Incentivizing Dual Process Thinking for Efficient Large Language Model Reasoning

Xiaoxue Cheng[1,2,3*], **Junyi Li**[4*], **Zhenduo Zhang**[5], **Xinyu Tang**[1], **Wayne Xin Zhao**[1,2,3†], **Xinyu Kong**[5], **Zhiqiang Zhang**[5]
[1] Gaoling School of Artificial Intelligence, Renmin University of China
[2] Beijing Key Laboratory of Research on Large Models and Intelligent Governance
[3] Engineering Research Center of Next-Generation Intelligent Search and Recommendation, MOE
[4] Department of Data Science, City University of Hong Kong [5] Ant Group
chengxiaoxue@ruc.edu.cn, junyili@cityu.edu.hk, batmanfly@gmail.com

Abstract

Large reasoning models (LRMs) have demonstrated strong performance on complex reasoning tasks, but often suffer from overthinking, generating redundant content regardless of task difficulty. Inspired by the dual process theory in cognitive science, we propose **A**daptive **C**ognition **P**olicy **O**ptimization (**ACPO**), a reinforcement learning framework that enables LRMs to achieve efficient reasoning through adaptive cognitive allocation and dynamic system switch. ACPO incorporates two key components: (1) introducing system-aware reasoning tokens to explicitly represent the thinking modes thereby making the model's cognitive process transparent, and (2) integrating online difficulty estimation and token length budget to guide adaptive system switch and reasoning during reinforcement learning. To this end, we propose a two-stage training strategy. The first stage begins with supervised fine-tuning to cold start the model, enabling it to generate reasoning paths with explicit thinking modes. In the second stage, we apply ACPO to further enhance adaptive system switch for difficulty-aware reasoning. Experimental results demonstrate that ACPO effectively reduces redundant reasoning while adaptively adjusting cognitive allocation based on task complexity, achieving efficient hybrid reasoning.

1 Introduction

Recent advances in large reasoning models (LRMs) [1] have demonstrated remarkable success on complex tasks such as mathematical reasoning [2, 3, 4, 5], largely attributed to reinforcement learning that encourages the generation of detailed, step-by-step reasoning processes. LRMs improve answer accuracy through self-reflection and self-verification during long reasoning paths. As the reasoning length increases, the performance of model tends to improve accordingly [2, 6, 7]. Although the long chain-of-thought (CoT) [8] reasoning in LRMs is effective for solving complex problems [2, 9], it often leads to *overthinking* [10, 11], producing redundant reasoning paths. Most existing LRMs rely on fixed reasoning strategies, lacking the ability to dynamically switch between different thinking modes based on task complexity. This rigidity results in inefficient inference, particularly for simple problems that could be resolved more effectively with concise and direct reasoning.

Several recent efforts have explored long CoT compression for efficient reasoning [12, 13]. One line of work fine-tunes LRMs using supervision from shorter chain-of-thought exemplars [14, 15], encouraging the model to arrive at correct answers with fewer intermediate steps. Another line

*Equal Contribution.
†Corresponding author.

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

 https://doi.org/10.52202/085713-5108

introduces length penalties into reinforcement learning reward functions [3, 16, 17, 18], explicitly discouraging unnecessarily long reasoning trajectories during training. Since many prior approaches overlook task difficulty and treat all samples uniformly with the sole objective of shortening reasoning paths, some recent methods attempt to address this by estimating length budgets offline and training the model on sampled trajectories accordingly [19]. However, such offline strategies depend on precomputed budgets and fixed preference data, restricting their adaptability and scalability.

Inspired by the dual process theory in cognitive science [20], which states that humans have two systems for thinking — fast, intuitive thinking (System 1) and slow, deliberate thinking (System 2), we pose the following research question: *Can LRMs learn to dynamically switch between fast and slow thinking modes based on task complexity, enabling more efficient and adaptive reasoning?*

In this paper, we propose **A**daptive **C**ognition **P**olicy **O**ptimization (**ACPO**), a reinforcement learning framework that enables LRMs to perform efficient and adaptive reasoning through dynamic system switch between fast and slow thinking modes. To achieve this, we first introduce system-aware reasoning tokens (*e.g.,* `<fast_think>`, `<slow_think>`) to explicitly indicate the model engagement in fast or slow thinking modes during the reasoning process. Based on these tokens, we construct a dataset from which the reasoning trajectory is interleaved with fast thinking and slow thinking steps, followed by the final answer. Then, we propose a two-stage training strategy to enable dynamic system switch reasoning. In the first stage, we perform supervised fine-tuning with the constructed dataset as a cold start, establishing the model's foundational ability to generate explicit thinking process. In the second stage, we apply reinforcement learning with ACPO for further enhancement. Specifically, we introduce an online token length budget (TLB) mechanism that estimates task difficulty based on the model's sampling success rate, providing a real-time signal to adjust the reasoning budget according to task complexity. These estimates are further incorporated into a reward function that guides cognitive allocation through two components: a TLB reward that encourages difficulty-aware length control, and a system pattern reward that incentivizes appropriate system switch between fast and slow thinking modes.

We evaluate ACPO on a range of complex reasoning benchmarks. Unlike traditional reinforcement learning methods that focus solely on accuracy or fixed-length compression, ACPO dynamically adjusts both reasoning length and cognitive effort based on task difficulty. For challenging problems, it effectively reduces redundant reasoning steps, while for simpler tasks, it avoids overcompression and maintains high accuracy. These results highlight the advantages of difficulty-aware reasoning, demonstrating the effectiveness of dynamic cognitive control through adaptive reward optimization. The main contributions of this work are as follows:

- We introduce system-aware reasoning tokens to explicitly annotate fast and slow thinking steps in LRMs, enabling transparent reasoning paths and providing a foundation for dynamic control of the model's cognitive strategies.
- We propose ACPO, a reinforcement learning framework that integrates online difficulty estimation and token length budget to dynamically calibrate the reward function and steer difficulty-aware cognitive allocation.
- We demonstrate through extensive experiments that ACPO effectively compresses redundant reasoning content while maintaining accuracy, achieving a robust balance between efficiency and performance by avoiding overcompression and underexploration.

2 Preliminary

2.1 Group Relative Policy Optimization (GRPO)

Group Relative Policy Optimization (GRPO) [21] is a reinforcement learning framework that eliminates the need for a value function by estimating advantages in a group-relative manner. Given a specific question-answer pair (q, a), the behavior policy $\pi_{\theta_{\text{old}}}$ samples a group of G individual responses $\{y_i\}_{i=1}^{G}$. For each token $y_{i,t}$ in response y_i, its normalized advantage is computed based on the group-level rewards $\{R_i\}_{i=1}^{G}$ as follows:

$$\hat{A}_{i,t} = \frac{R_i - \text{mean}(\{R_i\}_{i=1}^{G})}{\text{std}(\{R_i\}_{i=1}^{G})}. \tag{1}$$

https://doi.org/10.52202/085713-5108

Similar to Proximal Policy Optimization (PPO) [22], GRPO adopts a clipped surrogate objective with an additional KL regularization term to stabilize optimization:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{(q,a)\sim\mathcal{D},\{y_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|q)} \left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|y_i|}\sum_{t=1}^{|y_i|}\left(\min\left(r_{i,t}(\theta)\hat{A}_{i,t},\ \text{clip}\Big(r_{i,t}(\theta),1-\varepsilon,1+\varepsilon\Big)\hat{A}_{i,t}\right)-\beta D_{\text{KL}}(\pi_\theta||\pi_{\text{ref}})\right)\right], \quad (2)$$

where $r_{i,t}(\theta) = \frac{\pi_\theta(y_{i,t}|q,y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t}|q,y_{i,<t})}$ is the importance sampling ratio for the t-th token $y_{i,t}$.

2.2 Token Length Budget

The token length budget (TLB) [19] is a method introduced for estimating an appropriate token-level budget for reasoning trajectories. Instead of prescribing a fixed or hard constraint, TLB adaptively adjusts the expected output length based on the sampling accuracy of candidate responses, capturing the inherent difficulty of the task. Specifically, for a given question q, N candidate responses are sampled from a language model. The TLB, denoted as L_{budget}, is computed as:

$$L_{\text{budget}} = p \cdot L_r + (1-p)\cdot L_{\text{max}}, \quad (3)$$

where $p = \frac{c}{N}$ represents the sampling success rate, with c denoting the number of correct responses, L_r denotes the average token length among the correct responses, L_{max} is the maximum token length among all responses. This formulation allows the budget to adapt to task complexity, providing flexible length guidance based on sampling accuracy without external supervision. In the DAST method [19], TLB is computed in an offline manner by sampling multiple responses for each question before training. These responses are then converted to preference pairs for further fine-tuning according to their estimated TLB scores.

3 Method

In this section, we propose **ACPO**, a reinforcement learning framework that enables dynamic and adaptive system switch of LRMs to improve reasoning efficiency and adaptability to task complexity. This framework incorporates system-aware reasoning tokens to explicitly represent fast and slow thinking steps of LRMs, and integrates an online length budget estimation in RL training to dynamically adjust reasoning length based on task difficulty, guiding the model to balance accuracy and efficiency through adaptive cognitive effort allocation. We first describe the use of system-aware reasoning tokens for system explicitization, followed by the construction of explicit reasoning paths for supervised fine-tuning. Next, we present the reinforcement learning process of ACPO, providing a detailed overview of the online token length budget estimation and reward design. The overall framework of the proposed ACPO is illustrated in Figure 1.

3.1 Explicit Dual Process Reasoning

The dual process theory [20] models human thinking as a combination of fast, intuitive processes (System 1) and slower, deliberate reasoning (System 2). Inspired by this theory, we introduce *system-aware reasoning tokens* as explicit indicators of different thinking modes within the model's reasoning process. Specifically, we define four special tokens, `<fast_think>`, `</fast_think>`, `<slow_think>` and `</slow_think>`, to explicitly wrap fast and slow reasoning steps, respectively. The introduction of reasoning tokens serves two purposes. First, it makes the model's internal thinking modes explicitly observable, revealing the process of system switch. Second, it provides a fine-grained mechanism for controlling and monitoring the model's cognitive dynamics. With this explicit thinking process, we can train the model for system switch, optimizing its cognitive allocation based on task difficulty.

3.1.1 Data Construction

To enable the model to learn explicit thinking patterns, we construct a training dataset for supervised fine-tuning. Each sample consists of a question paired with its corresponding answer consisting of interleaved fast thinking and slow thinking segments. To ensure the quality of the training data, we utilize the LIMO dataset [23], a carefully curated, high-quality dataset of complex mathematical

 https://doi.org/10.52202/085713-5108

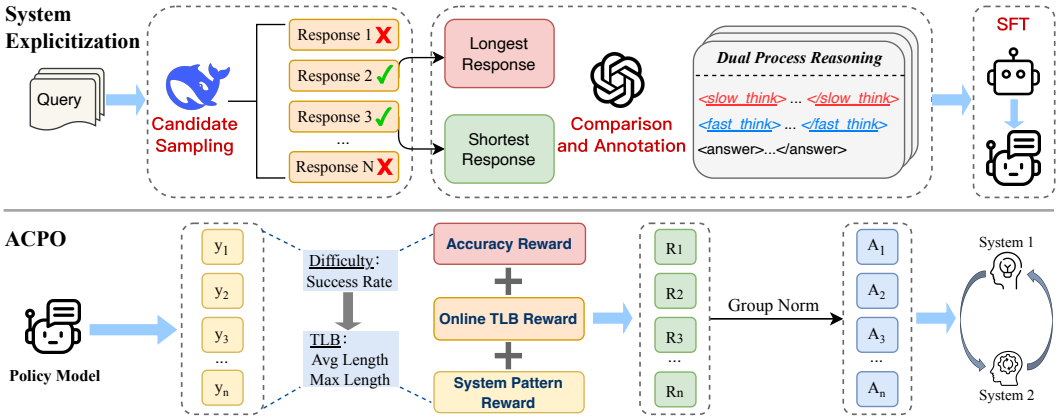

Figure 1: The overall framework of ACPO. The upper section illustrates the system explicitization process and cold start training via SFT. The lower section presents the ACPO training phase.

reasoning tasks with high challenge and diverse knowledge coverage. These tasks demand precise, multi-step reasoning to arrive at correct answers, making them well-suited for constructing explicit thinking processes. Specifically, our data construction involves two main steps: *Candidate Response Sampling* and *Response Comparison and Annotation*.

• **Candidate Response Sampling:** Building on the approach in TOPS [24], we prompt the *DeepSeek-R1-Distill-Qwen-32B* [2] model to generate multiple candidate responses with varying lengths for each question. We adopt the prompting strategies in TOPS to encourage diversity of the reasoning lengths. After generating these candidates, we filter out incorrect responses and select the longest and shortest correct answers for further processing.

• **Response Comparison and Annotation:** We employ *GPT-4* [25] as an evaluator to perform fine-grained comparison and annotation between the selected response pairs. For those reasoning steps that are present in both short and long responses, we consider them as essential and detailed components that should be marked as slow thinking, as they represent critical, non-omittable reasoning processes. For other steps that appear in the long response but are omitted or summarized in the short response, we view them as trivial steps that should be labeled as fast thinking. Based on this principle, we reframe the short response by enclosing essential reasoning steps with `<slow_think></slow_think>` tags and trivial reasoning steps with `<fast_think></fast_think>` tags. An illustrative example is provided in Figure 4.

3.1.2 Supervised Fine-Tuning (Cold Start)

After completing the data construction process, we obtain a dataset containing 745 annotated samples with explicit reasoning tokens. This dataset is used for supervised fine-tuning, allowing the model to learn to generate outputs with clearly interleaved fast and slow thinking modes. This cold start phase equips the model with the foundational ability to explicitly switch between different thinking modes. We train the model with a standard cross-entropy loss to ensure its thinking process aligns with the intended system switch patterns.

3.2 Reinforcement Learning with ACPO

While the SFT process allows the model to align its outputs with the annotated reasoning tokens, explicitly capturing fast and slow thinking patterns, this approach may lead to overfitting and a tendency to memorize fixed reasoning paths. To overcome this limitation, we introduce a reinforcement learning phase, optimizing the fine-tuned model to achieve more flexible and adaptive cognition allocation through interaction and exploration. In the following sections, we present the proposed ACPO method, including the online token length budget estimation and reward design that are specifically tailored for dynamic system switch.

https://doi.org/10.52202/085713-5108

3.2.1 Reward Design

In this work, we adopt GRPO as the underlying RL algorithm, extending its framework with a customized reward structure to explicitly guide the model's reasoning behavior. Formally, during the ACPO optimization process, we sample a set of N candidate responses $\{y_1, y_2, \ldots, y_N\}$ from the current policy model for each training query q_i. For each response, we compute a composite reward that consists of three components: accuracy reward, online TLB reward, and system pattern reward to encourage accurate and efficient reasoning with adaptive cognition allocation, as described below.

Accuracy Reward. For each sampled response y_i, we assess its correctness by comparing the generated answer with the ground-truth label in the training data. The accuracy reward is assigned as:

$$R_{\text{acc},i} = \begin{cases} +1, & \text{if } y_i \text{ is correct,} \\ -1, & \text{otherwise.} \end{cases} \tag{4}$$

Online TLB Reward. To guide the model in generating reasoning paths of appropriate length that align with the task difficulty, we introduce an online token length budget (TLB) reward. Unlike the offline approach in DAST, which requires extensive pre-sampling and pairwise preference construction, our method estimates the length budget L_{budget} online during training, without additional computational overhead. This is achieved by leveraging the natural group-based sampling process in GRPO, allowing the sampling success rate p and token length budget L_{budget} to be directly calculated from the existing candidate responses, as defined in Eq. 3. Given a sampled response y_i with actual length L_i, we compute its TLB reward based on the deviation from the estimated L_{budget}. The online TLB score $R_{\text{TLB},i}$ for each response is then defined as:

$$R_{\text{TLB},i} = \begin{cases} \tanh(-\lambda_i), & \text{if } y_i \text{ is correct,} \\ \tanh(\lambda_i), & \text{otherwise,} \end{cases} \tag{5}$$

where $\lambda_i = \frac{L_i - L_{\text{budget}}}{L_{\text{budget}}}$ and the $\tanh$ function is used to smoothly bound the reward within (-1, 1). The TLB reward adapts the model's reasoning strategy to the difficulty of each task, promoting efficient fast thinking for easier problems while allowing longer and more deliberate reasoning for harder problems. Additionally, the online TLB estimation enables real-time adaptation and avoids reliance on static preference data, leading to better generalization across diverse tasks.

System Pattern Reward. To further encourage difficulty-aware system switch, we use the sampling success rate p from the TLB estimation as a proxy for task difficulty. For easier queries with higher p, the model is encouraged to allocate a larger proportion of its reasoning path to fast thinking. In contrast, for harder queries with lower p, the model is incentivized to allocate more steps to slow thinking for more careful deliberation. Given a sampled response y_i, we compute the proportions of fast and slow thinking tokens within the total reasoning sequence, denoted as $\rho_{\text{fast},i}$ and $\rho_{\text{slow},i}$, respectively. The system pattern reward is then defined as:

$$R_{\text{think},i} = \begin{cases} \rho_{\text{fast},i}, & \text{if } p > p_{\text{thresh}}, \\ \rho_{\text{slow},i}, & \text{otherwise,} \end{cases} \tag{6}$$

where p_{thresh} is a predefined complexity threshold separating easy and hard questions.

The final reward R_i for each sampled response y_i is computed as a weighted combination of the three components:

$$R_i = \begin{cases} \max\left(w_{\text{acc}} \cdot R_{\text{acc},i} + w_{\text{len}} \cdot R_{\text{TLB},i} + w_{\text{think}} \cdot R_{\text{think},i},\ 0.1\right), & \text{if } y_i \text{ is correct,} \\ \min\left(w_{\text{acc}} \cdot R_{\text{acc},i} + w_{\text{len}} \cdot R_{\text{TLB},i} + w_{\text{think}} \cdot R_{\text{think},i},\ -0.1\right), & \text{if } y_i \text{ is incorrect.} \end{cases} \tag{7}$$

The weights w_{acc}, w_{len}, and w_{think} are hyperparameters that balance the importance of different reward signals during optimization. In order to ensure that the reward for correct responses remains strictly positive and the reward for incorrect responses remains strictly negative, we apply $\max$ and $\min$ operations to clip the final reward within the desired range. Based on the final reward, we leverage the GRPO objective in Eq. 2 to optimize the policy model.

 https://doi.org/10.52202/085713-5108

Table 1: Evaluation results of ACPO on three different reasoning models across the MATH 500 and AIME 2024 datasets. **Bold** fonts indicate the best performance for each reasoning model.

Methods	MATH 500			AIME 2024		
	Accuracy	#Token	ACU	Accuracy	#Token	ACU
DeepSeek-R1-Distill-Qwen-1.5B	**83.9**	5708	0.98	28.9	16894	0.11
THINKPRUNE	82.9	2356	2.35	27.0	7574	0.24
ACPO-1.5B	81.0	**1679**	**3.22**	**30.0**	**6670**	**0.30**
DeepScaleR-1.5B-Preview	87.8	3914	1.50	43.1	17206	0.17
L1-Exact	79.8	1044	5.09	16.7	1798	0.62
L1-Max	81.8	999	5.46	23.3	2230	0.69
DeepSeek-R1-Distill-Qwen-7B	**92.8**	3977	0.33	**55.5**	13254	0.06
SFT_Shortest	91.8	2954	0.44	50.0	10757	0.07
SimPO_Shortest	87.8	**970**	**1.29**	33.3	**2737**	**0.17**
SimPO_DAST	92.6	2802	0.47	53.3	6337	0.12
ACPO-7B	91.6	1405	0.93	52.8	4520	**0.17**
DeepSeek-R1-Distill-Llama-8B	**89.1**	5003	0.22	42.9	16374	0.04
ACPO-8B	87.4	**2232**	**0.49**	**43.3**	**7405**	**0.07**

4 Experiment

4.1 Experimental Setup

Datasets and Evaluation Metrics. We conduct training on the DeepScaleR-Preview-Dataset [26], a mathematical dataset consisting of 40K question-answer pairs drawn from AIME, AMC, Omni-Math [27] and STILL [28]. For evaluation, we assess model performance on three mathematical datasets: GSM8K [29], AIME 2024, and MATH 500 [30]. For each test question, we generate 16 responses using a sampling temperature of 0.6 and a top-p value of 0.95, and compute *pass@1* to measure accuracy. We report the average number of tokens generated per response in each dataset to assess reasoning efficiency. Moreover, we use Accuracy per Computation Unit (ACU) metric [31] to capture the balance between reasoning accuracy and efficiency, which is defined as :

$$\text{ACU} = \frac{\text{Accuracy}}{\text{\#Params} \times \text{\#Tokens}} \tag{8}$$

Baselines. We conduct experiments on DeepSeek-R1-Distill-Qwen-1.5B, DeepSeek-R1-Distill-Qwen-7B, and DeepSeek-R1-Distill-Llama-8B [2] and compare ACPO against the following methods.

- **DeepScaleR-1.5B-Preview** [26] is a model trained on DeepSeek-R1-Distill-Qwen-1.5B with GRPO. We include it as an important baseline in our evaluations.
- **THINKPRUNE** [17] trains models with a fixed token limit, pruning unfinished responses beyond this limit with zero reward. We include *THINKPRUNE* with 2k token length constraint trained on DeepSeek-R1-Distill-Qwen-1.5B for comparison.
- **L1** [18] trains models to follow specified response lengths by introducing exact and maximum length penalties in reinforcement learning, yielding two variants, *L1-Exact* and *L1-Max*, both trained on *DeepScaleR-1.5B-Preview*.
- **DAST** [19] introduces an offline approach to estimate token length budgets through sampling and construct length preference data for SimPO [32] training. We include the comparison methods *SFT_Shortest*, *SimPO_Shortest*, and *SimPO_DAST* from DAST in our experiments.

Implementation Details. In the cold start phase, we fine-tune the models for 3 epochs using 745 annotated samples with explicit reasoning tokens. For ACPO training, we adopt the same hyperparameter settings as used in DeepScaleR-1.5B-Preview. Specifically, we use a learning rate of 1×10^{-6}, a batch size of 128, and a maximum context length of 8K tokens during training. The models are trained for one epoch, and both the SFT and RL stages are conducted using the VeRL

Table 3: Performance comparison between three reasoning models trained with GRPO and ACPO on the MATH 500 and AIME 2024 datasets.

Methods	MATH 500			AIME 2024		
	Accuracy	#Token	ACU	Accuracy	#Token	ACU
DeepSeek-R1-Distill-Qwen-1.5B	83.9	5708	0.98	28.9	16894	0.11
+GRPO	84.5	3098	1.82	29.0	12990	0.15
+ACPO	81.0	1679	3.22	30.0	6670	0.30
DeepSeek-R1-Distill-Qwen-7B	92.8	3977	0.33	55.5	13254	0.06
+GRPO	92.5	3700	0.36	53.2	8577	0.08
+ACPO	91.6	1405	0.93	52.8	4520	0.17
DeepSeek-R1-Distill-Llama-8B	89.1	5003	0.22	42.9	16374	0.04
+GRPO	90.4	2172	0.52	43.3	8883	0.06
+ACPO	87.4	2232	0.49	43.3	7405	0.07

framework [33]. We set $p_{\text{thresh}} = 0.5$ in Eq. 6 and set the reward weights as $w_{\text{acc}} = 0.6$, $w_{\text{len}} = 0.3$, and $w_{\text{think}} = 0.1$ in Eq. 7.

4.2 Main Results

The evaluation results of our method and the baselines are presented in Table 1 and Table 2.

For more challenging datasets, such as MATH 500 and AIME 2024 in Table 1, ACPO achieves significant token reduction while maintaining competitive accuracy. For instance, on the AIME 2024 dataset, ACPO-1.5B reaches 30.0% accuracy with an average token count of 6670, representing a 60.5% reduction in token usage compared to 16,894 tokens required by DeepSeek-R1-Distill-Qwen-1.5B. For MATH 500 dataset, ACPO reduces the token count for DeepSeek-R1-Distill-Qwen-7B from 3977 to 1405 with only a slight accuracy decrease. Although L1-Max and L1-Exact also achieve substantial token compression, they suffer noticeable accuracy drops compared to DeepScaleR-1.5B-Preview. This demonstrates that ACPO can effectively shorten reasoning lengths without sacrificing too much accuracy, particularly on complex reasoning tasks.

Table 2: Evaluation results of ACPO with the three reasoning models on the GSM8K dataset.

Methods	Accuracy	#Token	ACU
R1-Distill-Qwen-1.5B	79.9	643	8.28
ACPO-1.5B	81.3	572	9.48
R1-Distill-Qwen-7B	86.5	445	2.78
ACPO-7B	88.3	413	3.05
R1-Distill-Llama-8B	82.9	1026	1.01
ACPO-8B	86.7	732	1.48

For simpler GSM8K dataset in Table 2, where the reasoning paths of DeepSeek-R1-Distill-Qwen-1.5B and 7B tend to be less redundant, ACPO maintains the original length scale while achieving notable accuracy improvements. Specifically, ACPO-1.5B reduces average token usage from 643 to 572 while improving accuracy from 79.9% to 81.3%. Similarly, ACPO-7B achieves 88.3% accuracy with 413 tokens, outperforming the baseline in both accuracy and efficiency. However, for DeepSeek-R1-Distill-Llama-8B, which has relatively long reasoning paths with 1026 tokens, ACPO achieves effective compression by reducing the token count to 732 tokens with 3.8% accuracy improvement. These results highlight the adaptability of the online token length budget estimation, enabling selective compression that preserves necessary reasoning length while avoiding excessive compression.

4.3 Further Analysis

4.3.1 Ablation Analysis

To assess the impact of the reward design in ACPO on both accuracy and reasoning length, we conduct ablation studies by training models with the GRPO reward setting, keeping all other parameters identical to those in ACPO. Specifically, we evaluate the three models on the MATH 500 and AIME 2024 datasets: DeepSeek-R1-Distill-Qwen-1.5B, DeepSeek-R1-Distill-Qwen-7B, and DeepSeek-R1-

 https://doi.org/10.52202/085713-5108

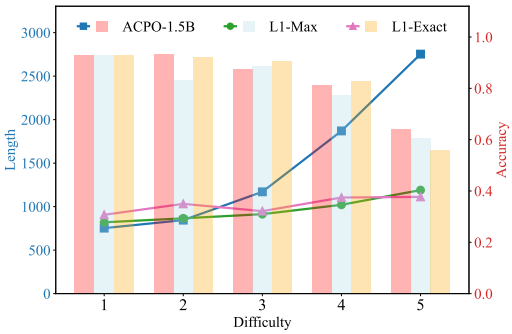

Figure 2: Average response length and accuracy across different difficulty levels on MATH 500.

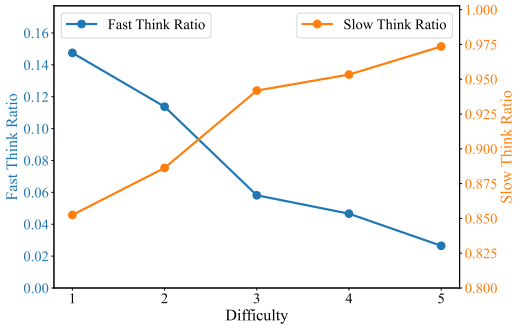

Figure 3: Average fast and slow thinking ratios across different difficulty levels on MATH 500.

Distill-Llama-8B, each trained for one epoch with GRPO and ACPO. We report accuracy, average token length, and ACU scores for comparison in Table 3.

From the results, we observe that both GRPO and ACPO can compress the reasoning length of the original models. In contrast, ACPO consistently achieves more significant compression, often exceeding 50% reduction in token length. For instance, it reduces the average reasoning length of DeepSeek-R1-Distill-Qwen-1.5B on the MATH dataset to 1679 tokens, compared to 3098 tokens for GRPO. In terms of accuracy, GRPO shows a slight advantage, especially on the MATH dataset. However, ACPO demonstrates comparable performance on the more challenging AIME dataset. For DeepSeek-R1-Distill-Qwen-1.5B, ACPO reaches an accuracy of 30.0% compared to 29.0% for GRPO, effectively balancing accuracy and efficiency. While GRPO focuses solely on correctness, the reward design of ACPO jointly optimizes answer accuracy, reasoning length, and system switch, enabling a flexible coordination between correctness, efficiency, and cognitive allocation.

4.3.2 Difficulty Adaptability Analysis

To validate the difficulty adaptability of ACPO, we perform analyses on the MATH 500 dataset, which is divided into five difficulty levels. In Figure 2, we compare the accuracy and average response length of ACPO-1.5B, L1-Exact, and L1-Max. Unlike the L1 approach, which applies uniform token constraints regardless of the difficulty of questions, ACPO demonstrates a more adaptive length control strategy. It effectively shortens responses for simpler problems while preserving the necessary reasoning length for complex questions, resulting in minimal accuracy loss on more challenging levels (*e.g.,* level 4, level 5). This adaptive capability highlights the advantage of our approach in difficulty-aware reasoning, aligning the token length budget more closely with problem complexity.

In Figure 3, we further analyze the fast and slow thinking ratios for ACPO-7B on the MATH dataset. It is observed that, as the problem difficulty increases, the proportion of fast thinking decreases while the slow thinking component increases. Notably, the proportion of slow thinking increases rapidly from difficulty level 1 to level 3, and then grows more gradually beyond level 3. This trend suggests that the model is able to allocate more reasoning effort when solving more difficult problems, dynamically allocating cognitive resources based on task complexity. Such adaptive behavior aligns with the dual process theory of human thinking, demonstrating the effectiveness of our method in reasonable cognition allocation and dynamic system switch.

4.3.3 Case Study

In Figure 4, we present an example comparing the reasoning behavior of DeepSeek-R1-Distill-Qwen-1.5B trained with GRPO and ACPO on a number theory problem from the Math 500 dataset. The task requires finding the smallest positive integer multiple of 30 that can be written using only the digits 0 and 2, which requires applying divisibility rules: the number must end in 0 and the sum of its digits must be divisible by 3. The model trained with GRPO adopts a cautious and exhaustive strategy, enumerating multiple candidate numbers (*e.g.,* 30, 60, 90) step by step, even after identifying a valid solution. This leads to a lengthy reasoning path with 1555 tokens, filled with redundant self-verification and repeated constraint checking. In contrast, the model trained with ACPO efficiently identifies the necessary constraints, rapidly filtering out invalid candidates and

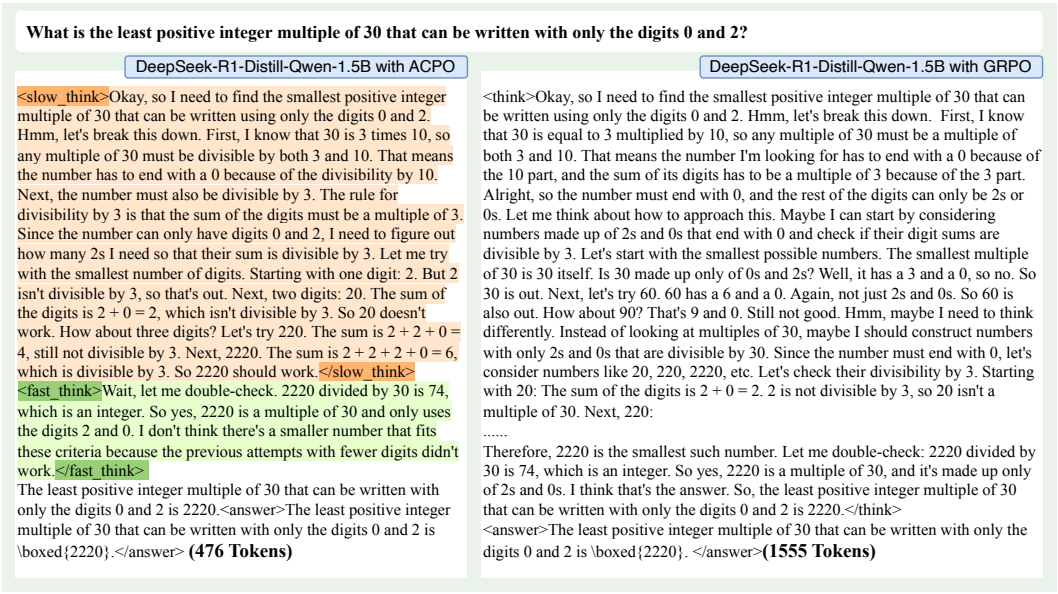

Figure 4: An case study comparing the reasoning process of DeepSeek-R1-Distill-Qwen-1.5B trained with GRPO and ACPO in MATH 500 Dataset.

quickly converging to the correct answer in just 476 tokens. Notably, the model exhibits effective system switch, employing slow thinking during problem analysis and fast thinking for verification once a solution is found, achieving substantial token savings especially in the verification stage by eliminating repetitive constraint checks. The system-aware reasoning tokens offer a transparent view of the model's cognitive process, enabling interpretable analysis of how reasoning behavior aligns with fast and slow thinking modes.

5 Related Work

Efficient Reasoning in LRMs. Large reasoning models (LRMs) like DeepSeek-R1 [2] and OpenAI o1 [4] have demonstrated strong reasoning capabilities, but their long reasoning paths often suffer from overthinking, introducing redundant content and reducing inference efficiency [10, 13, 11]. To address this, one line of work focuses on supervised fine-tuning with concise, high-quality data to reduce the length of reasoning paths [14, 15]. For instance, TokenSkip [14] generates short reasoning chains through token importance sampling, while Self-Training [15] selects correct short reasoning paths through sampling. Another line of work seeks to improve efficiency through reinforcement learning by modifying reward functions to penalize excessively long reasoning paths [3, 17]. Among these methods, length budgeting is a direct way to explicitly control reasoning length. For example, L1 [18] introduces a fixed length budget to penalize responses that exceed a predefined length during RL training. DAST [19] estimates task difficulty and token length budget through sampling, and constructs preference data for SimPO [32] training. In this work, we integrate online difficulty-aware length budgeting into reinforcement learning process, enabling real-time reasoning budget estimation and reward allocation for efficient reasoning.

Fast and Slow Thinking in LLMs. The dual process theory [20] describes two modes of human thinking: fast, intuitive thinking (System 1) and slow, deliberate thinking (System 2). Recent studies have investigated fast and slow thinking in LLMs, focusing on system switch based on task complexity or uncertainty. Specifically, System1.x [34] adapts a controller and System-1/2 planner to adjust reasoning systems based on task difficulty. FaST [35] develops a switching adapter that transitions between System 1 and System 2 for visual reasoning, depending on factors like uncertainty. HaluSearch [36] leverages model performance to generate supervised labels, enabling hierarchical dynamic switch between reasoning systems within MCTS. Dyna-Think[37] implements a training-free dynamic thinking mechanism, allowing the model to autonomously determine when to apply slow reasoning. In this work, we introduce system-aware reasoning tokens to explicitly

 https://doi.org/10.52202/085713-5108

represent fast and slow thinking modes, and leverage ACPO with online token length budget to enable adaptive cognitive allocation and dynamic system switch based on task difficulty.

6 Conclusion

In this paper, we proposed ACPO, a reinforcement learning framework designed to address the overthinking problem in LRMs through adaptive cognitive allocation and dynamic system switch. In our approach, we first introduce system-aware reasoning tokens to explicitly represent fast and slow thinking modes, making the model's cognitive process transparent and interpretable. Next, we proposed a two-stage training strategy, first fine-tuning the model to establish the ability to generate explicit thinking process, followed by reinforcement learning with ACPO to enhance adaptive cognition allocation. ACPO integrates online difficulty estimation and token length budget during training, guiding dynamic reasoning through a carefully designed reward function. Experimental results demonstrate that ACPO effectively reduces redundant reasoning without sacrificing too much accuracy. Our work provides a flexible and interpretable framework for adaptive hybrid reasoning, supporting efficient and difficulty-aware cognitive processes in LRMs.

7 Limitation

Although ACPO effectively reduces redundant reasoning and enables adaptive cognitive allocation, it has several limitations. First, its online difficulty estimation and token length budget mechanisms rely on verifiable data, which may limit generalization to open-domain tasks. Future work should explore more generalizable estimation methods for broader applicability. Second, ACPO can introduce accuracy trade-offs despite improving reasoning efficiency, highlighting the need for more refined reward designs to better balance efficiency and correctness.

Acknowledgments

This work was partially supported by National Natural Science Foundation of China under Grant No. 92470205 and 62222215, Beijing Natural Science Foundation under Grant No. L233008 and Beijing Municipal Science and Technology Project under Grant No. Z231100010323009. Xin Zhao is the corresponding author.

References

[1] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. A survey of large language models. *CoRR*, abs/2303.18223, 2023.

[2] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, and S. S. Li. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948, 2025.

[3] Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, Chuning Tang, Congcong Wang, Dehao Zhang, Enming Yuan, Enzhe Lu, Fengxiang Tang, Flood Sung, Guangda Wei, Guokun Lai, Haiqing Guo, Han Zhu, Hao Ding, Hao Hu, Hao Yang, Hao Zhang, Haotian Yao, Haotian Zhao, Haoyu Lu, Haoze Li, Haozhen Yu, Hongcheng Gao, Huabin Zheng, Huan Yuan, Jia Chen, Jianhang Guo, Jianlin Su, Jianzhou Wang, Jie Zhao, Jin Zhang, Jingyuan Liu, Junjie Yan, Junyan Wu, Lidong Shi, Ling Ye, Longhui Yu, Mengnan Dong, Neo Zhang, Ningchen Ma, Qiwei Pan, Qucheng Gong, Shaowei Liu, Shengling Ma, Shupeng Wei, Sihan Cao, Siying Huang, Tao Jiang, Weihao Gao, Weimin Xiong, Weiran He, Weixiao Huang, Wenhao Wu, Wenyang He, Xianghui Wei, Xianqing Jia, Xingzhe Wu, Xinran Xu, Xinxing Zu, Xinyu Zhou, Xuehai Pan, Y. Charles, Yang Li, Yangyang Hu, Yangyang Liu, Yanru Chen, Yejie Wang, Yibo Liu, Yidao Qin, Yifeng Liu, Ying Yang, Yiping Bao, Yulun Du, Yuxin Wu, Yuzhi Wang, Zaida Zhou, Zhaoji Wang, Zhaowei Li, Zhen Zhu, Zheng Zhang, Zhexu Wang, Zhilin Yang, Zhiqi Huang, Zihao Huang, Ziyao Xu, and Zonghan Yang. Kimi k1.5: Scaling reinforcement learning with llms. *CoRR*, abs/2501.12599, 2025.

[4] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

[5] Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025.

[6] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

[7] 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 *Findings of the Association for Computational Linguistics ACL 2024*, pages 1830–1842, 2024.

[8] 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.

[9] Mengdi Liu, Xiaoxue Cheng, Zhangyang Gao, Hong Chang, Cheng Tan, Shiguang Shan, and Xilin Chen. Protinvtree: Deliberate protein inverse folding with reward-guided tree search. *arXiv preprint arXiv:2506.00925*, 2025.

[10] 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. Do NOT think that much for 2+3=? on the overthinking of o1-like llms. *CoRR*, abs/2412.21187, 2024.

[11] Zhiyuan Zeng, Qinyuan Cheng, Zhangyue Yin, Bo Wang, Shimin Li, Yunhua Zhou, Qipeng Guo, Xuanjing Huang, and Xipeng Qiu. Scaling of search and learning: A roadmap to reproduce o1 from reinforcement learning perspective. *arXiv preprint arXiv:2412.14135*, 2024.

[12] Xiaoye Qu, Yafu Li, Zhaochen Su, Weigao Sun, Jianhao Yan, Dongrui Liu, Ganqu Cui, Daizong Liu, Shuxian Liang, Junxian He, Peng Li, Wei Wei, Jing Shao, Chaochao Lu, Yue Zhang, Xian-Sheng Hua, Bowen Zhou, and Yu Cheng. A survey of efficient reasoning for large reasoning models: Language, multimodality, and beyond. *CoRR*, abs/2503.21614, 2025.

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

[14] Heming Xia, Yongqi Li, Chak Tou Leong, Wenjie Wang, and Wenjie Li. Tokenskip: Controllable chain-of-thought compression in llms. *arXiv preprint arXiv:2502.12067*, 2025.

[15] Tergel Munkhbat, Namgyu Ho, Seo Hyun Kim, Yongjin Yang, Yujin Kim, and Se-Young Yun. Self-training elicits concise reasoning in large language models. *arXiv preprint arXiv:2502.20122*, 2025.

 https://doi.org/10.52202/085713-5108

[16] 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.

[17] Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. Thinkprune: Pruning long chain-of-thought of llms via reinforcement learning. *arXiv preprint arXiv:2504.01296*, 2025.

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

[19] Yi Shen, Jian Zhang, Jieyun Huang, Shuming Shi, Wenjing Zhang, Jiangze Yan, Ning Wang, Kai Wang, and Shiguo Lian. Dast: Difficulty-adaptive slow-thinking for large reasoning models. *arXiv preprint arXiv:2503.04472*, 2025.

[20] Daniel Kahneman. Thinking, fast and slow. *Farrar, Straus and Giroux*, 2011.

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

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

[23] Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. Limo: Less is more for reasoning. *arXiv preprint arXiv:2502.03387*, 2025.

[24] Wenkai Yang, Shuming Ma, Yankai Lin, and Furu Wei. Towards thinking-optimal scaling of test-time compute for llm reasoning. *arXiv preprint arXiv:2502.18080*, 2025.

[25] OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023.

[26] Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. `https://pretty-radio-b75.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-Scaling-RL-19681902c1468005bed8ca303` 2025. Notion Blog.

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

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

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

[30] 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.

[31] Xinyin Ma, Guangnian Wan, Runpeng Yu, Gongfan Fang, and Xinchao Wang. Cot-valve: Length-compressible chain-of-thought tuning. *arXiv preprint arXiv:2502.09601*, 2025.

[32] Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024.

[33] Bytedance Seed MLSys. verl: Volcano engine reinforcement learning for llms. `https://github.com/volcengine/verl`, 2025.

[34] Swarnadeep Saha, Archiki Prasad, Justin Chih-Yao Chen, Peter Hase, Elias Stengel-Eskin, and Mohit Bansal. System-1. x: Learning to balance fast and slow planning with language models. *arXiv preprint arXiv:2407.14414*, 2024.

[35] Guangyan Sun, Mingyu Jin, Zhenting Wang, Cheng-Long Wang, Siqi Ma, Qifan Wang, Tong Geng, Ying Nian Wu, Yongfeng Zhang, and Dongfang Liu. Visual agents as fast and slow thinkers. *arXiv preprint arXiv:2408.08862*, 2024.

[36] Xiaoxue Cheng, Junyi Li, Wayne Xin Zhao, and Ji-Rong Wen. Think more, hallucinate less: Mitigating hallucinations via dual process of fast and slow thinking. *arXiv preprint arXiv:2501.01306*, 2025.

[37] Jiabao Pan, Yan Zhang, Chen Zhang, Zuozhu Liu, Hongwei Wang, and Haizhou Li. Dynathink: Fast or slow? a dynamic decision-making framework for large language models. *arXiv preprint arXiv:2407.01009*, 2024.

[38] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report. *CoRR*, abs/2412.15115, 2024.

 https://doi.org/10.52202/085713-5108

A Prompts for Explicit Thinking Annotation

In Section 3.1.1, we prompt DeepSeek-R1-Distill-Qwen-32B to generate responses of varying reasoning lengths for candidate response sampling, and leverage GPT-4 for fine-grained comparison to annotate different thinking modes. The system prompts for these two components are provided below, where the system prompts for candidate sampling are adapted from TOPS [24].

System Prompts for Candidate Response Sampling

Low Reasoning Effort: You have extremely limited time to think and respond to the user's query. Every additional second of processing and reasoning incurs a significant resource cost, which could affect efficiency and effectiveness. Your task is to prioritize speed without sacrificing essential clarity or accuracy. Provide the most direct and concise answer possible. Avoid unnecessary steps, reflections, verification, or refinements UNLESS ABSOLUTELY NECESSARY. Your primary goal is to deliver a quick, clear and correct response.

High Reasoning Effort: You have unlimited time to think and respond to the user's question. There is no need to worry about reasoning time or associated costs. Your only goal is to arrive at a reliable, correct final answer. Feel free to explore the problem from multiple angles, and try various methods in your reasoning. This includes reflecting on reasoning by trying different approaches, verifying steps from different aspects, and rethinking your conclusions as needed. You are encouraged to take the time to analyze the problem thoroughly, reflect on your reasoning promptly and test all possible solutions. Only after a deep, comprehensive thought process should you provide the final answer, ensuring it is correct and well-supported by your reasoning.

System Prompts for Response Comparison and Annotation

Given a problem, a short answer, and a long answer, compare the short answer with the long answer and annotate the short answer based on the following rules. If the short answer omits certain reasoning or calculation steps that are present in the long answer, these omitted steps are considered fast thinking, and the corresponding parts in the short answer should be enclosed within <fast_think></fast_think>. If the short answer contains the same reasoning or calculation steps as the long answer, these parts are considered slow thinking and should be enclosed within <slow_think></slow_think>. Fast thinking parts typically involve intuitive judgments, skipped steps, or direct conclusions, whereas slow thinking parts involve full reasoning or calculations that align with those in the long answer. The output should be the short answer with the appropriate <fast_think></fast_think> and <slow_think></slow_think> tags added.

#Problem#:
#Long Answer#:
#Short Answer#:
#Annotated Answe#:

B Experimental Setting

B.1 Training and Evaluation Details

In the SFT stage, the learning rate is 1×10^{-5}, the batch size is 8, the number of epochs is 3. For evaluation, we use Qwen2.5 [38] tokenizer to calculate the number of tokens in the responses generated by each model for a fair comparison. All the experiments are conducted on 16 NVIDIA A100 GPUs.

B.2 Baseline Details

We present descriptions of the three baseline methods *SFT_Shortest*, *SimPO_Shortest*, and *SimPO_DAST* from DAST in Table 1. We adopt their evaluation results as reported in the original DAST paper [19].

- **SFT_Shortest**: Supervised fine-tuning using only the shortest correct sampled response of each problem as training data.
- **SimPO_Shortest**: SimPO with contrastive instance pairs, which take the shortest correct sampled response of each problem as positive instance and the longest as negative instance.
- **SimPO_DAST**: SimPO with contrastive instance pairs from D_{pre} constructed in DAST.

B.3 Prompt for ACPO Training

We present the prompts used in the ACPO training process below.

Prompt for ACPO Training

You are a helpful AI Assistant that provides well-reasoned and detailed responses. You first think about the reasoning process as an internal monologue and then provide the user with the answer. Respond in the following format: <think>...</think><answer>...</answer>. In the reasoning process, you think fast in simple steps and slowly in complex steps, and put the slow thinking content in <slow_think></slow_think> and fast thinking content in <fast_think></fast_think>. Let's think step by step and output the final answer within boxed{}.

C License for existing assets

We list the licenses for the datasets and models used in our experiments.

C.1 Models

- DeepSeek-R1-Distill-Qwen-1.5B: Apache 2.0 License.
- DeepSeek-R1-Distill-Qwen-7B: Apache 2.0 License.
- DeepSeek-R1-Distill-Llama-8B: Llama3.1 License.

C.2 Datasets

- The GSM8K [29] dataset: MIT License.
- The MATH [30] dataset: MIT License.
- The DeepScaleR-Preview-Dataset [26]: MIT License.

https://doi.org/10.52202/085713-5108

NeurIPS Paper Checklist

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 made sure our abstraction and introduction accurately reflect paper's main contribution and scope.

   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: [Yes]

   Justification: We include a standalone limitation section in the Appendix 7. We have discussed both the methodological limitations and performance bottlenecks of our approach.

   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: [NA]

Justification: Our paper does not include theoretical results.

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 the experiment details and results in Section 4 and Appendix B. We will also release our code upon publication.

   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**

https://doi.org/10.52202/085713-5108

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: Yes, we provide all the necessary code to reproduce every experiment mentioned in the paper. We will also release our code publicly upon publication.

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: Yes, we disclose all the training and test details in Section 4 and Appendix B.

   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: Our evaluation experiment is conducted by sampling 16 responses and computing the pass@1 score to demonstrate the performance.

   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 the computer resources in Appendix B.

   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: Yes, the paper conform, in every respect, with the NeurIPS Code of Ethics.

   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: Our method is a training technique for large language models which does not have negative societal impact that needs to be addressed.

    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.

https://doi.org/10.52202/085713-5108

- 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: This paper poses no such risks.

    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: We list licenses for used datasets and models in Appendix C.

    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: [Yes]

    Justification: The released code, data, and models are well documented including details about training, license, limitations.

    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: The 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: The 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.
    - 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.

 https://doi.org/10.52202/085713-5108

- 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: [Yes]

    Justification: We describe the usage of LLMs in experiments in Section 4.

    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.