Think Better, Not Longer: Token-Level Marginal Utility for Efficient Reasoning in Large Reasoning Models

Jiawei Li, Yang Gao*, Huashan Sun, Chong Feng
School of Computer Science and Technology,
Beijing Institute of Technology, Beijing, China
Beijing Engineering Research Center of High Volume Language Information
Processing and Cloud Computing Applications, Beijing, China
{jwli,gyang,hssun,fengchong}@bit.edu.cn

Abstract

While Large Reasoning Models (LRMs) have demonstrated remarkable capabilities through explicit Chain-of-Thought (CoT) generation, they frequently suffer from "overthinking". In this work, we bridge this gap by introducing **Token-level Marginal Utility**, which quantifies the per-token log-probability gain of the ground-truth answer. Leveraging this dense supervision signal, we propose **MUTO** (**M**arginal **U**tility Guided **T**hinking **O**ptimization), a unified training framework designed to synthesize concise reasoning chains. Rather than relying only on coarse trajectory-level length control, MUTO identifies tokens that reduce the model's likelihood of the correct answer and penalizes such negative-utility reasoning, yielding concise yet effective CoT trajectories. Experiments on DeepSeek-R1-Distill-Qwen backbones (1.5B and 7B) across six math reasoning benchmarks show that MUTO yields a markedly better efficiency-accuracy Pareto frontier. It reduces average token usage by 87.1% at 1.5B while improving accuracy by 2.3%, and cuts tokens by 80.2% at 7B with only -0.1% accuracy change, achieving the best length-normalized accuracy among baselines.

1 Introduction

Large reasoning models (LRMs) (OpenAI, 2024; Team, 2025; DeepSeek-AI, 2025; Chen et al., 2025) achieve strong performance by generating explicit Chain-of-Thought (CoT) rationales. However, a growing body of work has revealed a pervasive phenomenon of "overthinking" in these models: LRMs frequently generate unnecessarily long inference trajectories, often extending to thousands of tokens even for simple problems (Chen et al., 2024; Zhao et al., 2025; Li et al., 2025a; Sui et al., 2025). This behavior increases latency and cost, and often exhibits diminishing returns in accuracy, while introducing redundant steps that distract from the core solution path. Mitigating such redundancy is therefore crucial for efficient deployment of LRMs.

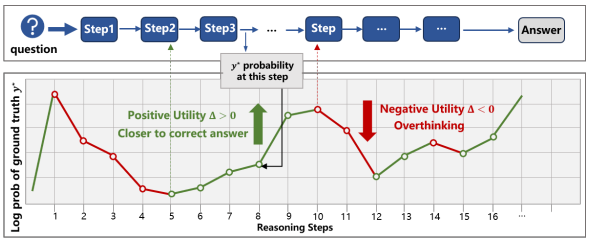

Figure 1: Conceptual framework for quantifying token-level marginal utility (Δ_t) in LRMs. The top panel displays a generated CoT $z_{1:T}$ responding to a problem input x. At each time step t, we compute the ground-truth answer log-likelihood under the model conditioned on the current prefix $z_{\leq t}$ (via a fixed probing format), without altering the prefix itself. The bottom panel illustrates the resulting probability trajectory.

Recent studies further indicate that the contribution of individual tokens to the final answer is highly non-uniform. For instance, Liu et al. (2025b) demonstrated a correlation between token probability distributions and task performance, but their analysis operates at a coarse-grained, sequence-level resolution. Wang et al. (2025a) observed that CoT sequences comprise a mix of low-entropy anchor tokens and high-entropy exploration tokens, each playing distinct roles in the reasoning process. Ding et al. (2025) provides evidence that some intermediate tokens can help reasoning whereas others can be detrimental. However, the lack of a fine-grained metric to quantify the specific impact of individual tokens in these works leads us to a critical question: *"Can we quantify the marginal utility of each token to the generation of the correct answer, thereby identifying and pruning tokens with negative utility to mitigate overthinking?"*

In this work, we introduce **token-level marginal utility** to quantify each reasoning token's contribution to the ground-truth answer. As illustrated in Figure 1, given an input x, a generated rationale

* Corresponding author.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 30052–30063
July 2-7, 2026

prefix $z_{\leq t}$, and the ground-truth answer $y^\star$, we define $\Delta_t = \log p(y^\star \mid x, z_{\leq t}) - \log p(y^\star \mid x, z_{\leq t-1})$, estimated by probing the model to answer at each prefix. Tokens with $\Delta_t > 0$ are positively useful, while $\Delta_t < 0$ indicates negative utility with respect to ground-truth answer log-probability.

Building upon this metric, we propose **MUTO** (**M**arginal **U**tility Guided **T**hinking **O**ptimization) to orchestrate adaptive thinking modes and synthesize concise reasoning chains. MUTO uses token-level marginal utility as dense supervision to distinguish useful reasoning from redundant or harmful generation. Concretely, MUTO augments trajectory-level optimization with two utility-driven signals: a marginal-utility penalty that discourages reasoning trajectories that lower the model's likelihood of the correct answer, and a token-level redundancy penalty that suppresses tokens with non-positive marginal utility. In our experiments, we instantiate MUTO on top of the dual-mode adaptive reasoning scaffold of AutoThink (Tu et al., 2025b). This scaffold provides a controlled setting for comparing direct-answer and CoT trajectories, whereas MUTO contributes a complementary token-level credit assignment signal that measures and optimizes the utility of individual reasoning tokens.

We validate MUTO on six math reasoning benchmarks with DS-R1-Distill-Qwen backbones (1.5B/7B). MUTO reduces average generation length by 87.1% at 1.5B while improving accuracy by 2.3%, and cuts tokens by 80.2% at 7B with only a -0.1% accuracy change. Under the length-normalized accuracy metric (L-Acc) (Liu et al., 2025a), MUTO achieves 47.5 (1.5B) and 57.6 (7B), outperforming existing efficiency-oriented baselines. Moreover, MUTO consistently outperforms recent LRMs and state-of-the-art adaptive reasoning methods in both specialized reasoning benchmarks and general capability evaluations. Our main contributions are as follows:

- We propose *token-level marginal utility*, a dense supervision signal that quantifies each reasoning token's marginal log-probability gain toward the ground-truth answer, enabling fine-grained diagnosis of overthinking without requiring labeled CoTs.
- We introduce **MUTO**, a reinforcement learning with verifiable rewards(RLVR) based utility-guided training framework that penalizes reasoning trajectories and individual tokens when they reduce the model's likelihood of the ground-truth answer, enabling LRMs to produce concise yet effective CoT trajectories.
- Extensive evaluations across model scales (1.5B/7B) and six benchmarks show that MUTO substantially improves the efficiency-accuracy trade-off, often achieving Pareto-optimal among evaluated baselines.

2 Related Works

Efficient Reasoning for LRMs LRMs often suffer from overthinking (Zeng et al., 2025c; Sui et al., 2025), generating long and redundant CoT traces even on easy instances, which increases latency and can induce reasoning drift. Existing efforts primarily improve efficiency by reducing output length through (i) training-time length control, e.g., reinforcement learning with length-based rewards or constraints (Xiang et al., 2025; Arora and Zanette, 2025; Tu et al., 2025a), and preference-based fine-tuning using short–long pairs obtained via best-of-N sampling or postprocessing (Luo et al., 2025a; Wang et al., 2025b); and (ii) inference-time control, including prompting strategies (Kang et al., 2025; Han et al., 2025), early stopping (Li et al., 2024; Sun et al., 2025), sample pruning (Hou et al., 2025; Zeng et al., 2025b), and related heuristics that limit superfluous computation. Despite progress, most methods rely on coarse signals (e.g., length) or external control, offering limited token-level guidance for suppressing harmful steps (Liu et al., 2025b; Liang et al., 2025). We instead optimize a token-level marginal utility signal, enabling MUTO to learn adaptive think/no-think routing and to reduce negative-utility tokens during reasoning.

Token-Efficient Reasoning and Budget Control To mitigate the computational overhead of LRMs, recent research has explored explicit mechanisms for token reduction and budget control (Li et al., 2025b; Yuan et al., 2025). Initial attempts focused on inference-time constraints, such as TALE (Han et al., 2025) and TOPS (Yang et al., 2025), which introduce token budgets via prompting or dynamic adjustments to limit generation. However, these methods often struggle to achieve precise control without compromising reasoning accuracy, as models lack intrinsic guidance on how to allocate the budget effectively (Lee et al., 2025). More advanced approaches integrate efficiency directly into training. Methods like Thinkless (Fang et al., 2025) and AdaCoT (Lou et al., 2025) learn to adaptively

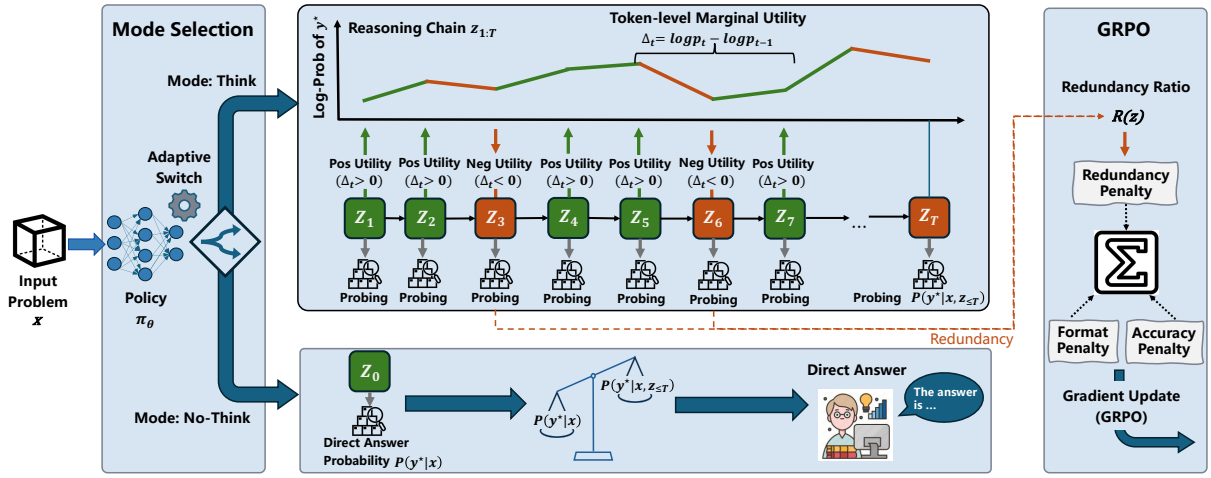

Figure 2: The overall training framework of *MUTO*. During the training phase, the model first determines the optimal inference mode (Think vs. No-Think) by comparing the probability of generating the ground truth via a direct answer against full reasoning. In the Think mode, we probe the probability of the ground truth after each reasoning token is generated to calculate the marginal utility of that token. Finally, the proportion of tokens with negative utility is quantified as the redundancy rate, which serves as a penalty signal for GRPO optimization.

route between extensive reasoning and direct answering. Nevertheless, these strategies predominantly rely on sequence-level signals, thereby treating the reasoning chain as a monolithic block. They fail to distinguish the specific utility of intermediate steps. In contrast, our work shifts the paradigm to a token-level perspective, leveraging marginal utility to quantify and optimize the contribution of each individual token towards the final answer.

3 Token-level Marginal Utility for CoT

Given an input problem x and an LLM policy π_θ, the model produces a reasoning trajectory

$$z_{1:T} = (z_1, z_2, \ldots, z_T), \tag{1}$$

which may include intermediate CoT tokens and a final answer segment (e.g., a span starting with "`Final answer is` $\cdot$"). We assume access to the ground-truth answer $y^\star$ during training.

To quantify how each intermediate token impacts the model's belief in the correct answer, we define the *token-level final-answer probability*. After generating prefix $z_{\le t} = (z_1, \ldots, z_t)$, we measure the probability that the model would immediately output the ground-truth answer $y^\star$:

$$p_t^{\text{final}} := P_\theta(y^\star \mid x, z_{\le t}). \tag{2}$$

In practice, p_t^{final} can be implemented as the conditional likelihood of $y^\star$ under the model when we force it to answer at step t:

$$p_t^{\text{final}} = \prod_{k=1}^{|y^\star|} P_\theta\big(y_k^\star \mid x, z_{\le t}, y_{<k}^\star\big). \tag{3}$$

We then define the *token-level marginal utility* of the t-th token z_t with respect to the ground-truth answer in log-space for numerical stability as

$$\Delta_t := \log p_t^{\text{final}} - \log p_{t-1}^{\text{final}}. \tag{4}$$

A token is said to have *positive marginal utility* if $\Delta_t > 0$, meaning that it moves the model closer to the ground-truth answer in terms of probability; conversely, $\Delta_t \le 0$ indicates that it is non-helpful or even harmful to predicting $y^\star$.

4 Utility-Driven Reasoning Optimization

Building on token-level marginal utility, we propose **MUTO**, a utility-guided training framework for reducing redundant reasoning. The central idea is to convert marginal-utility estimates into reward signals at two levels. At the trajectory level, MUTO penalizes reasoning traces that decrease the model's likelihood of the ground-truth answer compared with a direct-answer context. At the token level, MUTO penalizes trajectories containing a high fraction of non-positive-utility tokens. We instantiate this objective on top of a dual-mode adaptive reasoning scaffold (Tu et al., 2025b), which provides both direct-answer and CoT trajectories during training.

4.1 Training Scaffold and Base Reward

MUTO requires trajectories generated under different reasoning budgets so that utility-based rewards can compare when explicit reasoning helps or hurts. We therefore instantiate MUTO using the dual-mode scaffold of AutoThink (Tu et al., 2025b), where each rollout is generated either in a direct-answer mode or in an explicit-CoT mode.

Base reward from the scaffold. Following AutoThink (Tu et al., 2025b), we use a batch-balanced base reward to train a policy to choose between a THINK mode (with explicit reasoning) and a NO-THINK mode (direct answer). Concretely, each rollout i is assigned a mode label $\text{mode}_i \in \{\text{THINK}, \text{NO-THINK}\}$ and a correctness indicator $\text{correct}_i \in \{0, 1\}$ for the final answer. We use their batch-balanced reward

$$r_i^{\text{base}} = f_{\text{BBR}}(\text{mode}_i, \text{correct}_i; \mathcal{B}), \tag{5}$$

where $f_{\text{BBR}}(\cdot)$ denotes the batch-level reward balancing function defined in Tu et al. (2025b), and $\mathcal{B}$ is the current mini-batch. Intuitively, r_i^{base} gives higher reward to correct NO-THINK answers to promote efficiency, while softly down-weighting whichever mode is overrepresented in the batch to avoid mode collapse. Detailed formulations and hyperparameter settings for this backbone reward are provided in Appendix A. We focus on how to augment it with token-level marginal utility.

Marginal-utility-based reward augmentation. While r_i^{base} captures final correctness and the chosen reasoning mode at the trajectory level, it does not explicitly compare the model's belief in the correct answer *before* and *after* thinking. Our key idea is to introduce a marginal-utility-based *reward augmentation term* that evaluates whether thinking actually *helps* or *hurts* the final-answer probability, using the token-level notion from Section 3.

For a THINK trajectory, let c_0 denote the context before any reasoning tokens are generated (i.e., the prompt that would be used in NO-THINK mode). We define

$$p_0^{\text{final}} := P_\theta(y^\star \mid c_0), \tag{6}$$

the probability that the model would immediately produce the ground-truth answer without explicit thinking. Let c_T denote the full context after the entire reasoning trajectory $z_{1:T}$ (including the final answer segment). We then measure

$$p_T^{\text{final}} := P_\theta(y^\star \mid c_T), \tag{7}$$

the probability assigned to the correct answer after the model has finished thinking.

We are specifically interested in cases where *thinking makes the correct answer less likely*, i.e., $p_T^{\text{final}} < p_0^{\text{final}}$. To obtain a numerically stable signal, we operate in log-space and define

$$\tilde{p}_0 = \log p_0^{\text{final}}, \quad \tilde{p}_T = \log p_T^{\text{final}}. \tag{8}$$

For THINK trajectories, we introduce a marginal-utility penalty term

$$r_{\text{marginal}} = -\lambda \cdot \max(0, \tilde{p}_0 - \tilde{p}_T), \tag{9}$$

where $\lambda > 0$ controls the strength of the penalty. Intuitively, when thinking increases or at least preserves the log-probability of the correct answer ($\tilde{p}_T \geq \tilde{p}_0$), no penalty is applied; when thinking actually *decreases* the model's belief in $y^\star$ ($\tilde{p}_T < \tilde{p}_0$), the reward is reduced proportionally to the loss in log-probability.

The final reward used for RLVR is then

$$r_i^{\text{mode}} = \begin{cases} r_i^{\text{base}}, & \text{mode}_i = \text{NO-THINK}, \\ r_i^{\text{base}} + r_{\text{marginal}}, & \text{mode}_i = \text{THINK}. \end{cases} \tag{10}$$

We optimize the policy π_θ with RLVR using r_i^{mode} as the scalar feedback. In contrast to prior work that only conditions the reward on trajectory-level correctness and the chosen mode, this objective explicitly incorporates marginal utility into the signal, discouraging reasoning that empirically *harm* the answer probability compared to answering directly.

4.2 Token-level Utility Shaping

While the scaffold-level reward can regulate reasoning at the trajectory level, CoT generated in the THINK mode may still contain many intermediate tokens that are redundant. A trajectory-level signal alone cannot reveal which specific steps improve the model's belief in the correct answer and which ones induce reasoning drift. We therefore introduce token-level utility shaping, which aggregates marginal utility estimates into a reward signal that directly penalizes non-helpful reasoning tokens.

Auxiliary length prior for THINK trajectories. Prior work has proposed adjusting rewards based on the length of the reasoning trace in order to implicitly control the amount of thinking. In our setting, we restrict such length-aware shaping to trajectories generated in THINK mode, and view it as a coarse prior over how much computation should be spent.

For a trajectory i in THINK mode, let T_i denote the number of reasoning tokens (excluding the final answer segment), and let $\text{correct}_i \in \{0, 1\}$ indicate whether the final answer matches $y^\star$. We define a generic length-shaping term

$$r_i^{\text{len}} = g(\text{correct}_i, T_i; \alpha, \beta), \tag{11}$$

where $g(\cdot)$ is a monotonically increasing function of T_i when $\text{correct}_i = 1$ (penalizing overly short correct reasoning), and a monotonically decreasing function of T_i when $\text{correct}_i = 0$ (discouraging long unsuccessful reasoning). The hyperparameters $\alpha, \beta > 0$ control the strength of this dependence. A concrete instantiation of $g(\cdot)$ and its ablation are provided in Appendix B. This length-aware term serves as a coarse constraint that prefers spending more computation on successful trajectories while discouraging unproductive overthinking on failures.

Token-level redundancy penalty. Purely length-based shaping does not distinguish between *useful* and *redundant* reasoning tokens: two trajectories with the same length may have very different proportions of harmful steps. To capture this difference, we aggregate the token-level marginal utilities $\{\Delta_t\}$ into a trajectory-level redundancy ratio.

For a generated trajectory $z_{1:T}$, we define its redundancy ratio as

$$R(z_{1:T}) = \frac{1}{T} \sum_{t=1}^{T} \mathbf{1}\big[\Delta_t \le 0\big], \tag{12}$$

i.e., the fraction of tokens that do not increase the model's belief in the ground-truth answer. Intuitively, $R(z_{1:T})$ characterizes how many steps in a reasoning chain are redundant or misdirected with respect to the final answer.

We then use $R(z_{1:T_i})$ to define a redundancy penalty for trajectory i:

$$r_i^{\text{red}} = -\gamma \cdot R(z_{1:T_i}), \tag{13}$$

where $\gamma > 0$ controls how aggressively we penalize redundant steps. Unlike the length term, which only depends on the total number of reasoning tokens T_i, this penalty is sensitive to *which* tokens are actually helpful according to their marginal utility.

Final utility-shaped reward. For trajectories in THINK mode, we combine the coarse length prior and the fine-grained redundancy penalty into a single token-level shaping term

$$r_i^{\text{tok}} = r_i^{\text{len}} + r_i^{\text{red}}. \tag{14}$$

The overall reward for trajectory i then becomes

$$r_i^{\text{final}} = \begin{cases} r_i^{\text{mode}}, & \text{if mode}_i = \text{NO-THINK}, \\ r_i^{\text{mode}} + r_i^{\text{tok}}, & \text{if mode}_i = \text{THINK}, \end{cases} \tag{15}$$

where r_i^{mode} is the mode-level reward from Section 4.1 that already incorporates the marginal-utility-based penalty on harmful thinking at the trajectory level. Compared with prior reward designs that primarily regulate reasoning length at the trajectory level, MUTO introduces utility information directly into the training signal by penalizing trajectories with a high fraction of non-positive-utility tokens.

5 Experiments

5.1 Experimental Setup

Training Datasets and Models We adopt the same training corpus as DeepScaleR (Luo et al., 2025b), consisting of 40K mathematically oriented problems spanning a wide range of difficulty levels. Following standard practice in recent work on reasoning-centric LLMs (Tu et al., 2025b; Zeng et al., 2025a; Tu et al., 2025a), we evaluate on five widely used math benchmarks: MATH, Minerva, Olympiad, AIME24, and AMC23. To examine the robustness of our approach across scales, we instantiate our method on two backbone models, DeepSeek-R1-Distill-Qwen-1.5B and 7B (DeepSeek-AI, 2025), enabling a controlled study of how utility-driven reasoning optimization behaves under different capacity regimes.

Baselines Besides backbone models, we compare our approach with models trained using alternative algorithms that may enable adaptive thinking capabilities, including Concise-RL (Fatemi et al., 2025), ShorterBetter (Yi and Wang, 2025), ThinkPrune (Hou et al., 2025), and AutoThink (Tu et al., 2025a).

Training and Evaluation Exact computation of token-level marginal utility for every token is computationally expensive during training. We therefore use a sparse approximation: marginal utility is evaluated only once every 20 tokens, and its sign is propagated to neighboring tokens within a local window of size 20 centered at the sampled position. All experiments are implemented using the VerL (Sheng et al., 2025) framework [27], with most training hyperparameters retained at the default values. The training context size, batch size,

and the learning rate are set to 16K, 128, and 2e-6, respectively. During evaluation, all models use a 32K context window. We sample 8 rollouts per instance with temperature 0.6 and report the average pass@1 accuracy.

Evaluation Metrics. Beyond standard accuracy, evaluating the trade-off between performance and inference cost is crucial for efficient reasoning models. To strictly quantify this balance, we adopt the **Length-normalized Accuracy (L-Acc)** metric proposed by Liu et al. (2025a). L-Acc rewards models that maintain high accuracy while reducing token consumption. The metric is defined as:

$$\text{L-Acc} = \text{Acc} \times \sqrt{1 - \frac{L}{L_{\text{base}}}}, \qquad (16)$$

where Acc represents the model's accuracy, L denotes the average number of tokens generated by the evaluated method, and L_{base} is the token usage of the original backbone model. A higher L-Acc score indicates a superior accuracy and efficiency trade-off (Lou et al., 2025) in the reasoning.

5.2 Main Results

Alongside our baselines, we include several state-of-the-art reasoning models, including GPT-4o, o1-preview, o4-mini-high and DeepSeek-R1 (DeepSeek-AI, 2025), along with several DeepSeek-R1-Distill-Qwen (DS-R1-Distill) models ranging from 1.5B to 32B. We report our results in Table 1, and we have the following findings:

The "Overthinking" Phenomenon in SOTA Models. State-of-the-art reasoning models achieve strong accuracy but often at substantial token cost. For instance, compared with QwQ-32B, DeepSeek-R1 gains only $+2.9\%$ accuracy while using roughly $+13\%$ more tokens. This mismatch suggests diminishing returns in decoding: LRMs tend to overgenerate long traces with limited additional accuracy, a pattern we observe across backbones and efficiency baselines on six reasoning benchmarks.

MUTO Achieves Optimal Accuracy and Efficiency Trade-off. Table 1 presents a comprehensive comparison of MUTO against the DS-R1-Distill backbones and various baselines across six mathematical reasoning benchmarks.

In the 1.5B parameter regime, MUTO defies the conventional trade-off between efficiency and performance. Remarkably, MUTO not only reduces the average token usage by $\mathbf{87.1}\%$ (from 10,633 to 1,374 tokens) but also improves the average accuracy by $\mathbf{2.3}\%$ compared to the base DS-R1-Distill-1.5B model. Notably, on the challenging AIME24 benchmark, MUTO achieves a substantial gain of $+5.8\%$ (from 27.5% to 33.3%). This suggests that for smaller models, the original CoT often contains low-quality or distracted reasoning paths; MUTO effectively prunes these redundancies, resulting in a more concise and accurate reasoning process. The accuracy and efficiency trade-off is rigorously quantified by the **L-ACC** metric, where MUTO achieves a score of **47.5**, significantly outperforming baselines such as *Concise-RL* (40.9) and *AutoThink* (37.3). Figure 3 visualizes the trade-off landscape for 1.5B models. MUTO consistently occupies a position on the accuracy-efficiency trade-off compared to other methods.

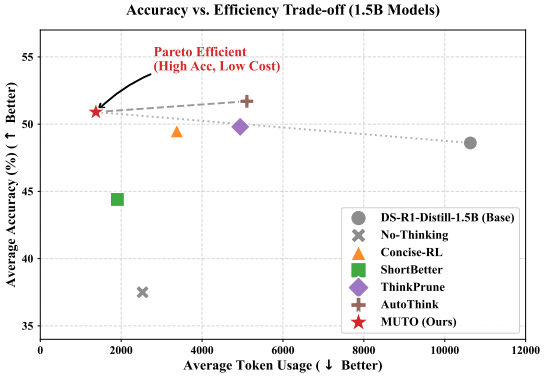

Figure 3: **Accuracy vs. Efficiency Trade-off on 1.5B models.** The x-axis represents average token usage (log scale recommended), and the y-axis represents average accuracy across 6 benchmarks. MUTO (red star) achieves the optimal balance, situated in the top-left region, strictly dominating baselines.

Scalability of Redundancy Reduction. MUTO demonstrates robust scalability across varying model sizes. In the 7B parameter regime, it sustains massive computational efficiency without degrading predictive capability. Specifically, MUTO curtails the inference budget by **80.2%** (7,815 to 1,544 tokens) with a minimal accuracy deviation of **-0.1%** (64.3% vs. 64.4%) compared to the standard backbone. Despite the marginal accuracy fluctuation, MUTO demonstrates a decisive advantage in the efficiency-accuracy trade-off, recording an **L-ACC of 57.6**. This score strictly dominates adaptive methods like *AutoThink* (L-ACC 41.3), which consumes nearly $3\times$ more tokens (4,635) to achieve a comparable accuracy level. These results confirm that MUTO effectively generalizes its token-level marginal utility to larger models, facilitating highly efficient inference with negligible quality loss.

Methods	Accuracy (↑)						Token Usage (↓)						Eff-Metric
	MATH	Minerva	Olym.	AIME	AMC	Avg.	MATH	Minerva	Olym.	AIME	AMC	Avg.	L-Acc (↑)
DS-R1-Distill-14B	92.1	37.6	58.1	62.3	82.4	66.5	3774	5434	8362	12064	7317	7390	/
DS-R1-Distill-32B	93.4	37.7	62.1	65.4	85.7	68.9	4947	8290	10096	12905	8432	8934	/
QwQ-32B	95.1	45.3	69.0	76.7	95.5	76.3	5547	8650	9445	13970	4222	8367	/
DeepSeek-R1	96.8	51.2	72.4	79.3	96.2	79.2	6102	9210	11500	15420	5100	9466	/
GPT-4o	92.5	44.1	65.3	68.0	91.5	72.3	1250	1800	2400	3500	1600	2110	/
o1-preview	97.1	53.5	75.8	81.2	97.4	81.0	8500	11200	14500	21000	9200	12880	/
o4-mini-high	94.5	48.2	68.5	74.1	94.0	75.9	4200	6500	8100	11500	4800	7020	/
DS-R1-Distill-1.5B*	83.1	26.0	43.7	27.5	62.5	48.6	5622	7688	11555	17322	10981	10633	/
+No-Thinking*	70.4	19.1	33.1	15.8	49.0	37.5	1256	628	2426	5793	2535	2528	32.7
+Concise-RL*	81.0	28.1	43.6	30.0	64.9	49.5	1963	2126	3683	5806	3300	3376	40.9
+ShortBetter*	79.4	27.6	38.4	20.0	56.6	44.4	1927	1147	1814	2703	1946	1907	40.2
+ThinkPrune*	83.5	28.4	43.4	28.3	65.4	49.8	2723	3375	5504	8072	5040	4943	36.4
+AutoThink*	84.0	28.1	44.8	34.6	67.0	51.7	2195	3212	5559	9514	5095	5108	37.3
+MUTO (ours)	82.4	27.6	44.6	33.3	66.4	50.9	**1017**	**1257**	**1506**	**1593**	**1496**	**1374**	**47.5**
Δ vs. Base	-0.7	+1.6	+0.9	+5.8	+3.9	+2.3	-81.9%	-83.6%	-86.9%	-90.8%	-86.4%	-87.1%	-
DS-R1-Distill-7B*	92.3	37.6	56.4	52.7	82.8	64.4	3928	5155	8815	13563	7613	7815	/
+No-Thinking*	78.2	22.1	40.2	22.7	53.7	43.4	722	486	1434	3269	1433	1496	39.0
+Concise-RL*	90.3	/	/	51.7	/	/	2041	/	/	6632	/	/	/
+ShortBetter*	/	44.1	50.7	53.3	75.9	/	/	**1341**	3410	5288	2580	/	/
+AutoThink*	91.2	38.2	56.4	54.8	83.3	64.8	2146	2838	5498	8051	4645	4635	41.3
+MUTO (ours)	90.5	37.5	57.1	55.0	82.6	64.3	**1154**	1425	**1682**	**1854**	**1620**	**1544**	**57.6**
Δ vs. Base	-1.8	-0.1	+0.7	+2.3	-0.2	-0.1	-70.7%	-72.4%	-80.9%	-86.3%	-78.7%	-80.2%	-

Table 1: Performance comparison of MUTO against various baselines. We report Accuracy (%), Average Token Usage, and the Efficiency-Accuracy metric **L-Acc** (Liu et al., 2025a). Δ denotes the improvement of MUTO over the respective DS-R1-Distill backbone. Note that L-Acc is calculated relative to the backbone model's token usage. Results marked with * are taken directly from Tu et al. (2025b). Best results in the 1.5B and 7B groups are bolded.

Methods	Accuracy (↑)			Token Usage (↓)		
	GPQA	TheoremQA	Avg.	GPQA	TheoremQA	Avg.
DS-R1-Distill-1.5B	33.8	33.1	33.45	8842	7979	8411
+MUTO (ours)	34.4	33.6	34.00	1982	2169	2076
Δ vs. Base	+0.6	+0.5	+0.55	-77.58%	-72.82%	-75.32%
DS-R1-Distill-7B	34.3	34.4	34.35	7980	7128	7554
+MUTO (ours)	34.6	35.8	35.20	2159	2461	2310
Δ vs. Base	+0.3	+1.4	+0.85	-72.94%	-65.47%	-69.21%

Table 2: Generality beyond mathematical reasoning on GPQA and TheoremQA. MUTO consistently reduces token usage while maintaining or improving accuracy on scientific and multi-disciplinary question answering.

6 Analysis

In this section, we investigate why MUTO improves reasoning efficiency by addressing three questions. First, does MUTO generalize beyond the mathematical reasoning setting and remain effective across different domains and model families? Second, does MUTO improve token-level marginal utility, shifting generation away from redundant or harmful steps? Third, in failure cases, are errors primarily due to a capability deficit or reasoning drift driven by accumulated negative-utility tokens?

6.1 Robustness Across Domains and Model Families

To examine whether the benefit of MUTO extends beyond the mathematical DeepSeek setting, we evaluate it along two complementary axes: (i) non-mathematical tasks with verifiable answers, (ii) a math-specialized but relatively concise backbone.

Beyond mathematical reasoning. Although MUTO is motivated by overthinking in mathematical reasoning, its utility signal only requires a verifiable target answer and is therefore not inherently tied to math. We evaluate MUTO on GPQA (Rein et al., 2024), which emphasizes scientific multi-hop reasoning, and TheoremQA (Chen et al., 2023), which covers broader multi-disciplinary question answering, using DS-R1-Distill-1.5B and 7B backbones. As shown in Table 2, MUTO consistently improves or preserves accuracy while substantially reducing generation length. On DS-R1-Distill-1.5B, MUTO increases average accuracy from 33.45 to 34.00 while reducing average token usage from 8,411 to 2,076 (-75.32%). On DS-R1-Distill-7B, it improves average accuracy from 34.35 to 35.20 and lowers average token usage from 7,554 to 2,310 (-69.21%). These results suggest that the utility signal is not specific to symbolic mathematics; it also helps suppress redundant reasoning in scientific and multi-disciplinary QA.

Math-specialized but concise backbones. We next study a setting where overthinking is less pronounced by applying MUTO to Qwen2.5-Math-1.5B-Instruct, a math-specialized model whose baseline responses are already short. As shown in Table 3, MUTO still improves average accuracy

Methods	MATH	Minerva	Olym.	AIME	AMC	Avg.
Qwen2.5	56.3 / 235	29.4 / 424	31.3 / 793	10.0 / 538	40.0 / 592	33.4 / 516.4
+MUTO (ours)	57.4 / 222	32.0 / 412	32.6 / 612	10.0 / 477	45.0 / 526	35.4 / 449.8
Δ vs. Base	+1.1 / -5.5%	+2.6 / -2.8%	+1.3 / -22.8%	0.0 / -11.3%	+5.0 / -11.1%	+2.0 / -12.9%

Table 3: Generalization to a math-specialized model, Qwen2.5-Math-1.5B-Instruct. Each cell reports Accuracy (%) / Average Token Usage. MUTO remains beneficial even when the backbone already produces relatively concise reasoning traces.

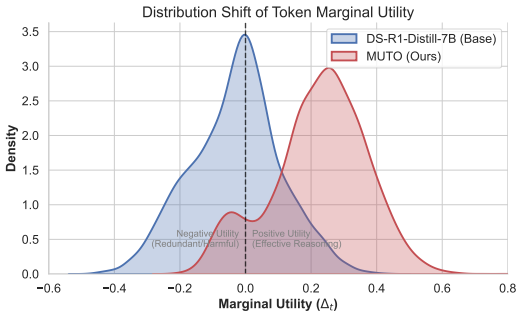

Figure 4: Distribution of Token-level Marginal Utility (Δ_t). We compare the density of utility values on the AIME24 dataset. The shaded red region highlights the significant reduction in negative-utility tokens achieved by MUTO compared to the Base model. MUTO shifts the distribution mass towards positive utility, indicating more efficient information accumulation.

from 33.4 to 35.4 (+2.0) while reducing average token usage from 516.4 to 449.8 (-12.9%). The efficiency gain is noticeably smaller than in our main DeepSeek-R1-Distill experiments, which is itself informative: when a backbone already produces concise standard CoT rather than long exploratory traces, there is less redundant computation for MUTO to remove.

6.2 Evolution of Token Utility

A core motivation of our method is to maximize the log-probability gain of each generated token. Recall from Section 3 that the token-level marginal utility is defined as $\Delta_t = \log p_t^{\text{final}} - \log p_{t-1}^{\text{final}}$, quantifying how much token z_t contributes to the correct prediction of the ground truth $y^\star$. To investigate the underlying mechanism of MUTO's efficiency gains, we conducted a comparative analysis of the token utility distribution between the backbone model (DS-R1-Distill-7B) and our MUTO-tuned counterpart on the MATH benchmark.

We hypothesize that the "overthinking" phenomenon is characterized by a high frequency of tokens with negative marginal utility ($\Delta_t < 0$), representing reasoning steps that confuse the model or drift away from the correct solution. Figure 4 illustrates the kernel density estimation (KDE) of marginal utility values for both models and the detailed formulations and bandwidth selection strategies for this estimation are provided in Appendix C. The backbone model (blue curve) exhibits a heavy tail in the negative region and a significant probability mass centered around zero, indicating a large proportion of tokens that are either detrimental or redundant (i.e., "stalling" without log-probability gain). In stark contrast, MUTO (red curve) significantly suppresses the negative tail. The distribution shifts towards the positive domain, suggesting that MUTO effectively prunes reasoning paths that do not contribute to the ground-truth answer.

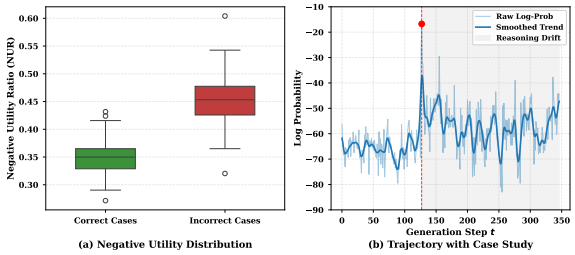

Figure 5: Failure analysis with token-level marginal utility on MATH. (a) NUR is higher for incorrect than correct trajectories. (b) The model briefly assigns high log-probability to $y^\star$, but continued negative-utility tokens reduce confidence and lead to an incorrect answer.

6.3 Failure Analysis: Capability Deficit or Reasoning Drift?

A critical question arises regarding the failure cases of CoT: Does the LRMs fail because it cannot inherently solve the problem (*capability deficit*), or because the extended reasoning chain introduces noise that distracts the LRMs from the correct path (*reasoning drift*)?

Using token-level marginal utility, we analyze failure modes of DS-R1-Distill-Qwen-1.5B on MATH. We split test instances into *Correct* vs. *Incorrect* according to the final answer, and examine negative-utility tokens ($\Delta_t < 0$) within the generated rationale tokens.

Observation 1: Incorrect trajectories contain more negative-utility tokens. We define the **Negative Utility Ratio (NUR)** as the fraction of rationale tokens with $\Delta_t < 0$ in a trajectory. As shown in Figure 5(a), incorrect trajectories exhibit a higher NUR on average (45.8%) than correct ones (34.9%), indicating a systematic shift toward denser detrimental generation in failures.

Observation 2: All failures transiently reach high log-probability in the ground truth but drift afterwards. To probe whether the model ever "locks onto" the correct answer during generation, we compute the (length-normalized) log-probability of the ground-truth answer string $y^\star$ under intermediate prefixes. Strikingly, in all incorrect trajectories, the model assigns substantial probability mass to the ground-truth answer, even though the final output is wrong. Figure 5(b) illustrates a typical example: the model reaches high confidence in $y^\star$ at an intermediate position, but continues generating additional tokens with predominantly negative marginal utility. As these negative-utility tokens accumulate, the model's confidence in $y^\star$ decreases, and the final prediction becomes incorrect. These findings are more consistent with reasoning drift than with a pure capability deficit for a substantial portion of failures.

7 Conclusion

In this paper, we addressed the critical challenge of "overthinking" in LRMs, where excessive reasoning steps often lead to computational inefficiency and reasoning drift. We introduced a novel metric, Token-level Marginal Utility, to rigorously quantify the incremental contribution of intermediate tokens to the final prediction. Based on this metric, we proposed MUTO, a unified training framework that leverages dense supervision signals to dynamically orchestrate thinking modes and prune redundant tokens. Our extensive experiments on the DeepSeek-R1-Distill-Qwen series demonstrate that the MUTO establishes a superior efficiency-performance trade-off. Furthermore, our in-depth analysis reveals that "overthinking" is often characterized by the accumulation of negative-utility tokens which dilute the model's confidence; the MUTO effectively mitigates this by enforcing a concise reasoning path.

Limitations

While MUTO demonstrates promising results, there are several avenues for further improvement.

First, marginal utility is computed using ground-truth labels, which makes it most suitable for training and offline analysis, and limits direct use at inference time for open-ended queries. An important direction is to train a lightweight "utility reward model" that predicts token utility from the prefix, so that similar pruning or early stopping can be applied at test time without labels.

Second, we primarily validate MUTO on mathematical and scientific reasoning tasks, where correctness is well defined. Extending token-level utility optimization to creative writing or long-context summarization remains challenging, because "utility" is more subjective and may involve multiple criteria beyond answer correctness. We hope our work encourages further research into efficient, transparent, and adaptive reasoning paradigms.

Ethical Considerations

This work studies efficiency and reliability issues in LRMs by introducing token-level marginal utility and using it as a dense supervision signal to reduce redundant reasoning and mitigate reasoning drift. Our method does not introduce new data sources or deployment settings beyond standard LRM training and evaluation, and we do not anticipate risks beyond those typical of language model research.

A primary risk is that encouraging shorter reasoning could remove useful intermediate steps in some cases, potentially leading to incorrect answers that appear confident. We mitigate this risk by defining utility directly with respect to the ground-truth log-probability, and by jointly training a think or no-think routing policy so that additional computation is allocated when it is beneficial. We also evaluate on quantitative reasoning benchmarks with verifiable solutions, where failures are easier to detect than in open-ended generation.

As with any approach built on pretrained language models, MUTO inherits potential biases and limitations from the underlying backbone and training data. However, since our experiments focus on mathematical and scientific reasoning tasks with objective correctness criteria, the likelihood of generating harmful or biased content is reduced relative to open-ended domains.

Acknowledgements

We sincerely thank the anonymous reviewers for their helpful feedback and the conference committee for their hard work. This work has received partial funding from the Major Research Plan of the National Natural Science Foundation of China (Grant No.92370110) and the National Natural Science Foundation of China General Program (Grant No.62576038).

References

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

Jiaze Chen, Tiantian Fan, Xin Liu, Lingjun Liu, Zhiqi Lin, Mingxuan Wang, Chengyi Wang, Xiangpeng Wei, Wenyuan Xu, Yufeng Yuan, Yu Yue, Lin Yan, Qiying Yu, Xiaochen Zuo, Chi Zhang, Ruofei Zhu, Zhecheng An, Zhihao Bai, Yu Bao, and 80 others. 2025. Seed1.5-thinking: Advancing superb reasoning models with reinforcement learning. *CoRR*, abs/2504.13914.

Wenhu Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony Xia. 2023. Theoremqa: A theorem-driven question answering dataset. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7889–7901.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2024. Do NOT think that much for 2+3=? on the overthinking of o1-like llms. *CoRR*, abs/2412.21187.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Bowen Ding, Yuhan Chen, Futing Wang, Lingfeng Ming, and Tao Lin. 2025. Do thinking tokens help or trap? towards more efficient large reasoning model. *CoRR*, abs/2506.23840.

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

Mehdi Fatemi, Banafsheh Rafiee, Mingjie Tang, and Kartik Talamadupula. 2025. Concise reasoning via reinforcement learning. *CoRR*, abs/2504.05185.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. 2025. Token-budget-aware llm reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 24842–24855.

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

Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. 2025. C3ot: Generating shorter chain-of-thought without compromising effectiveness. In *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 24312–24320. AAAI Press.

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

Jiawei Li, Yang Gao, Yizhe Yang, Yu Bai, Xiaofeng Zhou, Yinghao Li, Huashan Sun, Yuhang Liu, Xingpeng Si, Yuhao Ye, and 1 others. 2025a. Fundamental capabilities and applications of large language models: A survey. *ACM Computing Surveys*, 58(2):1–42.

Yiwei Li, Peiwen Yuan, Shaoxiong Feng, Boyuan Pan, Xinglin Wang, Bin Sun, Heda Wang, and Kan Li. 2024. Escape sky-high cost: Early-stopping self-consistency for multi-step reasoning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Zheng Li, Qingxiu Dong, Jingyuan Ma, Di Zhang, and Zhifang Sui. 2025b. Selfbudgeter: Adaptive token allocation for efficient LLM reasoning. *CoRR*, abs/2505.11274.

Guosheng Liang, Longguang Zhong, Ziyi Yang, and Xiaojun Quan. 2025. Thinkswitcher: When to think hard, when to think fast. *CoRR*, abs/2505.14183.

Hanbing Liu, Lang Cao, Yuanyi Ren, Mengyu Zhou, Haoyu Dong, Xiaojun Ma, Shi Han, and Dongmei Zhang. 2025a. Bingo: Boosting efficient reasoning of llms via dynamic and significance-based reinforcement learning. *CoRR*, abs/2506.08125.

Peijie Liu, Fengli Xu, and Yong Li. 2025b. Token signature: Predicting chain-of-thought gains with token decoding feature in large language models. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*. OpenReview.net.

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

Feng Luo, Yu-Neng Chuang, Guanchu Wang, Hoang Anh Duy Le, Shaochen Zhong, Hongyi Liu, Jiayi Yuan, Yang Sui, Vladimir Braverman, Vipin Chaudhary, and Xia Hu. 2025a. Autol2s: Auto long-short reasoning for efficient large language models. *CoRR*, abs/2505.22662.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Erran Li, Raluca Ada Popa, and Ion Stoica. 2025b. 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-19681902c1468005bed8ca303013a4e2. Notion Blog.

OpenAI. 2024. Learning to reasoning with llms. https://openai.com/index/learning-to-reasoning-with-llms/. [Accessed 12-05-2025].

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First conference on language modeling*.

David W Scott. 2015. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient RLHF framework. In *Proceedings of the Twentieth European Conference on Computer Systems, EuroSys 2025, Rotterdam, The Netherlands, 30 March 2025 - 3 April 2025*, pages 1279–1297. ACM.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, Hanjie Chen, and Xia Hu. 2025. Stop overthinking: A survey on efficient reasoning for large language models. *Trans. Mach. Learn. Res.*, 2025.

Renliang Sun, Wei Cheng, Dawei Li, Haifeng Chen, and Wei Wang. 2025. Stop when enough: Adaptive early-stopping for chain-of-thought reasoning. *CoRR*, abs/2510.10103.

Qwen Team. 2025. Qwq-32b: Embracing the power of reinforcement learning. https://qwenlm.github.io/blog/qwq-32b/. [Accessed 12-05-2025].

Songjun Tu, Jiahao Lin, Xiangyu Tian, Qichao Zhang, Linjing Li, Yuqian Fu, Nan Xu, Wei He, Xiangyuan Lan, Dongmei Jiang, and Dongbin Zhao. 2025a. Enhancing LLM reasoning with iterative DPO: A comprehensive empirical investigation. *CoRR*, abs/2503.12854.

Songjun Tu, Jiahao Lin, Qichao Zhang, Xiangyu Tian, Linjing Li, Xiangyuan Lan, and Dongbin Zhao. 2025b. Learning when to think: Shaping adaptive reasoning in r1-style models via multi-stage RL. *CoRR*, abs/2505.10832.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. 2025a. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for LLM reasoning. *CoRR*, abs/2506.01939.

Yibo Wang, Li Shen, Huanjin Yao, Tiansheng Huang, Rui Liu, Naiqiang Tan, Jiaxing Huang, Kai Zhang, and Dacheng Tao. 2025b. R1-compress: Long chain-of-thought compression via chunk compression and search. *CoRR*, abs/2505.16838.

Violet Xiang, Chase Blagden, Rafael Rafailov, Nathan Lile, Sang T. Truong, Chelsea Finn, and Nick Haber. 2025. Just enough thinking: Efficient reasoning with adaptive length penalties reinforcement learning. *CoRR*, abs/2506.05256.

Wenkai Yang, Shuming Ma, Yankai Lin, and Furu Wei. 2025. Towards thinking-optimal scaling of test-time compute for LLM reasoning. *CoRR*, abs/2502.18080.

Jingyang Yi and Jiazheng Wang. 2025. Shorterbetter: Guiding reasoning models to find optimal inference length for efficient reasoning. *CoRR*, abs/2504.21370.

Hang Yuan, Bin Yu, Haotian Li, Shijun Yang, Christina Dan Wang, Zhou Yu, Xueyin Xu, Weizhen Qi, and Kai Chen. 2025. Not all tokens are what you need in thinking. *CoRR*, abs/2505.17827.

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

Wenhao Zeng, Yaoning Wang, Chao Hu, Yuling Shi, Chengcheng Wan, Hongyu Zhang, and Xiaodong Gu. 2025b. Pruning the unsurprising: Efficient code reasoning via first-token surprisal. *CoRR*, abs/2508.05988.

Zhiyuan Zeng, Qinyuan Cheng, Zhangyue Yin, Yunhua Zhou, and Xipeng Qiu. 2025c. Revisiting the test-time scaling of o1-like models: Do they truly possess test-time scaling capabilities? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 4651–4665. Association for Computational Linguistics.

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

A Details of Adaptive Mode Selection

Our first objective is to endow the model with an *adaptive thinking* capability: for easy problems, the model should preferably answer directly without explicit reasoning (NO-THINK); for harder problems, it should choose to generate a longer CoT (THINK). We follow prior work that introduces an explicit control over the reasoning mode and trains it with group relative policy optimization (GRPO).

Dual reasoning modes and base rewards. Following the reward formulation proposed by Tu et al. (2025b), we define the base reward r_{base} to incentivize efficiency alongside accuracy. Let mode $\in \{\text{THINK}, \text{NO-THINK}\}$ denote the chosen

mode for a rollout. Given ground-truth correctness $\text{correct} \in \{0, 1\}$ of the final answer, we adopt the following base reward:

$$r_{\text{base}} = \begin{cases} +2, & \text{if mode} = \text{NO-THINK}, \ \text{correct} = 1, \\ -1, & \text{if mode} = \text{NO-THINK}, \ \text{correct} = 0, \\ +1, & \text{if mode} = \text{THINK}, \ \text{correct} = 1, \\ -1, & \text{if mode} = \text{THINK}, \ \text{correct} = 0. \end{cases} \tag{17}$$

This design prefers correct answers without explicit thinking whenever possible, while still rewarding correct answers obtained via longer reasoning. To avoid reward collapse to a single mode (e.g., always THINK), we further introduce a soft balancing factor. Let α_{think} and α_{no} denote the empirical proportions of THINK and NO-THINK rollouts in the current group (or mini-batch), respectively. We scale the base reward by $(1 - \alpha_{\text{mode}})$:

$$r_{\text{balanced}} = \begin{cases} r_{\text{base}} \cdot (1 - \alpha_{\text{think}}), & \text{if mode} = \text{THINK}, \\ r_{\text{base}} \cdot (1 - \alpha_{\text{no}}), & \text{if mode} = \text{NO-THINK}. \end{cases} \tag{18}$$

This encourages a balanced use of both modes during training.

B Details of Length-aware Shaping

Tu et al. (2025b) has proposed to adjust rewards based on the reasoning length in order to implicitly control the amount of thinking. Let r_i^{naive} denote a base reward (e.g., correctness-based) for trajectory i, and let y_i be a scalar reflecting its reasoning length (e.g., the number of reasoning tokens or a normalized length). For a correct final answer ($\text{correct}_i = 1$), the adjusted reward is

$$r_i^{\text{adj}} = r_i^{\text{naive}} + \left(-1 + e^{-\alpha y_i}\right), \text{if correct}_i = 1, \tag{19}$$

and for an incorrect final answer ($\text{correct}_i = 0$),

$$r_i^{\text{adj}} = r_i^{\text{naive}} + \left(1 - e^{-\beta y_i}\right), \text{if correct}_i = 0. \tag{20}$$

Here $\alpha, \beta > 0$ control the sensitivity to length. Equation (19) encourages *longer* reasoning for correct trajectories (the adjustment term is increasing in y_i from -1 to 0), while (20) encourages *shorter* reasoning for incorrect trajectories (the adjustment term increases from 0 to 1 as y_i grows).

C Estimation Methodology

To construct the density profiles shown in Figure 4, we aggregated token-level utility values across the entire AIME24 evaluation set. Formally, let $\mathcal{D} = \{(x^{(i)}, y^{\star(i)})\}_{i=1}^{M}$ denote the dataset containing M problems. For each problem $x^{(i)}$, the model generates a reasoning chain of length L_i, resulting in a sequence of utility values $\mathbf{\Delta}^{(i)} = [\Delta_1^{(i)}, \ldots, \Delta_{L_i}^{(i)}]$. We compile the global set of utility samples $\mathcal{S} = \bigcup_{i=1}^{M} \{\Delta_t^{(i)} \mid 1 \le t \le L_i\}$, comprising all generated tokens across the benchmark.

The probability density function (PDF) of the marginal utility is then estimated using Kernel Density Estimation (KDE). For a target utility value u, the estimated density $\hat{f}(u)$ is given by:

$$\hat{f}(u) = \frac{1}{Nh} \sum_{j=1}^{N} K\left(\frac{u - s_j}{h}\right), \tag{21}$$

where $N = |\mathcal{S}|$ is the total number of tokens (typically $N > 10^5$ in our experiments), $s_j \in \mathcal{S}$ represents the j-th observed utility value, $K(\cdot)$ is the kernel function, and $h > 0$ is the bandwidth parameter smoothing the distribution. In our analysis, we employ a standard **Gaussian kernel** $K(z) = \frac{1}{\sqrt{2\pi}} e^{-z^2/2}$. To ensure an unbiased representation of the distribution shape without over-smoothing, we determine the optimal bandwidth h using **Scott's Rule** (Scott, 2015), defined as $h = n^{-1/5} \cdot \sigma$, where σ is the standard deviation of the sample set $\mathcal{S}$. This non-parametric approach allows us to visualize the continuous distribution of token utility without the binning artifacts associated with histograms.

Methods	MATH	Minerva	Olym.	AIME	AMC	Avg.
Qwen3-1.7B	86.4 / 5495	33.5 / 4419	44.4 / 6508	50.0 / 11671	57.5 / 7615	54.4 / 7141.6
+MUTO (ours)	86.2 / 1276	34.2 / 1559	44.6 / 1954	53.3 / 2654	60.0 / 1866	55.7 / 1861.6
Δ vs. Base	-0.2 / -76.8%	+0.7 / -64.7%	+0.2 / -70.0%	+3.3 / -77.3%	+2.5 / -75.5%	+1.3 / -73.9%

Table 4: Generalization to another model family, Qwen3-1.7B. Each cell reports Accuracy (%) / Average Token Usage. MUTO substantially improves efficiency while preserving or improving accuracy across most benchmarks.

D Other deep-thinking model families.

To verify that the gains are not tied to the DeepSeek-R1-Distill family, we further evaluate MUTO on Qwen3-1.7B, another backbone that exhibits deep-thinking behavior. As shown in Table 4, MUTO improves average accuracy from 54.4 to 55.7 (+1.3) while reducing average token usage from 7,141.6 to 1,861.6 (-73.9%). The improvement is especially pronounced on AIME (+3.3) and AMC (+2.5), alongside large token reductions on all benchmarks. This close match to our main findings indicates that MUTO generalizes across model families when the baseline model exhibits prolonged exploratory reasoning.