specifically, we sample 4 answers for each thinking budget in {2000, 4000, 6000, 8000}, applying GRPO within each summary group. This approach trains a summary policy under uniformly distributed thinking budgets, while the thinking policy optimizes performance only under the maximum budget (8000).

As shown in Figure 6, the decoupled GRPO clearly outperforms the vanilla GRPO, especially in the AMC benchmark. Notably, the significant improvement in anytime accuracy (the average score under sampled thinking budgets) indicates that decoupled optimization results in a better summary policy for anytime reasoning.

3.2.3 Variance Reduction

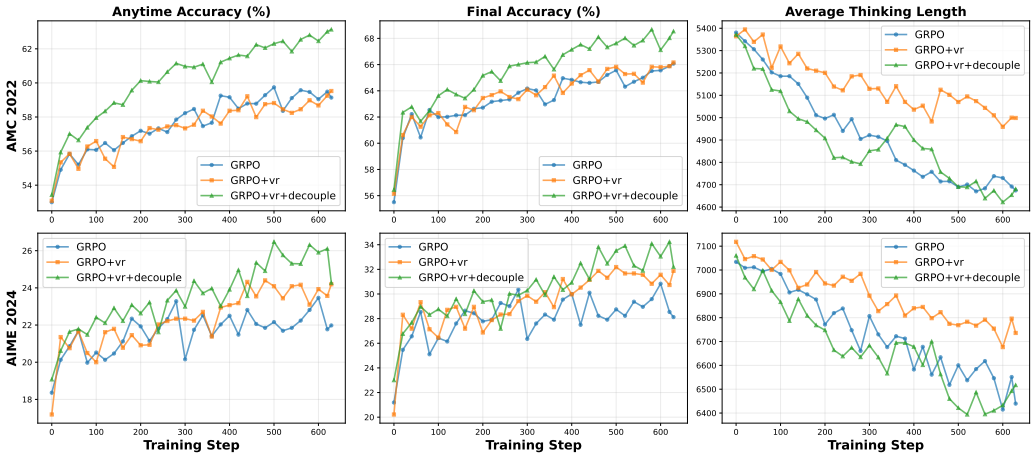


Figure 7: Ablation on variance reduction.

To evaluate the effectiveness of our BRPO variance reduction (as detailed in Section 2.2), we modified the training of the thinking policy by incorporating BRPO's variance reduction techniques, while maintaining the summary policy training consistent with GRPO. Specifically, we set $m = 4$ and $P(b_m) = 1$ in equation 7, aligning the objective exactly with GRPO.

Figure 7 shows that our approach enhances performance on the AIME benchmark. As discussed in Section 3.2.2, the suboptimal summary policy in GRPO may constrain the potential of BRPO's effectiveness. To address this, we introduced decoupled optimization (detailed in Section 2.3) to improve the summary policy, resulting in further performance gains.

3.3 Evaluation on 7B Model

We also evaluated our approach on a larger model, DeepSeek-R1-Distill-Qwen-7B. For this experiment, we modified the training setup by running two epochs on the DeepScaleR dataset with a batch size of 128 questions per iteration. We incorporated the *clip higher* technique from DAPO[Yu et al., 2025] to prevent entropy collapse[Cui et al., 2025b] observed in the training. As shown in Figure 8, our *AnytimeReasoner* framework achieves clearly superior performance in anytime reasoning, with a maximum improvement of about 5 absolute points. For standard reasoning, our methods outperform the GRPO baseline for most of the time during training, despite high variance in the final accuracy..

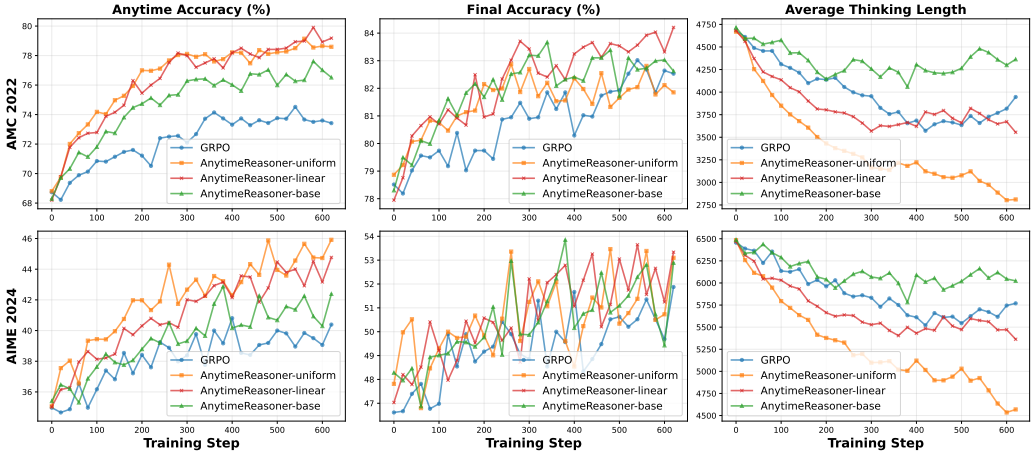


Figure 8: The training curves for DeepSeek-R1-Distill-Qwen-7B.

4 Related Works

Reinforcement Learning with Verifiable Rewards Since the introduction of DeepSeek-R1 [Guo et al., 2025], a growing body of research has adopted the reinforcement learning with verifiable rewards (RLVR) paradigm [Lambert et al., 2024] to improve the reasoning capabilities of large language models (LLMs). SimpleRL [Zeng et al., 2025] provides the first open-source replication of R1-Zero in mathematical domains and analyzes RL dynamics across various base models. Hu et al. [2025] demonstrate that removing the KL regularization used in RLHF [Christiano et al., 2017] improves both RL efficiency and asymptotic performance. Liu et al. [2025] identify an optimization bias in GRPO [Shao et al., 2024] and propose Dr.,GRPO, which applies a Monte Carlo policy gradient method with a baseline [Sutton and Barto, 2018]. While these works improve our understanding of R1-Zero-style training, they still depend on sparse outcome-based rewards, which pose challenges for credit assignment and learning efficiency [Kazemnejad et al., 2024]. In contrast, our method introduces a novel policy optimization framework that leverages cheaply estimated *verifiable dense rewards* to improve sample efficiency and learning stability.

Token Budget Efficiency of Reasoning Models Previous efforts have studied budgeted reasoning by reducing response length through prompting [Jin et al., 2024, Nayab et al., 2024, Lee et al., 2025, Ma et al., 2025] or adaptive sampling [Yang et al., 2025]. While these training-free approaches can shorten outputs, they often entail a trade-off between conciseness and task performance. More recent work explores token efficiency within online RL frameworks, enabling models to jointly optimize for accuracy and brevity. Yeo et al. [2025] observe that the output lengths on harder questions tend to grow during RL training, and propose a cosine-shaped reward to constrain length. Liu et al. [2025] trace this issue to optimization bias in GRPO and show that correcting it enhances token efficiency.

Further, Arora and Zanette [2025] and Aggarwal and Welleck [2025] apply explicit reward shaping to target shortened or fixed outputs. Our work differs by operating in an *anytime reasoning* framework, where the reasoning process can be interrupted at anytime and the best-effort solution should be provided [Dean and Boddy, 1988, Zilberstein and Russell, 1995]. Despite not explicitly enforcing conciseness, our objective naturally encourages efficient reasoning, as demonstrated empirically.

Connection to MRT An independent work to ours, MRT [Qu et al., 2025], optimizes test-time compute by minimizing cumulative regret relative to an oracle. Since the oracle is unknown, they employ meta-RL [Xiang et al., 2025, Beck et al., 2023] as an approximation, aiming to maximize the "progress" of each newly generated *episode*. Despite sharing a similar high-level goal, our formulation fundamentally differs. Rather than minimizing regret, we optimize anytime performance by sampling the thinking budget from a prior distribution, remaining tractable with standard RL techniques. These foundational distinctions lead to significant methodological differences. Firstly, our approach operates on a per-token basis, instead of on *episode* which is ambiguous and can be hackable in RL if not well handled. Secondly, our method is grounded in principled RL, explicitly accounting for long-term returns. In contrast, MRT adopts a greedy strategy, optimizing the progress of immediate next episode only. Our experimental results also significantly outperform their reported outcomes. We achieve an accuracy of 32.7% compared to their reported 30.3% on AIME 2024.

5 Conclusion

The effectiveness of test-time scaling in LLM reasoning is commonly attributed to the generation-verification gap [Xiang et al., 2025], where verifying solutions is substantially easier than generating them. During reasoning, the model engages in an iterative search process, exploring potential solutions until a valid one is found. Once generated, the solution is verified for correctness, and this search-verification loop continues until a confident answer is produced.

In this work, we present a framework that systematically exploits this generation-verification gap. Our approach is based on the key observation that verifying answers and extracting them from partial reasoning traces is easy and computationally cheap. Building on this insight, we design our framework to produce answers at some predefined thinking budgets, thereby introducing verifiable dense rewards to enhance RL training. Furthermore, we utilize these additional rewards to construct a more effective variance reduction baseline than GRPO, significantly improving the stability and efficiency of RL training. By integrating these techniques, our framework achieves superior performance in both standard and anytime reasoning tasks.

References

Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning. *arXiv preprint arXiv: 2503.04697*, 2025.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms. *arXiv preprint arXiv:2402.14740*, 2024.

Daman Arora and Andrea Zanette. Training language models to reason efficiently. *arXiv preprint arXiv: 2502.04463*, 2025.

Jacob Beck, Risto Vuorio, Evan Zheran Liu, Zheng Xiong, Luisa Zintgraf, Chelsea Finn, and Shimon Whiteson. A survey of meta-reinforcement learning. *arXiv preprint arXiv:2301.08028*, 2023.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025a.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, et al. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*, 2025b.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35: 16344–16359, 2022.

Thomas L Dean and Mark S Boddy. An analysis of time-dependent planning. In *AAAI*, volume 88, pages 49–54, 1988.

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

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. Token-budget-aware llm reasoning. *arXiv preprint arXiv:2412.18547*, 2024.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*, 2024.

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

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, and Heung-Yeung Shum Xiangyu Zhang. Open-reasoner-zero: An open source approach to scaling reinforcement learning on the base model. https://github.com/Open-Reasoner-Zero/Open-Reasoner-Zero, 2025.

Mingyu Jin, Qinkai Yu, Dong Shu, Haiyan Zhao, Wenyue Hua, Yanda Meng, Yongfeng Zhang, and Mengnan Du. The impact of reasoning step length on large language models. In *ACL (Findings)*, 2024.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. Vineppo: Unlocking rl potential for llm reasoning through refined credit assignment. *arXiv preprint arXiv:2410.01679*, 2024.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. T\" ulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

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

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857, 2022.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, et al. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13:9, 2024a.

Junyan Li, Delin Chen, Tianle Cai, Peihao Chen, Yining Hong, Zhenfang Chen, Yikang Shen, and Chuang Gan. Flexattention for efficient high-resolution vision-language models. In *European Conference on Computer Vision*, pages 286–302. Springer, 2024b.

Ziniu Li, Tian Xu, Yushun Zhang, Zhihang Lin, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. Remax: A simple, effective, and efficient reinforcement learning method for aligning large language models. *arXiv preprint arXiv:2310.10505*, 2023.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. https://github.com/agentica-project/deepscaler, 2025.

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

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

Sania Nayab, Giulio Rossolini, Marco Simoni, Andrea Saracino, Giorgio Buttazzo, Nicolamaria Manes, and Fabrizio Giacomelli. Concise thoughts: Impact of output length on llm reasoning and cost. *arXiv preprint arXiv: 2407.19825*, 2024.

OpenAI. Learning to reason with llms, 2024. URL https://openai.com/index/learning-to-reason-with-llms/.

Yuxiao Qu, Matthew YR Yang, Amrith Setlur, Lewis Tunstall, Edward Emanuel Beeching, Ruslan Salakhutdinov, and Aviral Kumar. Optimizing test-time compute via meta reinforcement fine-tuning. *arXiv preprint arXiv:2503.07572*, 2025.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

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

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.

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

Violet Xiang, Charlie Snell, Kanishk Gandhi, Alon Albalak, Anikait Singh, Chase Blagden, Duy Phung, Rafael Rafailov, Nathan Lile, Dakota Mahan, et al. Towards system 2 reasoning in llms: Learning how to think with meta chain-of-though. *arXiv preprint arXiv:2501.04682*, 2025.

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Zheng Lin, Li Cao, and Weiping Wang. Dynamic early exit in reasoning models. *arXiv preprint arXiv: 2504.15895*, 2025.

Edward Yeo, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. Demystifying long chain-of-thought reasoning in llms. *arXiv preprint arXiv:2502.03373*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv preprint arXiv:2503.18892*, 2025.

Shlomo Zilberstein and Stuart Russell. Approximate reasoning using anytime algorithms. In *Imprecise and approximate computation*, pages 43–62. Springer, 1995.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We focus on optimizing anytime reasoning task, and present a novel framework in Section 2.1.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [NA]

Justification: There is no obvious limitation.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We provide assumptions in Section 2.1 and proofs in Appendix C.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide implement details in Section B, 3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The data is public accessible. We will open-source our code.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide training and test details in 3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: For the training curves in ablation studies (Section 3.2.1, 3.2.2, 3.2.3), the experiments are expensive (8 GPUs for 30 hours). For the main results in Section 3.1, we run many enough data points to report the average score.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide these information in Section 3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: No ethics issue involved.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: It is just a research task, no any societal impact.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We do not release any data and model, there is no risk.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The code and data is cited in Section 3.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We do not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: This paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: No such risks.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

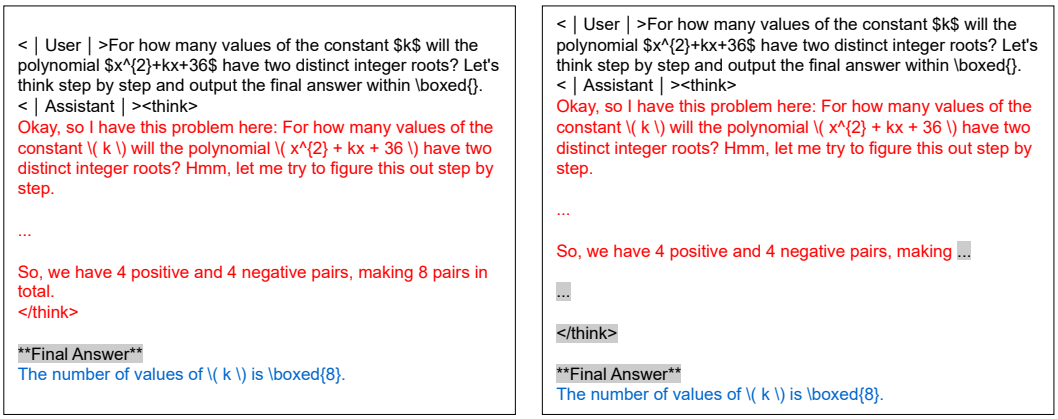
- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

Appendix

Table of Contents

A Implementation Details

We illustrate the implementation details about how we truncate the reasoning process and prompt the model to output an answer.



(a) Thinking is stopped by *</think>*. (b) Thinking is stopped due to out of budget.

Figure 9: We decouple the generation of thinking and its summary. Given the question, the model first generates the thinking, which can be stopped by a special token *</think>* or the budget limit. Then we insert $**$ Final Answer $**$ (and two ellipsis $\cdots$ plus *</think>* for out of budget cases) to prompt the model to summarize the answer. In training, these inserted tokens will be ignored when calculating the loss.

B Tree-like Generation and Training

Unlike previous methods with sequential question-response generation and training, our approach employs a tree-like structure. In this section, we introduce how to address implementation challenges for efficient training.

During generation, we use the prefix caching feature of vLLM [Kwon et al., 2023] to reuse computations. We sample a complete thinking process z for a question x, then split it based on predefined token budgets ($\{i, j, k\}$ in Figure 10). Each partial thinking process is appended with a special end-of-think token (*</think>*), and the model is prompted to output the answer directly (see Appendix A for more details).

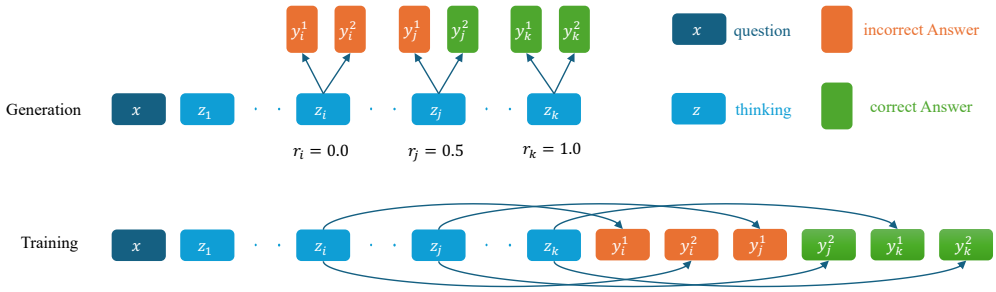


Figure 10: Our methods utilize a tree-like structure for generation and training.

During training, each response is typically concatenated with its corresponding question using FlashAttention [Dao et al., 2022] for speed. However, this introduces significant duplicated computation for tree-like structures, making it impractical due to high computational demands for LLM training. We implement a tree structure attention mask based on FlexAttention [Li et al., 2024b]. As shown in Figure 10, we append all summaries at the end of the thinking process and record their connection positions in a 1D tensor. This tensor is converted to a block mask by FlexAttention, avoiding 2D tensors that can cause out-of-memory issues for long generation lengths.

C Relation Between Standard and Anytime Reasoning

In this section, we provide a proof for the inequality below:

$$\mathcal{J}_{\text{anytime}}(\theta, \phi^*) \leq \mathcal{J}(\theta, \phi^*) \leq \frac{1}{P_m} \mathcal{J}_{\text{anytime}}(\theta, \phi^*).$$

According to equation 4, we have:

$$\mathop{\mathbb{E}}_{z \sim \pi_\theta(\cdot|x)} \left[r_{\phi^*}(x, z_{\leq b}) \right] \leq \mathop{\mathbb{E}}_{z \sim \pi_\theta(\cdot|x)} \left[r_{\phi^*}(x, z) \right],$$

Thus, it follows that:

$$\begin{aligned} \mathcal{J}_{\text{anytime}}(\theta, \phi^*) &= \mathop{\mathbb{E}}_{x \sim p_{\mathcal{X}}, z \sim \pi_\theta(\cdot|x)} \left[\mathop{E}_{b \sim p_{\mathcal{B}}} \left[r_\phi(x, z_{\leq b}) \right] \right] \\ &\leq \mathop{\mathbb{E}}_{x \sim p_{\mathcal{X}}, z \sim \pi_\theta(\cdot|x)} \left[r_\phi(x, z) \right] \\ &= \mathcal{J}(\theta, \phi^*). \end{aligned} \tag{11}$$

Assuming $r(x, y) \geq 0$, which is always achievable by adding a constant to each reward, we also have:

$$\begin{aligned} \mathcal{J}_{\text{anytime}}(\theta, \phi^*) &= \mathop{\mathbb{E}}_{x \sim p_{\mathcal{X}}, z \sim \pi_\theta(\cdot|x)} \left[\mathop{E}_{b \sim p_{\mathcal{B}}} \left[r_\phi(x, z_{\leq b}) \right] \right] \\ &\geq \mathop{\mathbb{E}}_{x \sim p_{\mathcal{X}}, z \sim \pi_\theta(\cdot|x)} \left[P_m r_\phi(x, z_{\leq b_m}) \right] \\ &= \mathop{\mathbb{E}}_{x \sim p_{\mathcal{X}}, z \sim \pi_\theta(\cdot|x)} \left[P_m r_\phi(x, z) \right] \\ &= P_m \mathcal{J}(\theta, \phi^*). \end{aligned} \tag{12}$$

Combining 11 and 12, we can get

$$\mathcal{J}_{\text{anytime}}(\theta, \phi^*) \leq \mathcal{J}(\theta, \phi^*) \leq \frac{1}{P_m} \mathcal{J}_{\text{anytime}}(\theta, \phi^*). \tag{13}$$

This completes the proof.

Algorithm	AMC22	AIME24	MATH500	Minerva	OlympiadBench	**Avg.**
R1-Distill-1.5B	56.4	22.3	81.1	26.3	42.0	45.6
GRPO	65.0	28.9	84.7	28.9	45.9	50.7
AR-base	68.4	**32.7**	85.5	**29.6**	**47.3**	**52.7**
AR-linear	**68.6**	32.1	**85.6**	**29.6**	**47.3**	52.6
AR-uniform	68.5	32.2	**85.6**	29.2	47.2	52.5

Table 1: The **Final Accuracy** by evaluating the maximum budget (8000) for the final models.

Algorithm	AMC22	AIME24	MATH500	Minerva	OlympiadBench	**Avg.**
R1-Distill-1.5B	48.2	16.3	74.5	24.1	36.0	39.8
GRPO	53.4	19.0	77.2	26.6	38.8	43.0
AR-base	57.0	21.9	78.2	27.3	40.2	44.9
AR-linear	58.2	22.3	79.0	**27.7**	40.9	45.6
AR-uniform	**58.8**	**22.9**	**79.4**	27.5	**41.2**	**46.0**

Table 2: The **Anytime Accuracy** by evaluating 32 budgets (every 250 tokens) for the final models.

D Experimental Results

D.1 Main Results

We present the training curves of our *AnytimeReasoner* in Figure 11, corresponding to the experiments in Section 3.1. We also evaluate the performance of the models at training step of 600, and report the final accuracy in Table 1 and the anytime accuracy in Table 2.

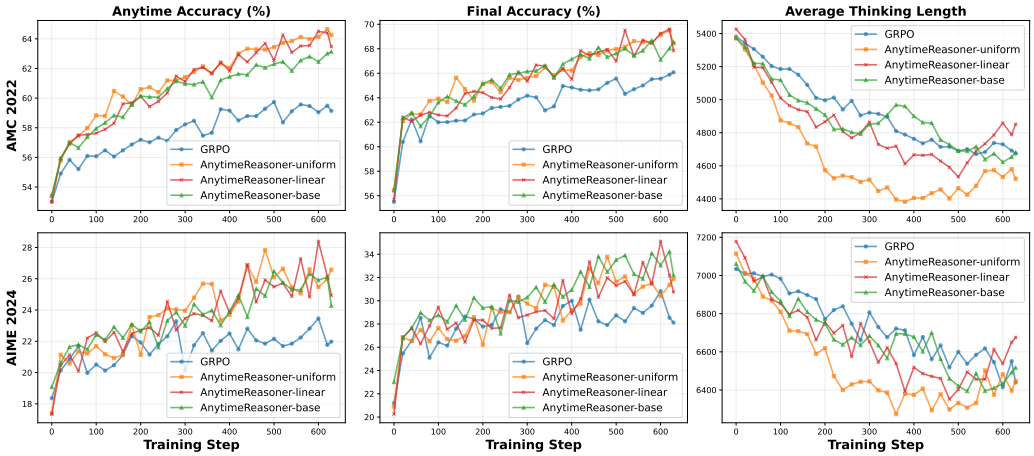


Figure 11: The training curves for main results.

D.2 Evaluation on 16k Context Length

We also verified the effectiveness of our method on longer context lengths by fine-tuning DeepSeek-R1-Distill-Qwen-1.5B up to a 16k maximum context. We used a *uniform* prior over eight thinking budgets: {2k, 4k, 6k, 8k, 10k, 12k, 14k, 16k}. As shown in Figure 12, our approach consistently achieves stronger performance in both anytime and standard reasoning.

Figure 12: The training curves for DeepSeek-R1-Distill-Qwen-1.5B with maximum 16k context length.