Adaptive Dual Reasoner: Large Reasoning Models Can Think Efficiently by Hybrid Reasoning

Yujian Zhang[1,*]
Tencent Youtu Lab
23s003081@stu.hit.edu.cn

Keyu Chen[1]
Tencent Youtu Lab
yolochen@tencent.com

Zhifeng Shen
Tencent Youtu Lab
billsshen@tencent.com

Ruizhi Qiao
Tencent Youtu Lab
ruizhiqiao@tencent.com

Xing Sun
Tencent Youtu Lab
winfredsun@tencent.com

Abstract

Although Long Reasoning Models (LRMs) have achieved superior performance on various reasoning scenarios, they often suffer from increased computational costs and inference latency caused by overthinking. To address these limitations, we propose an **A**daptive **D**ual **R**easoner, which supports two reasoning modes: fast thinking and slow thinking. **ADR** dynamically alternates between these modes based on the contextual complexity during reasoning. **ADR** is trained in two stages: (1) A cold-start stage using supervised fine-tuning (SFT) to equip the model with the ability to integrate both fast and slow reasoning modes, in which we construct a hybrid reasoning dataset through a dedicated pipeline to provide large-scale supervision. (2) A reinforcement learning stage for optimizing reasoning effort, where we introduce **E**ntropy-guided **H**ybrid **P**olicy **O**ptimization (**EHPO**), an RL training framework employing an entropy-guided dynamic rollout strategy for branching at high-entropy units and a difficulty-aware penalty to balance fast and slow reasoning. Across challenging mathematical reasoning benchmarks, **ADR** achieves an effective balance between reasoning performance and efficiency among state-of-the-art approaches. Specifically, **ADR** yields a performance gain of up to 6.1%, while reducing the reasoning output length by 49.5% to 59.3%.

1 Introduction

With the recent emergence of Long Reasoning Models (LRMs) [1, 2, 3], the Chain-of-Thought (CoT) [4] reasoning has been further popularized as the mainstream paradigm for tackling complex tasks such as mathematical or logical problems. However, LRMs are notorious for over-thinking [5, 6], wherein the model unnecessarily generates redundant reasoning. Recently, a surge of research has focused on addressing the overthinking problem in LRMs. One of the simplest approaches [7, 8, 9] is prompt engineering, which aims to make the model's output steps more concise through specific prompts. Other methods, such as probability manipulation [10], token budget [11, 9, 12, 13], early exiting [14, 15, 16], and CoT compression [17, 18], focus on avoiding frequent shifts in thought or shorten the output length. However, these length-driven approaches may lead to insufficient exploration of complex reasoning steps that require deeper thinking. To further refine the control of reasoning length in LRMs for different problems, a variety of reinforcement learning methods have been proposed [19, 20, 21, 22, 23] to perform various length or difficulty-based rewards. To control reasoning behaviors, a range of approaches [24, 25, 26, 27, 28, 29, 30] has introduced the

[1]Equal contribution.
[*]Work done during internship at Tencent.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Efficient Reasoning.

concept of hybrid reasoning modes (*e.g.*, fast thinking, and slow thinking). Some hybrid reasoning methods utilize a router to select appropriate models [31] or reasoning modes [32, 33] according to the estimated difficulty of the query. Subsequent works have sought to eliminate the dependency on routers by adopting reinforcement learning frameworks, enabling autonomously routing to select the appropriate reasoning mode [28, 29, 30]. Nevertheless, actual reasoning trajectories often comprise sub-problems of varying complexity, and coarse-grained control over reasoning modes is unable to adaptively allocate cognitive resources for each reasoning path. To tackle this issue, recent approaches [34, 35, 36] decompose reasoning steps into smaller units for fine-grained control, but they rely on static rollout strategies that restrict deeper exploration on hard subproblems.To overcome this limitation, we propose the **A**daptive **D**ual **R**easoner, which dynamically switches between fast and slow reasoning modes according to contextual complexity. **ADR** is trained in two stages: first, a supervised fine-tuning stage equips the model with both reasoning modes; second, we introduce **E**ntropy-guided **H**ybrid **P**olicy **O**ptimization (**EHPO**), a reinforcement learning framework that leverages entropy trends for dynamic rollout strategy and a difficulty-aware penalty to balance efficiency and accuracy. Our contributions are summarized as follows:

- **ADR**, a novel reasoning paradigm, which enables large reasoning models to adaptively switch between fast reasoning for straightforward cases and slow reasoning for complex dependencies, laying the foundation for flexible allocation of reasoning effort.
- An automated hybrid reasoning data construction curator. We build a scalable pipeline that automatically constructs hybrid reasoning data, enabling existing LRMs to transition smoothly into the new hybrid reasoning paradigm.
- **EHPO**, a reinforcement learning framework that integrates entropy trends for dynamic rollout strategy and a difficulty-aware penalty to balance efficiency and accuracy, thereby optimizing reasoning under the hybrid paradigm.

2 Methodology

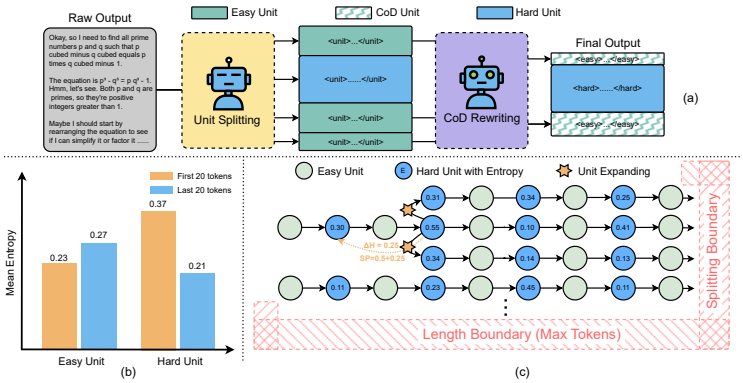

Figure 1: (a) Hybrid Reasoning Data Construction. (b) Entropy Analysis of Two Reasoning Units: Transitions from easy to hard mode exhibit higher entropy. (c) Entropy-Guided Dynamic Rollout Strategy: Branching occurs with probability $\text{SP} = \alpha + \Delta H$ when transitioning from easy mode to hard mode, where ΔH denotes the normalized entropy difference.

2.1 Aligning the Model to the Adaptive Dual Reasoning Paradigm

To align the model with the adaptive dual reasoning paradigm that supports both fast and slow reasoning modes, we conduct cold-start training via supervised fine-tuning (SFT) and construct hybrid reasoning data from an open-source reasoning dataset. Inspired by the observation that higher entropy is associated with keywords related to reflection, verification, and exploration[37, 38], we propose a data construction process based on CoT decomposition and rewriting, as shown in Figure 1(a). Specifically, we decompose reasoning trajectories into reasoning units, labeling those with high-entropy content as hard and others as easy. Easy units are compressed using CoD-style to

minimize token usage, while hard units remain uncompressed to retain reasoning depth. These units are then annotated with special tokens to form the final reasoning format as following:

$$\text{<think> <easy> } u_1 \text{ </easy> <hard> } u_2 \text{ </hard> } \cdots \text{ <easy> } u_n \text{ </easy> </think> } a \tag{1}$$

2.2 Entropy-Guided Hybrid Policy Optimization

To further improve reasoning efficiency while preserving accuracy, we propose **EHPO**, a reinforcement learning framework that updates the model using a GRPO-based objective. EHPO combines mode control reward with entropy-guided dynamic rollout to suppress unnecessary deep reasoning while retaining essential hard units across problems of varying difficulty.

2.2.1 Reward Design

We design a reward function with four signals to jointly optimize the allocation of reasoning effort:

$$R = \mathcal{R}_{\text{format}} * \mathcal{R}_{\text{accuracy}} * \mathcal{R}_{\text{unit}} * \mathcal{R}_{\text{mode}} \tag{2}$$

While the first two rewards enforce structural compliance and correctness, we highlight the latter two below.

Unit semantic Reward To encourage the model to distinguish between two modes of reasoning rather than collapsing into the original paradigm, we define a unit semantic reward based on keyword matching. Each reasoning unit u_i is semantically correct only if (i) u_i is easy and contains no reflection/verification keywords such as "Wait", "However" and "Alternatively", or (ii) u_i is hard and contains at least one such keyword. Then, the overall unit semantic reward is defined as follows:

$$\mathcal{R}_{\text{unit}} = \begin{cases} 1, & \text{if all units are semantic correct,} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

Mode Control Reward To optimize the model's utilization of reasoning effort, we introduce a difficulty-aware mode control reward, which encourages the preferential use of the easy mode on lower-difficulty tasks while promoting deeper reasoning in the hard mode for challenging ones:

$$\mathcal{R}_{\text{mode}} = \beta + (1-\beta) \cdot \left(\frac{N_{pass}}{N} \cdot p_{\text{easy}} + (1 - \frac{N_{pass}}{N}) \cdot p_{\text{hard}} \right), \tag{4}$$

where N and N_{pass} denote total and correct samples, p_{easy} and p_{hard} are the token ratios of the easy mode and hard mode, respectively, and β is a hyperparameter controlling the reward scale within the range $[\beta, 1]$. We set $\beta = 0.7$ by default.

2.2.2 Entropy-Guided Dynamic Rollout Strategy

In pilot training experiments, we found that $\mathcal{R}_{\text{mode}}$, which discourages deep reasoning, compresses the exploration space and undermines response accuracy. To analyze the model's exploration behavior, we measure the entropy at the beginning and end of each reasoning unit. As demonstrated in Figure 1 (b), the terminal entropy values of easy units are generally higher than their initial entropy values, whereas hard units exhibit the opposite trend, which indicates that transitions from easy to hard mode exhibit higher entropy, consistent with the requirement for deeper exploration.

Based on this observation, we propose an **E**ntropy-guided **D**ynamic **R**ollout (**EDR**) strategy: when transitioning from easy to hard mode, model generates multiple branches to expand its exploration space, compensating for reduced reasoning depth by increasing reasoning breadth.Specifically, we record the entropy of the first k tokens as the initial entropy H_0 when generating the first hard unit. Upon transitioning from an easy to a hard unit, the model branches with probability $\alpha + \Delta H$, where $\alpha = 0.5$ is the base probability and ΔH is the normalized entropy difference, as illustrated in 1(c).

Table 1: Performance comparison of various baselines and our method. The **bold** and underlined values denote the best and second-best results, respectively. Accuracy-Efficiency Score (AES), introduced by O1-Pruner [44], measures efficiency by rewarding shorter outputs without compromising accuracy. The Avg AES is computed over benchmarks with reported results, excluding tasks where outcomes are unavailable. The accuracy (Acc.) is measured by the pass@1, which is estimated as the average correctness over 16 sampled generations.

	AIME2025			AIME2024			MATH500			
	Acc.	**Tokens**	**AES**	**Acc.**	**Tokens**	**AES**	**Acc.**	**Tokens**	**AES**	**Avg. AES**
Baseline	23.5	12119	—	30.4	12290	—	81.7	4802	—	—
O1-Pruner	23.2	$8731_{\downarrow 28.0\%}$	0.22	—	—	—	**84.3**	$2913_{\downarrow 39.3\%}$	0.49	0.35
DRP	—	—	—	33.3	$6135_{\downarrow 50.1\%}$	0.79	82.0	$2122_{\downarrow 55.8\%}$	0.57	<u>0.68</u>
ES	**24.2**	$7458_{\downarrow 38.5\%}$	**0.47**	28.3	$9024_{\downarrow 26.6\%}$	-0.08	<u>83.0</u>	$2400_{\downarrow 50.0\%}$	0.55	0.31
ACPO	—	—	—	30.0	$6670_{\downarrow 45.7\%}$	0.39	81.0	$\mathbf{1679}_{\downarrow 65.0\%}$	**0.61**	0.50
ADR w/o EDR	21.5	$\mathbf{5890}_{\downarrow 51.4\%}$	0.09	<u>33.8</u>	$\mathbf{5971}_{\downarrow 51.4\%}$	<u>0.85</u>	81.6	$1992_{\downarrow 58.5\%}$	<u>0.58</u>	0.51
ADR	<u>23.3</u>	$6126_{\downarrow 49.5\%}$	<u>0.45</u>	**36.5**	$6110_{\downarrow 50.3\%}$	**1.10**	81.0	$1955_{\downarrow 59.3\%}$	0.55	**0.70**

3 Experiments

3.1 Experimental Setup

We build the cold-start dataset with 300k examples sampled from the OpenMathReasoning [39] dataset, then conduct our EHPO training on the DeepScaleR-Preview [40] dataset and adopt a two-stage training procedure with max response length limits of 8k and 16k tokens following DeepScaleR. Note that the 8k stage is intended to rapidly strengthen the model's foundational capabilities, and thus the entropy-guided dynamic rollout strategy is applied exclusively in the 16k stage. For evaluation, we use four mathematical reasoning benchmarks: AIME25 [41], AIME24 [42], and MATH500 [43].

We use DeepSeek-R1-Distill-Qwen-1.5B as the base model and compare against the following baselines: (1) O1-Pruner [44], a fine-tuning method that uses pre-sampling and RL-style optimization to reduce reasoning length while preserving accuracy in LRMs; (2) DRP [45], a distillation–pruning framework that reduces token usage via teacher-guided step pruning; (3) Efficiency Steering (ES) [46], leveraging large models' intrinsic potential to produce concise reasoning while preserving accuracy;(4) ACPO [36] also trains models to switch reasoning modes; but in contrast to our approach, it adopts standard GRPO with a customized reward function, without enforcing a strict distinction between the two modes during RL training.

3.2 Results

Balancing Reasoning Efficiency and Accuracy As shown in Table 1, our method achieves strong performance across datasets. On challenging tasks, it attains the highest accuracy on AIME2024 (36.5%, 6.1% higher than baseline) with 50.3% shorter outputs, yielding the best efficiency score of 1.10, and maintains competitive accuracy on AIME2025 with 49.5% fewer tokens with AES of 0.45. On MATH500, it preserves accuracy while reducing token usage by nearly 60% with AES of 0.55. Overall, our approach achieves the best average AES of 0.70. Note that as DRP and ACPO did not provide results on the most challenging AIME2025, their average AES is likely upward-biased, whereas ADR still outperforms the strongest baseline DRP (0.68).

Ablation of Entropy-Guided Dynamic Rollout Strategy Unoptimized RL training (ADR w/o EDR) brings only limited benefits, reaching 33.8% accuracy on AIME2024 with an average AES of just 0.51. In contrast, adding EDR significantly improves both accuracy and efficiency: on AIME2024, accuracy rises to 36.5% (2.7% higher than ADR w/o EDR) with the highest AES of 1.10, and similar efficiency gains are observed across tasks. Overall, EDR boosts the Avg. AES from 0.51 to 0.70, confirming that EDR enables more effective accuracy–efficiency trade-offs.

3.3 Conclusions

In this work, we present **ADR**, a new reasoning paradigm for large reasoning models, enabling model dynamically switches between fast reasoning for straightforward cases and intensive reasoning for complex dependencies. To optimize reasoning allocation, we introduce **EHPO**, which combines mode control reward with entropy-guided dynamic rollout to expand the exploration space while maintaining accuracy. Extensive experiments on multiple mathematical reasoning benchmarks show that our approach achieves an effective balance between reasoning efficiency and accuracy, demonstrating robust performance across tasks of varying difficulty.

References

[1] 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.

[2] ByteDance Seed, Jiaze Chen, Tiantian Fan, Xin Liu, Lingjun Liu, Zhiqi Lin, Mingxuan Wang, Chengyi Wang, Xiangpeng Wei, Wenyuan Xu, et al. Seed1. 5-thinking: Advancing superb reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.13914*, 2025.

[3] Qwen Team. Qwen3 technical report, 2025.

[4] 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.

[5] Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Hanjie Chen, et al. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*, 2025.

[6] Jinyan Su, Jennifer Healey, Preslav Nakov, and Claire Cardie. Between underthinking and overthinking: An empirical study of reasoning length and correctness in llms. *arXiv preprint arXiv:2505.00127*, 2025.

[7] Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. Chain of draft: Thinking faster by writing less. *arXiv preprint arXiv:2502.18600*, 2025.

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

[9] 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.

[10] Yue Wang, Qiuzhi Liu, Jiahao Xu, Tian Liang, Xingyu Chen, Zhiwei He, Linfeng Song, Dian Yu, Juntao Li, Zhuosheng Zhang, et al. Thoughts are all over the place: On the underthinking of o1-like llms. *arXiv preprint arXiv:2501.18585*, 2025.

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

[12] Zheng Li, Qingxiu Dong, Jingyuan Ma, Di Zhang, and Zhifang Sui. Selfbudgeter: Adaptive token allocation for efficient llm reasoning. *arXiv preprint arXiv:2505.11274*, 2025.

[13] 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.

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

[15] Yichao Fu, Junda Chen, Siqi Zhu, Zheyu Fu, Zhongdongming Dai, Aurick Qiao, and Hao Zhang. Efficiently serving llm reasoning programs with certaindex. *arXiv e-prints*, pages arXiv–2412, 2024.

[16] Haoran Zhao, Yuchen Yan, Yongliang Shen, Haolei Xu, Wenqi Zhang, Kaitao Song, Jian Shao, Weiming Lu, Jun Xiao, and Yueting Zhuang. Let llms break free from overthinking via self-braking tuning. *arXiv preprint arXiv:2505.14604*, 2025.

[17] Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. C3ot: Generating shorter chain-of-thought without compromising effectiveness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24312–24320, 2025.

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

[19] Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning, 2025. *URL https://arxiv. org/abs/2503.04697*, 2025.

[20] Jingyang Yi, Jiazheng Wang, and Sida Li. Shorterbetter: Guiding reasoning models to find optimal inference length for efficient reasoning. *arXiv preprint arXiv:2504.21370*, 2025.

[21] Razvan-Gabriel Dumitru, Darius Peteleaza, Vikas Yadav, and Liangming Pan. Conciserl: Conciseness-guided reinforcement learning for efficient reasoning models. *arXiv preprint arXiv:2505.17250*, 2025.

[22] Junjie Yang, Ke Lin, and Xing Yu. Think when you need: Self-adaptive chain-of-thought learning. *arXiv preprint arXiv:2504.03234*, 2025.

[23] Zhengxiang Cheng, Dongping Chen, Mingyang Fu, and Tianyi Zhou. Optimizing length compression in large reasoning models. *arXiv preprint arXiv:2506.14755*, 2025.

[24] Gongfan Fang, Xinyin Ma, and Xinchao Wang. Thinkless: Llm learns when to think. *arXiv preprint arXiv:2505.13379*, 2025.

[25] Siye Wu, Jian Xie, Yikai Zhang, Aili Chen, Kai Zhang, Yu Su, and Yanghua Xiao. Arm: Adaptive reasoning model. *arXiv preprint arXiv:2505.20258*, 2025.

[26] Songjun Tu, Jiahao Lin, Qichao Zhang, Xiangyu Tian, Linjing Li, Xiangyuan Lan, and Dongbin Zhao. Learning when to think: Shaping adaptive reasoning in r1-style models via multi-stage rl. *arXiv preprint arXiv:2505.10832*, 2025.

[27] Jiajie Zhang, Nianyi Lin, Lei Hou, Ling Feng, and Juanzi Li. Adaptthink: Reasoning models can learn when to think. *arXiv preprint arXiv:2505.13417*, 2025.

[28] Lingjie Jiang, Xun Wu, Shaohan Huang, Qingxiu Dong, Zewen Chi, Li Dong, Xingxing Zhang, Tengchao Lv, Lei Cui, and Furu Wei. Think only when you need with large hybrid-reasoning models. *arXiv preprint arXiv:2505.14631*, 2025.

[29] Xiaoyun Zhang, Jingqing Ruan, Xing Ma, Yawen Zhu, Haodong Zhao, Hao Li, Jiansong Chen, Ke Zeng, and Xunliang Cai. When to continue thinking: Adaptive thinking mode switching for efficient reasoning. *arXiv preprint arXiv:2505.15400*, 2025.

[30] Chenwei Lou, Zewei Sun, Xinnian Liang, Meng Qu, Wei Shen, Wenqi Wang, Yuntao Li, Qingping Yang, and Shuangzhi Wu. Adacot: Pareto-optimal adaptive chain-of-thought triggering via reinforcement learning. *arXiv preprint arXiv:2505.11896*, 2025.

[31] OpenAI. Introducing gpt-5, 2025.

[32] Simon A Aytes, Jinheon Baek, and Sung Ju Hwang. Sketch-of-thought: Efficient llm reasoning with adaptive cognitive-inspired sketching. *arXiv preprint arXiv:2503.05179*, 2025.

[33] Guosheng Liang, Longguang Zhong, Ziyi Yang, and Xiaojun Quan. Thinkswitcher: When to think hard, when to think fast. *arXiv preprint arXiv:2505.14183*, 2025.

[34] Zihao Zeng, Xuyao Huang, Boxiu Li, Hao Zhang, and Zhijie Deng. Done is better than perfect: Unlocking efficient reasoning by structured multi-turn decomposition. *arXiv preprint arXiv:2505.19788*, 2025.

[35] Sohyun An, Ruochen Wang, Tianyi Zhou, and Cho-Jui Hsieh. Don't think longer, think wisely: Optimizing thinking dynamics for large reasoning models. *arXiv preprint arXiv:2505.21765*, 2025.

[36] Xiaoxue Cheng, Junyi Li, Zhenduo Zhang, Xinyu Tang, Wayne Xin Zhao, Xinyu Kong, and Zhiqiang Zhang. Incentivizing dual process thinking for efficient large language model reasoning. *arXiv preprint arXiv:2505.16315*, 2025.

[37] Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, et al. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*, 2025.

[38] Zhengkai Lin, Zhihang Fu, Ze Chen, Chao Chen, Liang Xie, Wenxiao Wang, Deng Cai, Zheng Wang, and Jieping Ye. Controlling thinking speed in reasoning models. *arXiv preprint arXiv:2507.03704*, 2025.

[39] Ivan Moshkov, Darragh Hanley, Ivan Sorokin, Shubham Toshniwal, Christof Henkel, Benedikt Schifferer, Wei Du, and Igor Gitman. Aimo-2 winning solution: Building state-of-the-art mathematical reasoning models with openmathreasoning dataset. *arXiv preprint arXiv:2504.16891*, 2025.

[40] 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, 2025. Notion Blog.

[41] Mathematical Association of America. American Invitational Mathematics Examination 2025, 2025. Accessed: 2025-05-14.

[42] Mathematical Association of America. American invitational mathematics examination 2024, 2024. Accessed: 2025-05-14.

[43] 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.

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

[45] Yuxuan Jiang, Dawei Li, and Frank Ferraro. Drp: Distilled reasoning pruning with skill-aware step decomposition for efficient large reasoning models. *arXiv preprint arXiv:2505.13975*, 2025.

[46] Weixiang Zhao, Jiahe Guo, Yang Deng, Xingyu Sui, Yulin Hu, Yanyan Zhao, Wanxiang Che, Bing Qin, Tat-Seng Chua, and Ting Liu. Exploring and exploiting the inherent efficiency within large reasoning models for self-guided efficiency enhancement. *arXiv preprint arXiv:2506.15647*, 2025.

A Details of Cold-Start Data Construction

A.1 Datasets

We construct the cold-start dataset based on the open-source mathematical reasoning corpus [39], which comprises 306k math problems and 3.2 million long CoT responses generated by Deepseek-R1 and QwQ-32B. From this corpus, we extract and reformulate 300k examples for cold-start SFT training.

A.2 Prompts

Prompt for ADR Generating We provide the prompt we use for ADR-paradigm reasoning:

Prompt for ADR Generating

{QUESTION}
Let's think step by step and output the final answer within \boxed{}. But switch between two modes based on the difficulty of each thought process: In normal scenarios, complete each step with minimal tokens as quickly as possible. Wrap the thinking content in <easy></easy> tags.When encountering difficult steps requiring reflection, verification, or iterative exploration, prioritize correctness without token limitations. Wrap the thinking content in <hard> </hard> tags. Once resolved and subsequent steps no longer require deep thinking, revert to easy mode.

Figure 2: Prompt for ADR Generating

Prompt for CoD-Style Shortening We use Deepseek-R1-0528 to shorten the content of easy units in CoD-style [7]. As shown in 3, we provide a paired comparison example between a real CoD-style output and the original R1-style output as a reference.

Prompt for CoD-style Shortening

Given an LLM output, carefully compare the writing styles in the example texts. Refer to the concise style example to rewrite this output, but make sure no useful information may be omitted. Output only the fully revised text as plain text. Do not include explanations. Do not wrap the text in any form.

Example original version:

First, since he's making deposits at the end of each year, and interest is compounded annually, I need to calculate the future value of an annuity. An annuity is a series of equal payments made at regular intervals. Here, it's an ordinary annuity because payments are at the end of each period.
The future value FV of an ordinary annuity is given by:
F V=P \times \frac{(1+r)^{n}-1}{r}
Where:
P is the payment amount per period,
r is the interest rate per period,
n is the number of periods.

Example concise version:

First, since the deposits are made at the end of each year, this is an ordinary annuity problem. The future value of an ordinary annuity formula is:
FV = P * [(1 + r)^n - 1]/r
where P is the annual payment, r is the interest rate, and n is the number of periods.

Current original output:
{text}

Revised version:

Figure 3: Prompt for CoD-Style Shortening

B Case Study

As illustrated in Figure 4, we observe a striking contrast between the baseline model and our ADR-trained model. The baseline (Deepseek-R1-Distill-Qwen1.5B) produced an output of 4344 tokens, reflecting an almost exclusive reliance on the deep reasoning. Its reasoning process was verbose and exploratory: the model repeatedly attempted factorization, re-derived discriminants, and engaged in extensive verification, even when the solution path had already been established. While correct, this exhaustive style incurred substantial inefficiency.

By contrast, the ADR-trained model (ADR 1.5B) completed the same task in only 1900 tokens. The improvement stems from ADR's ability to dynamically switch between fast and slow thinking. Most steps were carried out in the easy mode—directly computing discriminants, deriving closed-form solutions, and quickly mapping integer constraints, exhibiting higher information density. Compared with the baseline, ADR reaches the same intermediate conclusions using fewer tokens. The model switched into hard mode only at crucial junctures, such as verifying overlaps between solution families in this case. This selective use of deeper reasoning preserved correctness while avoiding unnecessary elaboration.

Overall, ADR does not merely compress outputs; rather, it strategically allocates slow thinking only where needed, yielding concise yet rigorous reasoning. This case study highlights ADR's effectiveness in substantially improving inference efficiency without sacrificing accuracy.

Figure 4: A case study comparing the reasoning process of DeepSeek-R1-Distill-Qwen-1.5B and ADR in AIME2025.