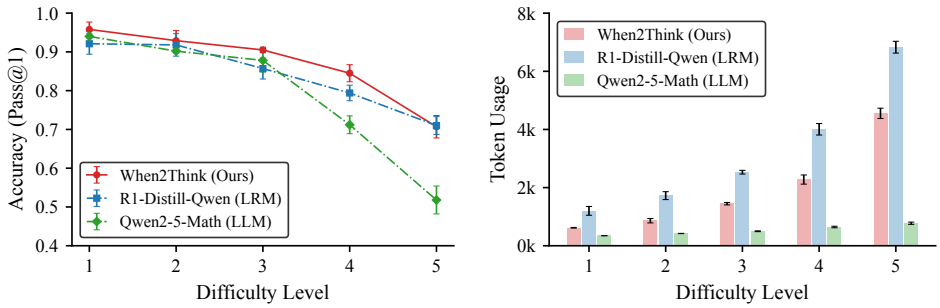
WHEN2THINK: LEARNING WHEN AND HOW MUCH TO REASON

Anonymous authors
Paper under double-blind review

ABSTRACT

Large Reasoning Models (LRMs) often overthink easy problems and underthink hard ones, leading to inefficient computation allocation. Existing methods regulate generated computation or select between direct answering and explicit reasoning, but do not jointly control *whether* to reason and *how much* computation to allocate within reasoning. We call the resulting difficulty-dependent loss in accuracy under computation reduction the *efficiency tax*. We propose *When2Think*, an RLVR-based post-training framework for instance-adaptive computation allocation. Its core mechanism, *Instance-level Difficulty-Aware Control (IDAC)*, uses cached reference statistics of success and token cost to modulate a correctness-gated efficiency bonus based on generated token count. Importance sampling supports exploration of THINK and NOTHINK, while Batch-Wise Standardization constructs standardized advantages for critic-free optimization. The framework requires neither a learned reward model nor a learned critic, and offline reference caching avoids online reference-model queries during policy updates. On AIME24, *When2Think* improves `Pass@3` by 10.0 percentage points while reducing token usage by 27.9% relative to the backbone.

1 INTRODUCTION



(a) MATH-500 Accuracy

(b) MATH-500 Tokens

Figure 1: **Performance and efficiency on the MATH-500 benchmark** stratified by problem difficulty (Levels 1–5). We compare *When2Think* (red), LRM baseline (blue) and LLM baseline (green). **Lines:** Mean `Pass@1` accuracy ± standard deviation. **Bars:** Mean tokens per instance ± standard deviation. Statistics computed over five independent sampling runs. *When2Think* mitigates the efficiency tax by reducing token usage on easier instances while preserving hard-instance accuracy.

The emergence of *large language models (LLMs)* (Brown et al., 2020; Rae et al., 2022; Hoffmann et al., 2022; Chowdhery et al., 2023; Touvron et al., 2023) has advanced language understanding and generation across a wide range of tasks. Beyond fluent text generation, many challenging tasks require reliable multi-step reasoning (Zelikman et al., 2022; Sprague et al., 2025). A prominent direction is to internalize structured reasoning capabilities through *supervised fine-tuning (SFT)* or *reinforcement fine-tuning (RFT)* (Huang & Chang, 2023; Tie et al., 2025; Kumar et al., 2026), producing **large reasoning models (LRMs)** that generate coherent reasoning trajectories and achieve strong performance on challenging benchmarks (Ke et al., 2025; Xu et al., 2025; Zhang et al., 2025a).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Despite these advances, LRMs suffer from *reasoning inefficiency* (Feng et al., 2025). They often **overthink** simple tasks by generating redundant reasoning tokens without accuracy gains (Chen et al., 2025b; Sui et al., 2025), yet **underthink** complex tasks by terminating reasoning prematurely or producing fragmented chains (Wang et al., 2025b). Existing efficiency methods reduce computation through computation control or reasoning-mode selection, but can under-allocate computation to harder instances and sacrifice accuracy (Table 1). We refer to this difficulty-dependent loss in hard-instance accuracy from reducing computation as the **efficiency tax**, highlighting the need to control both *whether* and *how much* to reason.

We adopt *dual-process theory* as a conceptual lens for this allocation problem (Sloman, 1996; Hua & Zhang, 2022). SYSTEM 1-style direct answering provides a low-cost operating point, whereas SYSTEM 2-style deliberation enables explicit multi-step reasoning (Zhang et al., 2025a). As shown in Figure 1, the LLM baseline uses fewer tokens and performs competitively on easier instances, whereas the LRM baseline benefits more from additional computation as difficulty increases (Shojaee et al., 2025; Wang et al., 2025a). This contrast motivates *single-model hybrid reasoning*, where a single model adapts computation to the input by selecting between direct answering and explicit deliberation, while varying computation depth within deliberation.

Motivated by these observations, we propose *When2Think*, a post-training framework for difficulty-aware computation control. At its core, *Instance-level Difficulty-Aware Control (IDAC)* modulates computation-sensitive rewards using reference-estimated difficulty and generated trajectory token count. Importance-sampled THINK/NOTHINK exploration supports direct answering as an additional low-cost behavior, while IDAC controls how deeply reasons within the THINK mode.

We make the following contributions:

- **Difficulty-stratified analysis.** A difficulty-stratified analysis connects overthinking and underthinking as manifestations of a shared computation-allocation problem, characterizing an *efficiency tax* in which computation savings come at the cost of hard-instance accuracy.
- **Difficulty-aware computation control.** *Instance-level Difficulty-Aware Control (IDAC)* modulates a correctness-gated efficiency bonus using cached reference statistics and the generated token count of each trajectory.
- **Mechanism and behavioral analysis.** The THINK-only IDAC+BWS variant outperforms standard RFT on most evaluated benchmarks, demonstrating performance gains within explicit reasoning independently of NOTHINK selection.

2 RELATED WORK

We review prior work on reducing the inference cost of large reasoning models (LRMs) through post-training. Following recent surveys (Feng et al., 2025; Sui et al., 2025), we organize this literature by the primary control variable: (1) *computation control*, which regulates the amount or structure of generated reasoning, and (2) *hybrid Reasoning mode control*, which determines whether to use direct answering or explicit deliberation. Inference-time budgets, stopping rules, and routers provide complementary control over fixed policies.

Computation Control. Global methods apply common efficiency constraints across inputs, including iterative token limits in ThinkPrune (Hou et al., 2026), fixed-target step rewards in LASER (Liu et al., 2026), and optimization-aware truncation in DLER (Liu et al., 2025). Structural approaches remove redundant reasoning segments through chunk-level distillation (Skip-Thinking (Chen et al., 2025a)) or sufficient-prefix extraction (LC-R1 (Cheng et al., 2026)). Adaptive methods derive problem-dependent computation signals from reference statistics or current-policy rollouts, including explicit token budgets in DAST (Shen et al., 2025), prompt-relative length rewards (Arora & Zanette, 2025), shortest-correct targets in ShorterBetter (Yi et al., 2025), and dynamically adjusted difficulty-specific thresholds in LASER and difficulty-aware DLER (Liu et al., 2026; 2025). A complementary line studies optimization and credit assignment, showing that computation-sensitive objectives can fail through harmful advantage estimates, entropy collapse, or sparse rollout feedback (Li et al., 2026; Liu et al., 2025). These studies establish global, structural, and instance-adaptive computation control, but primarily regulate computation within a trajectory.

Reasoning Mode Control. Hybrid methods learn whether an input should receive a direct response or explicit deliberation. ARM (Wu et al., 2025) selects among multiple reasoning formats using diversity-aware reward scaling. Among binary hybrid policies, LHRM (Jiang et al., 2025) separates inter-mode utility comparison from intra-mode response optimization, while ThinkLess (Fang et al., 2025) decouples the control-token and response-token objectives to address length-induced gradient imbalance. AdaptThink (Zhang et al., 2025b) formulates THINK/NOTHINK selection as a reference-relative constrained objective and uses Importance Sampling to maintain exploration of both modes. Inference-time alternatives instead route fixed policies using prompts, confidence estimates, external verifiers, or internal-state probes (Tan et al., 2025). These methods primarily control computation through discrete mode or format decisions, while instance-conditioned control of computation within the selected deliberative mode remains less explored.

3 PRELIMINARIES

Post-training is one effective mechanism for encouraging SYSTEM 2-style reasoning in LLMs by modifying model parameters, leading to models commonly referred to as *Large Reasoning Models (LRMs)* or *Reasoning Language Models (RLMs)* (OpenAI et al., 2024; Team, 2025; Guo et al., 2025; Abdin et al., 2025; Comanici et al., 2025). We focus on *reinforcement learning (RL)* approaches within the broader paradigm of *learning to reason* (Huang & Chang, 2023; Besta et al., 2025; Kumar et al., 2026), which directly optimize reasoning policies to enhance both accuracy and efficiency (Trung et al., 2024; Ke et al., 2025; Xu et al., 2025; Zhang et al., 2025c).

Verifiable Rewards. Lambert et al. (2025) introduce *Reinforcement Learning with Verifiable Rewards (RLVR)*, which replaces learned reward models with deterministic verification, directly aligning the RL objective with task correctness and reducing reward-model bias. Prior work (Cobbe et al., 2021; Lightman et al., 2024) uses verifiers either as *Outcome-supervised Reward Models (ORMs)* for final answers or *Process-supervised Reward Models (PRMs)* for step-level supervision, trading simplicity for annotation cost. RLVR leverages outcome-based verification to provide reliable rewards for multi-step reasoning without requiring step-level supervision.

Importance Sampling. To mitigate cold-start bias and avoid premature collapse to THINK, we adopt the importance-sampling strategy of AdaptThink (Zhang et al., 2025b). During rollout collection, an auxiliary policy π_{IS} selects THINK or NOTHINK with equal probability, after which all tokens are generated by the current policy π_θ. The collected trajectories are reweighted during optimization to account for the modified mode-selection distribution. This mechanism provides balanced hybrid-mode exploration, whereas our contribution focuses on difficulty-aware control of reasoning depth. At inference time, π_{IS} is removed and the trained policy selects the mode directly.

Clipped Surrogate Objective. We optimize the target policy π_θ using a PPO-style clipped policy gradient objective following AdaptThink (Zhang et al., 2025b), which stabilizes learning of when to invoke explicit reasoning. Although the objective is defined with respect to trajectories generated by π_θ, in practice we optimize using samples collected from an auxiliary policy π_{IS}, with importance weighting to ensure consistency with the target policy.

Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ denote a dataset of N instances. During training, we sample a mini-batch $\mathcal{B} \subset \mathcal{D}$ of size M and reindex its elements as $\{(x_i, y_i)\}_{i=1}^{M}$. For each $x_i \in \mathcal{B}$, we sample K reasoning trajectories, denoted by $\{o_{i,k}\}_{k=1}^{K}$. The trajectory-level importance ratio:

$$\rho_{i,k} = \frac{\pi_\theta(o_{i,k} \mid x_i)}{\pi_{\text{IS}}(o_{i,k} \mid x_i)}, \tag{1}$$

and the clipped surrogate loss:

$$\mathcal{L}(\theta) = -\mathbb{E}_{x_i \sim \mathcal{B}, o_{i,k} \sim \pi_{\text{IS}}(\cdot|x_i)} \Big[\min(\rho_{i,k}\, A_{i,k}, \min(\max(\rho_{i,k}, 1-\epsilon), 1+\epsilon)\, A_{i,k}) \Big], \tag{2}$$

where $A_{i,k}$ denotes a trajectory-level advantage estimate and ϵ is the PPO clipping parameter.

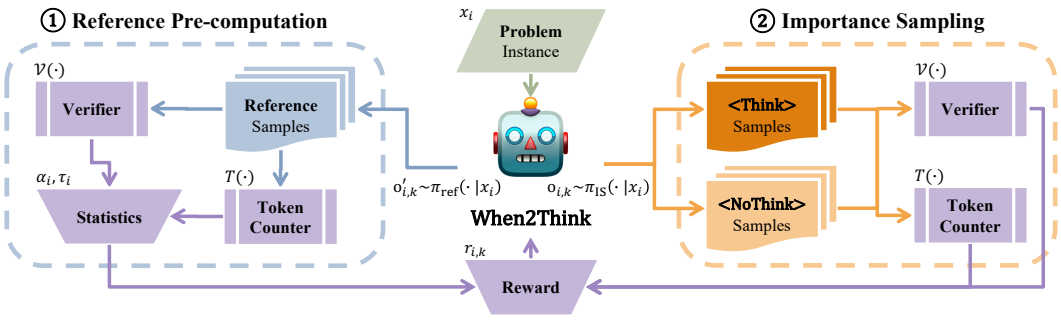


Figure 2: **Overview of the *When2Think* RLVR framework** for adaptive hybrid reasoning. The framework comprises two stages. **(Left) Reference pre-computation:** A reference policy π_{ref} generates trajectories for each input x_i to estimate instance-level difficulty α_i and expected reasoning cost τ_i. **(Right) Training with Importance Sampling:** An exploration distribution π_{IS} enforces balanced exploration between THINK (explicit reasoning) and NOTHINK (direct answering) by sampling the initial mode token uniformly, while subsequent tokens are generated by the policy π_θ.

4 WHEN2THINK

We propose *When2Think*, a Reinforcement Learning with Verifiable Rewards (RLVR)-based post-training framework for difficulty-aware computation allocation in hybrid reasoning. For each input, *When2Think* regulates computation within explicit reasoning while retaining direct answering (NOTHINK) as an additional low-cost behavior. This allocation is guided by cached reference statistics that estimate instance difficulty and expected reasoning cost (Figure 2). A verifier provides trajectory-level correctness signals, Importance Sampling (IS) supports balanced THINK/NOTHINK exploration, Instance-level Difficulty-Aware Control (IDAC) modulates a correctness-gated efficiency bonus using these statistics and generated token counts, and Batch-Wise Standardization (BWS) constructs standardized advantages for policy optimization. Together, these components enable stable optimization without a learned reward model or critic. Reference statistics are computed once per epoch and cached, so subsequent policy updates require no online reference-model queries.

4.1 REFERENCE STATISTICS

To estimate instance difficulty and expected reasoning cost without querying the reference model during inner-loop optimization, we pre-compute epoch-wise reference statistics using a reference policy π_{ref} over the dataset $\mathcal{D}$ and cache them for subsequent policy update steps. For each input $x_i \in \mathcal{D}$, we sample K independent trajectories $o'_{i,k} \sim \pi_{\text{ref}}(\cdot \mid x_i)$. The final answer is extracted from each trajectory using $g(\cdot)$. We then define the answer verification function:

$$\mathcal{V}(x_i, o'_{i,k}) = \mathbb{I}\big[\, g(o'_{i,k}) = y_i \,\big], \tag{3}$$

where $\mathbb{I}[\cdot]$ equals 1 if the extracted answer matches the ground-truth y_i, and 0 otherwise.

Instance Difficulty and Cost. To estimate the difficulty and expected reasoning cost of each instance, we compute the reference success rate and average reference token count:

$$\alpha_i = \gamma \left(\frac{1}{K} \sum_{k=1}^{K} \mathcal{V}(x_i, o'_{i,k}) \right) + \varepsilon, \qquad \tau_i = \frac{1}{K} \sum_{k=1}^{K} \mathcal{T}(o'_{i,k}) + \varepsilon. \tag{4}$$

Here, α_i is the scaled empirical success rate of the reference policy on instance x_i and serves as a model-relative proxy for instance difficulty, with larger values indicating easier instances. The term τ_i denotes the average reference token count and provides an instance-specific reference cost scale. The function $\mathcal{T}(\cdot)$ returns the number of generated tokens in a trajectory, while $\gamma \in (0, 1)$ controls reward scaling and $\varepsilon > 0$ ensures numerical stability. We use $\gamma = 0.9$ in our experiments.

4.2 INSTANCE-ADAPTIVE REWARD

The adaptive reward combines verifier-based correctness, an instance-level reference baseline, and a correctness-gated efficiency bonus. The reference baseline centers trajectory rewards according to the reference policy's empirical success rate for each instance, while IDAC modulates the efficiency bonus using the reasoning mode, generated token count, and cached reference statistics. BWS subsequently constructs standardized advantages from the resulting trajectory rewards.

Instance-level Difficulty-Aware Control. We introduce *Instance-level Difficulty-Aware Control (IDAC)* to regulate computation according to instance difficulty. Formally, it is defined:

$$\tilde{\lambda}_{i,k} = \exp\left(-\frac{\mathcal{T}(o_{i,k})\,\alpha_i}{\tau_i}\right), \qquad \lambda_{i,k} = \begin{cases} \tilde{\lambda}_{i,k} & \text{if Think,} \\ 1 & \text{if NoThink.} \end{cases} \tag{5}$$

For THINK trajectories, $\lambda_{i,k}$ decreases as the generated token count increases relative to the reference cost scale, with stronger decay at higher reference success for a fixed cost scale. For NOTHINK trajectories, $\lambda_{i,k} = 1$. Figure 3 illustrates this THINK-mode scaling behavior.

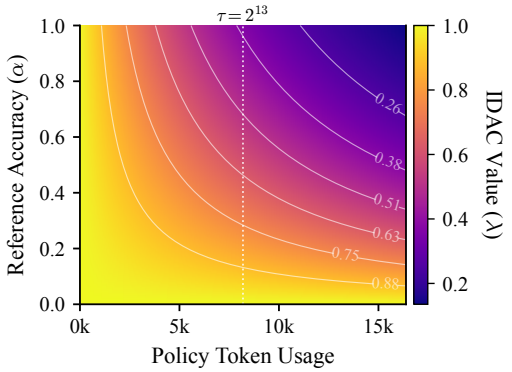


Figure 3: **Visualization of the Instance-level Difficulty-Aware Control (IDAC) mechanism.** The heatmap shows the IDAC scaling factor applied to THINK trajectories as a function of generated token count and reference success rate. For visualization, the reference cost scale is fixed at $\tau = 2^{13}$ tokens, half of the maximum THINK token limit of 2^{14}. Higher reference success rates and larger generated token counts produce stronger decay.

Adaptive Reward. The final reward combines the answer verification function with an instance-level reference baseline and a correctness-gated efficiency bonus:

$$r_{i,k} = \mathcal{V}(x_i, o_{i,k})\,(1 + \lambda_{i,k}\,\delta) - \alpha_i. \tag{6}$$

Here, $\mathcal{V}(x_i, o_{i,k}) \in \{0, 1\}$ is the binary correctness signal provided by the verifier, $\lambda_{i,k}$ is the IDAC scaling factor, and $\delta > 0$ controls the magnitude of the efficiency bonus. The term α_i serves as an instance-specific reference baseline, calibrating trajectory rewards against the reference policy's empirical success rate on x_i. For correct trajectories, NOTHINK receives the full efficiency bonus. For THINK, the bonus decreases with generated token count, with stronger decay for higher reference success relative to the reference cost scale.

4.3 BATCH-WISE STANDARDIZED ADVANTAGE

To enable stable trajectory-level credit assignment without a learned critic, we adopt *batch-wise standardization (BWS)*, drawing upon the stabilization insights of *REINFORCE++* (Hu et al., 2025).

Given a mini-batch of M instances, we sample K trajectories for each instance i, $\{o_{i,k}\}_{k=1}^{K}$, and compute a scalar adaptive reward $r_{i,k}$ for each trajectory. For each trajectory index k, we compute the batch-wise mean and standard deviation of rewards across the M instances:

$$\mu_k = \frac{1}{M}\sum_{i=1}^{M} r_{i,k}, \qquad \sigma_k = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(r_{i,k} - \mu_k)^2}. \tag{7}$$

The trajectory-level advantage is defined:

$$A_{i,k} = \frac{r_{i,k} - \mu_k}{\sigma_k + \varepsilon}, \tag{8}$$

where $\varepsilon > 0$ is a small constant for numerical stability. All tokens in a trajectory share the same standardized advantage, yielding a low-variance estimator that naturally supports trajectory-level credit assignment in the PPO-style clipped surrogate optimization described in § 3.

5 EXPERIMENTAL SETUP

We evaluate whether this adaptive control maintains accuracy on difficult instances while reducing unnecessary computation on easier ones. Implementation details are provided in § A.

5.1 TRAINING

We train *When2Think* on a reasoning-capable base model, *R1-distill-1.5B*, a compact language model distilled from *DeepSeek-R1* (Guo et al., 2025) and built on *Qwen2.5-Math-1.5B* (Yang et al., 2024). The model was distilled using approximately 800k curated samples via SFT. Training is conducted on the *DeepScaleR* dataset (Tan et al., 2026), which contains approximately 40k competition-level mathematics problems with verified solutions. The dataset aggregates publicly available sources, including AIME (1984–2023), AMC (pre-2023), Omni-MATH (Gao et al., 2025), and Still (Min et al., 2024), covering algebra, geometry, number theory, and combinatorics.

5.2 EVALUATION

We evaluate models on mathematical reasoning benchmarks spanning grade-school arithmetic, olympiad challenges, competition-level problems, and undergraduate STEM tasks, including *GSM-Plus* (Li et al., 2024), *OlympiadBench* (He et al., 2024), *AIME I/II* (2024–2025), *Minerva* (Lewkowycz et al., 2022), and *MATH-500* (Lightman et al., 2024). We additionally evaluate cross-domain transfer on *MMLU-Pro Stratified* (Shi et al., 2025), which extends beyond the mathematical post-training distribution.

We evaluate models along two dimensions: *answer accuracy* and *token usage*. Accuracy is measured using sampling-based `Pass@k` (Chen et al., 2021), while efficiency is measured by the average number of generated tokens per instance, computed independently of the `Pass@k` sampling budget. Final answers are validated using two verifiers, *Math-Verify* (Kydlíček) and *MARIO Eval* (Zhang et al., 2024); a prediction is considered correct if accepted by either verifier. Further evaluation settings are provided in § B.

5.3 BASELINES

We compare *When2Think* with representative baselines from three categories: (1) base LLMs, (2) standard LRMs trained via SFT or RFT, and (3) efficiency-aware LRMs. All evaluated models are built on the Qwen2.5 (Qwen et al., 2025) family, providing a broadly consistent architectural basis for comparison. The base LLMs include the general-purpose *Qwen2.5-Instruct* (Qwen et al., 2025) and math-specialized *Qwen2.5-Math-Instruct* (Yang et al., 2024). For explicit reasoning, we evaluate *R1-Distill-Qwen* (Guo et al., 2025), which also serves as the backbone of *When2Think*, and *DeepScaleR-Preview* (Tan et al., 2026), which is trained on the same *DeepScaleR* dataset. Following the control variables identified in § 2, we group efficiency-aware baselines into computation-control and reasoning-mode-control methods. Computation-control baselines include *LC-R1* (Cheng et al., 2026), *ThinkPrune* (Hou et al., 2026), and *LASER* (Liu et al., 2026). Reasoning-mode-control baselines include *AdaptThink* (Zhang et al., 2025b) and *ThinkLess* (Fang et al., 2025). Among these efficiency-aware baselines, *LASER*, *AdaptThink*, and *ThinkLess* are trained on the same *DeepScaleR* dataset as *When2Think*, enabling more controlled comparisons of their post-training objectives (Tan et al., 2026; Liu et al., 2026; Fang et al., 2025; Zhang et al., 2025b).

6 EXPERIMENTAL RESULTS & ANALYSIS

We evaluate *When2Think* through three questions covering trade-off, allocation, and mechanism.

RQ1: Accuracy–Efficiency Trade-off under Adaptive Allocation. Does *When2Think* improve the accuracy–efficiency trade-off and mitigate the *efficiency tax*?

RQ2: Difficulty-Sensitive Mode and Computation Allocation. Does the policy allocate less computation to easy instances and preserve deeper reasoning on hard ones?

RQ3: Mechanism Effects. How do *Instance-level Difficulty-Aware Control (IDAC)*, *Batch-Wise Standardization (BWS)*, *Importance Sampling (IS)* enable stable control of reasoning depth?

Table 1: **Performance comparison on mathematical reasoning benchmarks**. We report `Pass@3` accuracy (%, higher is better), average tokens per response (`Tokens`, lower is better) on GSM-Plus, and AIME24. All results use sampling-based `Pass@k` with matched decoding settings and are reported as mean ± standard deviation over five independent sampling runs. *When2Think* is compared with base LLMs, LRM-based baselines, and efficiency-oriented reasoning models. Parentheses indicate differences relative to R1-Distill-Qwen (green: improvement; red: degradation).

Model \ Benchmark	GSM-Plus		AIME24		AIME25	
	↑ Pass@3(%)	↓ Tokens(#)	↑ Pass@3(%)	↓ Tokens(#)	↑ Pass@3(%)	↓ Tokens(#)
Qwen2.5-Math-Instruct	84.3 ±0.3	360 ±1.8	22.7 ±1.5	1,057 ±57.7	19.3 ±2.8	923 ±64.8
R1-Distill-Qwen (Base)	79.4 ±0.3	590 ±11.6	46.0 ±2.8	14,195 ±1,026.7	32.0 ±5.1	12,616 ±688.4
DeepScaleR-Preview	85.4 ±0.3	1,358 ±13.4	58.0 ±5.6	8,473 ±479.5	39.3 ±6.8	8,074 ±288.6
LC-R1	79.0 ±0.3 (- 0.4)	546 ±9.9 (- 44)	36.0 ±4.9 (- 10.0)	7,972 ±1,024.3 (- 6,223)	32.0 ±3.8 (± 0)	6,971 ±278.7 (- 5,645)
ThinkPrune-iter3k	81.8 ±0.3 (+ 2.4)	609 ±5.5 (+ 19)	46.0 ±4.3 (± 0)	6,236 ±555.4 (- 7,959)	36.0 ±3.7 (+ 4.0)	5,582 ±273.3 (- 7,034)
Laser-DE-L4096	80.1 ±0.2 (+ 0.7)	562 ±4.1 (- 28)	45.3 ±3.8 (- 0.7)	8,425 ±362.3 (- 5,770)	37.3 ±8.9 (+ 5.3)	6,971 ±1,024.3 (- 5,645)
AdaptThink-delta0.05	83.6 ±0.2 (+ 3.2)	716 ±17.5 (+ 126)	44.7 ±4.5 (- 1.3)	5,806 ±438.8 (- 8,389)	30.7 ±8.0 (- 1.3)	6,883 ±440.2 (- 5,733)
Thinkless-RL	85.8 ±0.2 (+ 6.4)	1,799 ±16.2 (+ 1,209)	46.7 ±4.1 (+ 0.7)	11,023 ±209.4 (- 3,172)	33.3 ±7.5 (+ 1.3)	11,056 ±641.7 (- 1,560)
When2Think (Ours)	85.7 ±0.3 (+ 6.3)	1,052 ±17.9 (+ 462)	56.0 ±2.8 (+ 10.0)	10,236 ±438.0 (- 3,959)	40.0 ±0.0 (+ 8.0)	9,549 ±719.4 (- 3,067)
w/o IS (IDAC + BWS)	86.8 ±0.1 (+ 7.4)	1,652 ±13.2 (+ 1,062)	57.3 ±8.3 (+ 11.3)	10,046 ±487.9 (- 4,149)	40.0 ±3.3 (+ 8.0)	9,846 ±269.2 (- 2,770)

6.1 RQ1: Accuracy–Efficiency Trade-off under Adaptive Allocation

We evaluate whether *When2Think* improves the accuracy–efficiency trade-off by reducing unnecessary reasoning on easy instances without inducing under-allocation on hard ones.

The Efficiency Tax. Many efficient reasoning methods reduce computation through fixed length constraints, heuristic compression, or discrete mode switching. While these strategies can suppress redundant reasoning on easy instances, they may incur an *efficiency tax*: token savings are obtained by removing reasoning that is still needed on harder instances. For example, on AIME24 (Table 1), LC-R1 and AdaptThink reduce inference cost relative to the R1-Distill-Qwen backbone, but their accuracy decreases by 10.0 and 1.3 points, respectively. A similar hard-instance under-allocation pattern appears on MATH-500 Level 5 (Table 10 in § C.3).

Top-Right Accuracy–Efficiency Improvements. Figure 6 shows the accuracy–efficiency landscape across five benchmarks: GSM-Plus, Minerva, OlympiadBench, AIME24, and AIME25. With the token axis reversed, the top-right region represents higher accuracy with fewer tokens. *When2Think* does not necessarily achieve the lowest token usage; routing-only methods such as AdaptThink can reach lower-cost operating points on easier tasks. Instead, *When2Think* targets computation reduction while maintaining accuracy on harder instances. On AIME24, it reduces token usage by 3,959 tokens (−27.9%) while improving accuracy from 46.0% to 56.0% (10.0 points), showing that adaptive allocation can reduce the efficiency tax rather than merely shift it.

6.2 RQ2: Difficulty-Sensitive Mode and Computation Allocation

We analyze whether *When2Think* dynamically allocates computation in a *difficulty-aware* manner, balancing System 1/2 across benchmarks of varying difficulty.

System 1 Preference on Easy Instances. On easy MATH-500 (Table 10 in § C.3) Level 1 problems, the R1-Distill generates an average of 1,199 tokens despite minimal reasoning demand. In contrast, *When2Think* reduces token usage to 619 (−580) while maintaining high accuracy (95.8%), indicating a clear preference for System 1-style direct answering when additional deliberation provides limited benefit. On GSM-Plus, *When2Think* instead allocates additional computation to perturbed inputs, improving Pass@3 by 6.3 percentage points at a higher average token cost.

SYSTEM 2 Engagement on Hard Instances. On hard MATH-500 (Table 10 in § C.3) Level 5 problems, *When2Think* maintains accuracy while reducing token by 2,276 relative to R1-Distill, indicating more selective yet sustained SYSTEM 2 reasoning. In contrast, on adversarially perturbed GSM-Plus instances, it deliberately increases computation (462 tokens), yielding a 6.3% absolute accuracy gain. On long-horizon tasks such as AIME24/25 (Table 1), *When2Think* maintains sufficient deliberation depth, while compression-based and discrete hybrid methods often truncate reasoning prematurely, leading to poorer performance on hard instances. When2Think (§ C.2) selects THINK for all evaluated AIME24 and AIME25 instances. These benchmarks therefore isolate computation control within the reasoning mode rather than active THINK/NOTHINK switching.

Instance-Level Difficulty-Aware Allocation. *When2Think* adapts reasoning behavior at the instance level. On MATH-500 (Figure 4 and § C.2), the fraction of THINK trajectories increases monotonically with difficulty, from about 0.2 at Level 1 to over 0.7 at Level 5. By contrast, baselines exhibit weak or misaligned responses: ThinkLess allocates nearly constant computation, DeepScaleR overthinks easy instances, and AdaptThink under-allocates on hard ones. This demonstrates that *When2Think* more accurately matches computation to problem difficulty.

6.3 RQ3: MECHANISM EFFECTS

We analyze the mechanisms underlying *When2Think*, focusing on the roles of IDAC, BWS, IS, and model scale. Additional ablations are provided in § D.

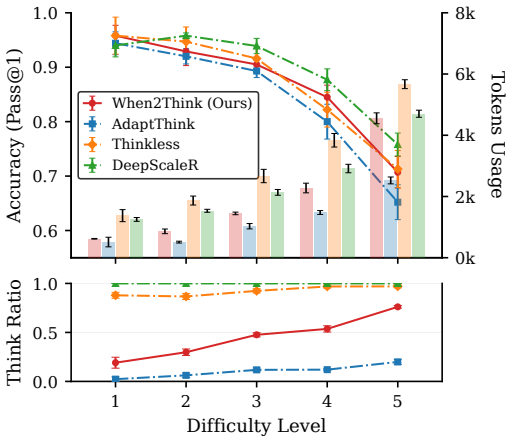


Figure 4: **Performance and behavioral analysis on the MATH-500 benchmark** stratified by problem difficulty (Levels 1–5). Results are reported for *When2Think* (red), DeepScaleR (green), *Thinkless* (orange), and AdaptThink (blue). **(Top) Accuracy and efficiency**: `Pass@1` accuracy (lines) and average token usage (bars). **(Bottom) Think ratio**: Fraction of trajectories that trigger explicit reasoning across difficulty levels. All results are reported as mean ± standard deviation over five independent sampling runs.

IDAC enables depth control. Compared with AdaptThink (Zhang et al., 2025b), which primarily learns whether to invoke THINK, *When2Think* further controls how deeply to reason once THINK is selected. On AIME24 (Table 1), AdaptThink uses fewer tokens (5,806) but drops to 44.7% Pass@3, below the R1-Distill-Qwen backbone, whereas *When2Think* achieves 56.0% Pass@3 while still reducing token usage. Consistent with this, the IS+BWS variant–which retains balanced THINK/NOTHINK exploration but removes IDAC–exhibits substantially worse validation accuracy, particularly on the hard AIME24 benchmark (Figure 10 and § D.3).

BWS stabilizes optimization. BWS converts IDAC-shaped trajectory rewards into standardized advantages, enabling stable critic-free optimization. Beyond variance reduction, it preserves cross-instance difficulty structure, allowing successful long trajectories on hard instances to receive positive credit. As shown in §§ D.1 and D.2, BWS improves convergence, stabilizes response length and think ratio, and prevents degenerate reasoning behaviors, maintaining a stable think ratio of approximately 0.9 compared to 0.6 without BWS (Figure 9).

IS improves efficiency, not accuracy. Removing Importance Sampling (while retaining IDAC and BWS, Table 1) yields a strong variant that outperforms DeepScaleR-Preview on most benchmarks, indicating that the primary accuracy gains stem from IDAC and BWS. However, IS shifts the policy toward a more efficient operating point: compared with the w/o IS variant, the full *When2Think* model substantially reduces GSM-Plus token usage (1,652 → 1,052 tokens) while maintaining competitive accuracy. This suggests that IS primarily promotes effective NOTHINK usage on easier instances, rather than serving as the main source of accuracy gains (§ D.3).

7 DISCUSSION

7.1 FROM COMPRESSION TO COMPUTATION ALLOCATION

Our results suggest that reasoning efficiency is better understood as an *instance-adaptive computation-allocation problem* than as uniform trajectory compression. Reducing computation can improve efficiency on easier instances but incur an *efficiency tax* when harder instances are under-allocated and lose accuracy (Tables 9 and 10). Effective reasoning therefore requires adapting computation to individual inputs rather than uniformly shortening responses. Difficulty-stratified evaluation is important for revealing where computation is reduced, preserved, or expanded.

7.2 MODE SELECTION AND WITHIN-THINK CONTROL

Hybrid reasoning involves two distinct but coupled decisions: whether to invoke explicit reasoning and how much computation to allocate after it is invoked. Our results show that these behaviors can vary independently. On MATH-500 (Figure 4 and Table 10), the THINK ratio increases with difficulty, while on AIME24 and AIME25 (§ C.2) the policy selects THINK for every evaluated instance. Thus, computation control within THINK remains relevant even when mode switching is inactive.

The ablations further clarify the roles of the training components. The THINK-only IDAC+BWS variant remains strong without NOTHINK exploration (Table 9), indicating that the primary performance gains do not depend on hybrid routing alone. IDAC provides instance-conditioned incentives for generated computation within THINK, while BWS converts the resulting reward structure into standardized advantages for critic-free optimization. Importance sampling plays a complementary role by maintaining coverage of both modes and enabling lower-cost NOTHINK behavior where it is effective (Table 9 and § D). The full hybrid and THINK-only policies therefore represent different accuracy–computation operating points rather than a strict performance ordering.

7.3 SCOPE AND LIMITATIONS

Allocation behavior varies with model scale and the reference policy (§§ C.3 and F). At 7B, *When2Think* reduces computation on MATH-500 but allocates more computation on AIME24 and AIME25, showing that effective allocation does not always mean shorter responses.

IDAC uses reference success and token usage as proxies for difficulty and computation cost, so its allocation behavior may depend on the reference policy. Generated token count is also only an operational proxy for inference cost. Finally, our evaluation focuses on verifiable tasks, leaving generalization to open-ended reasoning as future work.

8 CONCLUSION

We formulate efficient reasoning as an *instance-adaptive computation-allocation problem* involving two coupled decisions: whether to invoke explicit reasoning and how much computation to allocate once it is invoked. We introduce *When2Think*, an RLVR-based post-training framework that combines importance-sampled mode exploration, IDAC-based computation control, and BWS-based cross-instance credit assignment. IDAC uses cached reference success and token-cost statistics to shape a correctness-gated efficiency bonus within THINK, while BWS enables stable, critic-free optimization without a learned reward model.

Our results show difficulty-sensitive mode allocation on mixed-difficulty inputs and effective within-THINK computation control when explicit deliberation remains necessary. The THINK-only IDAC+BWS variant demonstrates that computation control within explicit reasoning can improve performance independently of NOTHINK exploration, while the full hybrid policy additionally provides a lower-cost operating point on suitable inputs. Together, these findings establish mode selection and within-mode computation as distinct but coupled components of efficient reasoning, shifting the objective from uniform trajectory shortening toward instance-adaptive computation allocation.

AI USE STATEMENT

We used generative AI agents to assist with interpreting experimental results, improving the clarity and readability of the manuscript, suggesting titles and keywords, and refactoring research code. The agents were not used to generate experimental data or conduct experiments. All AI-assisted outputs were reviewed by the authors: experimental interpretations were verified against the experimental records, manuscript revisions were checked for technical accuracy, and refactored code was inspected and tested to preserve its intended functionality. All scientific conclusions and research decisions were made by the authors, who take full responsibility for the content of this work.

ETHICS STATEMENT

This work uses existing models and publicly available mathematical reasoning datasets and does not involve human subjects or private personal data. By improving inference-time computation allocation, the proposed method may reduce unnecessary resource consumption. Improved computational efficiency does not guarantee the correctness or reliability of model-generated reasoning, which should be independently verified in high-stakes applications.

REPRODUCIBILITY STATEMENT

We specify the models, datasets, baselines, evaluation metrics, and evaluation protocols used in this work. Additional implementation details, including the training procedure, hyperparameters, computation of reference statistics, and inference configurations, are provided in the appendix § A.

REFERENCES

Marah Abdin, Sahaj Agarwal, Ahmed Awadallah, Vidhisha Balachandran, Harkirat Behl, et al. Phi-4-reasoning technical report, 2025. URL https://arxiv.org/abs/2504.21318.

Daman Arora and Andrea Zanette. Training language models to reason efficiently. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=AiZxn84Wdo.

Maciej Besta, Julia Barth, Eric Schreiber, Ales Kubicek, Afonso Catarino, et al. Reasoning language models: A blueprint, 2025. URL https://arxiv.org/abs/2501.11223.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, et al. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, et al. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Xiaoshu Chen, Sihang Zhou, Ke Liang, Xiaoyu Sun, and Xinwang Liu. Skip-thinking: Chunk-wise chain-of-thought distillation enable smaller language models to reason better and faster. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 12153–12168, Suzhou, China, November 2025a. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.610. URL https://aclanthology.org/2025.emnlp-main.610/.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, et al. Do NOT think that much for 2+3=? on the overthinking of long reasoning models. In *Forty-second International Conference on Machine Learning*, 2025b. URL https://openreview.net/forum?id=MSbU3L7V00.

Zhengxiang Cheng, Dongping Chen, Mingyang Fu, and Tianyi Zhou. Optimizing length compression in large reasoning models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3232–3250, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.146. URL https://aclanthology.org/2026.acl-long.146/.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240): 1–113, 2023. URL http://jmlr.org/papers/v24/22-1144.html.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, et al. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities, 2025. URL https://arxiv.org/abs/2507.06261.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. Thinkless: LLM learns when to think. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=ariVQf0KZx.

Sicheng Feng, Gongfan Fang, Xinyin Ma, and Xinchao Wang. Efficient reasoning models: A survey. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=sySqlxj8EB.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, et al. Omni-MATH: A universal olympiad level mathematic benchmark for large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=yaqPf0KAlN.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, et al. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.211. URL https://aclanthology.org/2024.acl-long.211/.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, et al. Measuring mathematical problem solving with the math dataset. In J. Vanschoren and S. Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021b. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/be83ab3ecd0db773eb2dc1b0a17836a1-Paper-round2.pdf.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, et al. An empirical analysis of compute-optimal large language model training. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 30016–30030. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/c1e2faff6f588870935f114ebe04a3e5-Paper-Conference.pdf.

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. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=V51gPu1uQD.

Jian Hu, Jason Klein Liu, Haotian Xu, and Wei Shen. Reinforce++: Stabilizing critic-free policy optimization with global advantage normalization, 2025. URL https://arxiv.org/abs/2501.03262.

Wenyue Hua and Yongfeng Zhang. System 1 + system 2 = better world: Neural-symbolic chain of logic reasoning. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 601–612, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.42. URL https://aclanthology.org/2022.findings-emnlp.42/.

Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 1049–1065, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.67. URL https://aclanthology.org/2023.findings-acl.67/.

Lingjie Jiang, Xun Wu, Shaohan Huang, Qingxiu Dong, Zewen Chi, et al. Think only when you need with large hybrid-reasoning models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=fDjDVE4qdj.

Zixuan Ke, Fangkai Jiao, Yifei Ming, Xuan-Phi Nguyen, Austin Xu, et al. A survey of frontiers in LLM reasoning: Inference scaling, learning to reason, and agentic systems. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=SlsZZ25InC. Survey Certification.

Komal Kumar, Tajamul Ashraf, Omkar Thawakar, Rao Muhammad Anwer, Hisham Cholakkal, Mubarak Shah, Ming-Hsuan Yang, Phillip H.S. Torr, Fahad Shahbaz Khan, and Salman Khan. Llm post-training: A deep dive into reasoning large language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–20, 2026. doi: 10.1109/TPAMI.2026.3720414.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, et al. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, SOSP '23, pp. 611–626, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400702297. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Hynek Kydlíček. Math-Verify: Math Verification Library. URL https://github.com/huggingface/math-verify.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, et al. Tulu 3: Pushing frontiers in open language model post-training. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=i1uGbfHHpH.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, et al. Solving quantitative reasoning problems with language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 3843–3857. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/18abbeef8cfe9203fdf9053c9c4fe191-Paper-Conference.pdf.

Gang Li, Yan Chen, Ming Lin, and Tianbao Yang. DRPO: Efficient reasoning via decoupled reward policy optimization. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=GP5RHZnEsw.

Qintong Li, Leyang Cui, Xueliang Zhao, Lingpeng Kong, and Wei Bi. GSM-plus: A comprehensive benchmark for evaluating the robustness of LLMs as mathematical problem solvers. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2961–2984, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.163. URL https://aclanthology.org/2024.acl-long.163/.

Zhiyuan Li, Yi Chang, and Yuan Wu. Think-bench: Evaluating thinking efficiency and chain-of-thought quality of large reasoning models, 2025. URL https://arxiv.org/abs/2505.22113.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, et al. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Mingjie Liu, et al. Dler: Doing length penalty right - incentivizing more intelligence per token via reinforcement learning, 2025. URL https://arxiv.org/abs/2510.15110.

Wei Liu, Ruochen Zhou, Yiyun Deng, Yuzhen Huang, Junteng Liu, Yuntian Deng, Yizhe Zhang, and Junxian He. Learn to reason efficiently with adaptive length-based reward shaping. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=hj9eKpqxQl.

Ximing Lu, Seungju Han, David Acuna, Hyunwoo Kim, Jaehun Jung, et al. Retro-search: Exploring untaken paths for deeper and efficient reasoning, 2025. URL https://arxiv.org/abs/2504.04383.

Yingqian Min, Zhipeng Chen, Jinhao Jiang, Jie Chen, Jia Deng, et al. Imitate, explore, and self-improve: A reproduction report on slow-thinking reasoning systems, 2024. URL https://arxiv.org/abs/2412.09413.

Team Olmo, :, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, et al. Olmo 3, 2025. URL https://arxiv.org/abs/2512.13961.

OpenAI, :, Aaron Jaech, Adam Kalai, Adam Lerer, et al. Openai o1 system card, 2024. URL https://arxiv.org/abs/2412.16720.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, et al. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, et al. Scaling language models: Methods, analysis & insights from training gopher, 2022. URL https://arxiv.org/abs/2112.11446.

Yi Shen, Jian Zhang, Jieyun Huang, Shuming Shi, Wenjing Zhang, et al. DAST: Difficulty-adaptive slow-thinking for large reasoning models. In Saloni Potdar, Lina Rojas-Barahona, and Sebastien Montella (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 2322–2331, Suzhou (China), November 2025. Association for Computational Linguistics. ISBN 979-8-89176-333-3. doi: 10.18653/v1/2025.emnlp-industry.160. URL https://aclanthology.org/2025.emnlp-industry.160/.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, et al. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, pp. 1279–1297, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400711961. doi: 10.1145/3689031.3696075. URL https://doi.org/10.1145/3689031.3696075.

Yao Shi, Rongkeng Liang, and Yong Xu. EducationQ: Evaluating LLMs' teaching capabilities through multi-agent dialogue framework. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 32799–32828, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1576. URL https://aclanthology.org/2025.acl-long.1576/.

Parshin Shojaee, Seyed Iman Mirzadeh, Keivan Alizadeh, Maxwell Horton, Samy Bengio, and Mehrdad Farajtabar. The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=YghiOusmvw.

Steven A. Sloman. The empirical case for two systems of reasoning. *Psychological Bulletin*, 119 (1):3–22, 1996. doi: 10.1037/0033-2909.119.1.3.

Zayne Rea Sprague, Fangcong Yin, Juan Diego Rodriguez, Dongwei Jiang, Manya Wadhwa, Prasann Singhal, Xinyu Zhao, Xi Ye, Kyle Mahowald, and Greg Durrett. To cot or not to cot? chain-of-thought helps mainly on math and symbolic reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=w6nlcS8Kkn.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, et al. Stop overthinking: A survey on efficient reasoning for large language models. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=HvoG8SxggZ.

Sijun Tan, Michael Luo, Justin Wong, Colin Cai, Xiaoxiang Shi, William Yuan Tang, Manan Roongta, Tianjun Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Effective RL scaling of reasoning models via iterative context lengthening, 2026. URL https://openreview.net/forum?id=I6GzDCne7U.

Yuqiao Tan, Shizhu He, Kang Liu, and Jun Zhao. The zero-step thinking: An empirical study of mode selection as harder early exit in reasoning models. In *NeurIPS 2025 Workshop on Efficient Reasoning*, 2025. URL https://openreview.net/forum?id=CPXmurtK0H.

Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025. URL https://qwenlm.github.io/blog/qwq-32b/.

Guiyao Tie, Zeli Zhao, Dingjie Song, Fuyang Wei, Rong Zhou, et al. A survey on post-training of large language models, 2025. URL https://arxiv.org/abs/2503.06072.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, et al. Llama: Open and efficient foundation language models, 2023. URL https://arxiv.org/abs/2302.13971.

Luong Trung, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. ReFT: Reasoning with reinforced fine-tuning. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7601–7614, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.410. URL https://aclanthology.org/2024.acl-long.410/.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xiong-Hui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 115452–115486. Curran Associates, Inc., 2025a. doi: 10.52202/085713-3850. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/a797c2d2e0c1fdabf4d1ab8cd0b465c6-Paper-Conference.pdf.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 95266–95290. Curran Associates, Inc., 2024. doi: 10.52202/079017-3018.

URL https://proceedings.neurips.cc/paper_files/paper/2024/file/ad236edc564f3e3156e1b2feafb99a24-Paper-Datasets_and_Benchmarks_Track.pdf.

Yue Wang, Qiuzhi Liu, Jiahao Xu, Tian Liang, Xingyu Chen, et al. Thoughts are all over the place: On the underthinking of long reasoning models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=WcUo7Z2Jnh.

Siye Wu, Jian Xie, Yikai Zhang, Aili Chen, Kai Zhang, et al. ARM: Adaptive reasoning model. In *NeurIPS 2025 Workshop on Efficient Reasoning*, 2025. URL https://openreview.net/forum?id=FCcQ7o0WdS.

Fengli Xu, Qianyue Hao, Chenyang Shao, Zefang Zong, Yu Li, et al. Toward large reasoning models: A survey of reinforced reasoning with large language models. *Patterns*, 6(10):101370, 2025. ISSN 2666-3899. doi: https://doi.org/10.1016/j.patter.2025.101370. URL https://www.sciencedirect.com/science/article/pii/S2666389925002181.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, et al. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement, 2024. URL https://arxiv.org/abs/2409.12122.

Jingyang Yi, Justin Wang, and Sida Li. Shorterbetter: Guiding reasoning models to find optimal inference length for efficient reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=MJvwM5dBZM.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 15476–15488. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/639a9a172c044fbb64175b5fad42e9a5-Paper-Conference.pdf.

Boning Zhang, Chengxi Li, and Kai Fan. Mario eval: Evaluate your math llm with your math llm–a mathematical dataset evaluation toolkit, 2024. URL https://arxiv.org/abs/2404.13925.

Duzhen Zhang, Zhong-Zhi Li, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, et al. From system 1 to system 2: A survey of reasoning large language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–20, 2025a. doi: 10.1109/TPAMI.2025.3637037.

Jiajie Zhang, Nianyi Lin, Lei Hou, Ling Feng, and Juanzi Li. AdaptThink: Reasoning models can learn when to think. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 3716–3730, Suzhou, China, November 2025b. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.184. URL https://aclanthology.org/2025.emnlp-main.184/.

Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, et al. A survey of reinforcement learning for large reasoning models, 2025c. URL https://arxiv.org/abs/2509.08827.

A IMPLEMENTATION DETAILS

A.1 REPRODUCIBILITY

All experiments prioritize transparency and reproducibility. Each model is trained once with a fixed seed of 42 and evaluated over five runs using distinct sampling seeds. Hyperparameter configurations (Table 2) follow prior studies on reasoning alignment for fair comparison.

Table 2: **Hyperparameter** for training configuration.

Hyperparameter	1.5B	7B
Optimization		
Total Episodes	40,315	40,315
Total Epochs	1	1
Total Steps	314	628
Batch Size	128	64
Learning Rate	3×10^{-6}	7×10^{-7}
Dynamic Batch	True	False
Generation & Architecture		
Max Prompt Length	1,024	1,024
Max THINK Response Length	16,384	16,384
Max NOTHINK Response Length	4,096	4,096
Reference Sampling (K)	16	16
Importance Sampling (K)	16	16
Temperature	0.6	0.6
Top-p	0.95	0.95
Random Seed	42	42
Weight Type	float32	float32
Reward Configuration (When2Think)		
Reference Scaler (γ)	0.9	0.9
NOTHINK Ratio	0.5	0.5
Bonus (δ)	0.05	0.05

A.2 INFRASTRUCTURE

Training the 1.5B model required approximately 70 GPU-hours on 2× NVIDIA H100 80GB GPUs, while the 7B model required approximately 120 GPU-hours. Evaluation across all reported benchmarks required approximately 20 GPU-hours on 2× NVIDIA A100 80GB GPUs. Training throughput was maximized using the *verl* framework (Sheng et al., 2025) with Distributed Data Parallel (DDP) and FlashAttention-2 integration. Inference and evaluation employed the *vLLM* engine (Kwon et al., 2023), which reduces memory overhead and latency for long chain-of-thought generation. Comprehensive software and hardware specifications are summarized in Table 3.

Table 3: **Computational environment and software stack** used in experiments.

Environment	Train	Test
Hardware Accelerator		
Accelerator	NVIDIA H100 (80GB HBM3)	NVIDIA A100 (80GB)
System Environment		
OS Kernel	Linux-5.15.0-161-generic	Linux-5.4.0-173-generic
Python	CPython-3.11.14	CPython 3.11.14
CUDA	13.0	12.8
Core Frameworks		
Frameworks	PyTorch-2.9.0 Transformers-4.57.1 verl-0.2.0.dev	PyTorch-2.9.0 Transformers-4.57.3 vLLM-0.11.2

B Evaluation Setup

B.1 Inference and Verification Protocol

To ensure a rigorous and reproducible evaluation, we standardize inference-time generation parameters and the answer verification pipeline across all models.

Generation Configuration. We adopt decoding strategies tailored to the characteristics of each model class while keeping them fixed within each category for fair comparison. For general-purpose LLMs, we use a `temperature` of 0.7 and `top-p` of 0.8 to encourage diverse solution exploration. For Large Reasoning Models (LRMs), we apply a more deterministic configuration with a `temperature` of 0.6 and `top-p` of 0.95, which stabilizes long-horizon reasoning while preserving sufficient diversity in intermediate steps. This distinction follows common practice in prior work and accounts for the differing sensitivity of LLMs and LRMs to sampling noise during reasoning, while all generation settings are held constant across benchmarks to ensure fair comparison.

To ensure reliable answer extraction and automated evaluation, all models are instructed to follow a strict output format using the system prompt shown in § B.1, which is instantiated using each model's native chat template.

System Prompt for Mathematical Reasoning

Math assistant. Provide a step-by-step solution in English using LaTeX. Final answer in `\boxed{...}`.

Answer Extraction and Verification. We report `Pass@k` accuracy based on answers extracted from the final `\boxed{...}` tag. To ensure robustness to formatting variations and symbolic ambiguity, we employ a **Dual Math Verifier** pipeline that integrates *Math-Verify* (Kydlíček) and *MARIO Eval* (Zhang et al., 2024). Math-Verify performs primary extraction and symbolic equivalence checking, while MARIO Eval provides complementary semantic verification using type-specific Computer Algebra Systems (CAS). The resulting unified binary correctness signal is used consistently as both the evaluation metric and the terminal reward for policy optimization.

B.2 Dual Math Verifier

Accurate and robust verification is essential for evaluating mathematical reasoning and for constructing reliable reward signals during policy optimization. Because mathematically correct answers can be expressed in multiple symbolic or numerical forms, exact string matching is often insufficient. We therefore employ a dual-verifier pipeline that evaluates semantic equivalence between model-generated answers and ground-truth solutions.

Math-Verify. We use Math-Verify (Kydlíček) as a format-agnostic verifier for mathematical reasoning tasks. Given a model-generated response, Math-Verify extracts the candidate answer using prioritized parsing rules that support diverse output formats, including LaTeX expressions and plain text. The extracted answer is then normalized and converted into a symbolic representation, enabling comparison with the ground-truth solution beyond exact string matching. Verification is performed through symbolic equivalence checking and numerical evaluation with tolerance, allowing equivalent mathematical expressions to be recognized even when their surface forms differ.

MARIO Eval. We additionally adopt MARIO Eval (Zhang et al., 2024) to provide type-aware verification across diverse mathematical answer formats. MARIO Eval first identifies the answer type, such as a scalar, algebraic expression, matrix, equation, or inequality, using rule-based parsing and optional language-model assistance when type inference is ambiguous. It then applies type-specific equivalence checks, primarily relying on symbolic computation through a computer algebra system and numerical evaluation with tolerance. When symbolic verification is inconclusive due to structural ambiguity or natural-language formulations, MARIO Eval can optionally invoke a math-capable language model as a fallback judge.

B.3 BENCHMARKS

We evaluate models on a diverse suite of mathematical reasoning benchmarks spanning foundational arithmetic, intermediate competition problems, olympiad-level challenges, and undergraduate STEM tasks. Together, these benchmarks cover a broad difficulty spectrum, enabling fine-grained analysis of both reasoning accuracy and computational efficiency across problem complexities.

GSM-Plus. GSM-Plus (Li et al., 2024) is a medium-scale benchmark of approximately 10.5k problems that extends GSM-8K (Cobbe et al., 2021) with systematically perturbed variants. It introduces controlled modifications, including paraphrasing, numerical changes, distractor insertion, and reasoning-structure perturbations, enabling evaluation of robustness under increased reasoning complexity and distribution shift. Although GSM-Plus contains a small number of instances annotated with `None` answers for unanswerable or incomplete cases, we exclude them to maintain a consistent answer-extraction and verification protocol. This filtering yields a final evaluation set of approximately 9k problems.

MATH-500. The MATH-500 benchmark (Lightman et al., 2024) is a held-out evaluation subset derived from the original MATH (Hendrycks et al., 2021b) dataset, preserving its structure while avoiding contamination from the PRM-800K (Lightman et al., 2024) training pipeline. The full MATH dataset spans seven subjects–Prealgebra, Algebra, Number Theory, Counting and Probability, Geometry, Intermediate Algebra, and Precalculus–and assigns each problem a difficulty level from 1 to 5 based on AoPS conventions, ranging from introductory AMC 8–style problems to AIME-level items. To prevent overfitting due to inclusion of test-split MATH problems in PRM-800K, 500 uniformly sampled problems were isolated to form MATH-500. This subset retains the subject distribution and difficulty profile of the original test split, providing a leakage-free benchmark for evaluating advanced mathematical reasoning.

Minerva. The Undergraduate-Level STEM Problems, also known as the Minerva benchmark and referred to as OCW Courses in Lewkowycz et al. (2022), consists of 272 self-contained problems collected from MIT OpenCourseWare materials across multiple STEM subjects, including Solid-State Chemistry, Information and Entropy, Differential Equations, and Special Relativity. Each problem provides a clearly delineated final answer and is automatically verifiable, either numerically (191 problems) or symbolically via SymPy (81 problems), emphasizing multi-step reasoning while excluding formal proofs or open-ended short answers. This benchmark is specifically designed to evaluate scientific and quantitative reasoning capabilities in large language models at the undergraduate level, bridging the gap between standard K-12 mathematics benchmarks and more advanced domain-specific problem-solving tasks.

OlympiadBench-Math. OlympiadBench (He et al., 2024) is a high-difficulty, bilingual benchmark comprising 8,476 problems across multiple STEM domains in both multimodal and text-only formats. For evaluation of mathematical reasoning, only text-only math problems are used to isolate multi-step reasoning performance. Each problem includes an expert-verified solution, enabling reliable assessment of model reasoning. Experiments are conducted on a curated subset of 674 open-ended, text-only English problems from the Math Competition track of OlympiadBench.

AIME. The American Invitational Mathematics Examination (AIME) is a selective competition featuring integer-answer problems ranging from 000 to 999. These problems are substantially more challenging than standard AMC questions, requiring multi-step reasoning, careful case analysis, and deeper understanding of algebra, combinatorics, number theory, and geometry. For evaluation, both administrations (AIME I and AIME II) are combined to form a set of 30 problems. The 2024 and 2025 edition is used to avoid overlap with the DeepScaleR dataset, which includes material only through 2023.

B.4 THINK/NOTHINK TRAJECTORY FORMATS

We implement hybrid reasoning using two trajectory formats: THINK and NOTHINK. Let $\langle \mathrm{BOT} \rangle$ and $\langle \mathrm{EOT} \rangle$ denote the backbone-specific begin-of-thinking and end-of-thinking delimiters, respectively. The concrete delimiters and chat-template representation may vary across model families.

A THINK trajectory contains an explicit reasoning block:

$$\textsc{Think} : [x_i] + \langle \mathrm{BOT} \rangle + [\text{reasoning}] + \langle \mathrm{EOT} \rangle + [\text{answer}].$$

A NOTHINK trajectory bypasses the dedicated reasoning block by placing the end-of-thinking delimiter immediately after the begin-of-thinking delimiter:

$$\textsc{NoThink} : [x_i] + \langle \mathrm{BOT} \rangle + \langle \mathrm{EOT} \rangle + [\text{answer}].$$

Thus, NOTHINK is conceptually implemented as an empty reasoning block. In our experiments, the concrete formatting follows the native reasoning template of the DeepSeek-R1 (Guo et al., 2025). For a different backbone, the corresponding model-specific reasoning delimiters or chat-template controls should be used.

During rollout collection, the importance-sampling exploration policy constructs THINK and NOTHINK trajectories with equal probability. After the mode-specific prefix is constructed, all remaining tokens are generated autoregressively by the current policy. At inference time, the exploration policy is removed, and the trained policy directly determines whether to continue generating within the reasoning block or terminate the block and enter the answer channel. Operationally, NOTHINK denotes bypassing the dedicated reasoning block.

B.5 THOUGHTS & STEPS

Lu et al. (2025) propose a hierarchical decomposition of reasoning trajectories by distinguishing between *thoughts* and *steps*.

Given a question $q \in \mathcal{Q}$, a reasoning model $\mathcal{M}$ produces a reasoning trajectory T and a final answer a, denoted as $(T, a) := \mathcal{M}(q)$. The trajectory T is organized as a sequence of higher-level thoughts, each of which consists of multiple intermediate reasoning steps.

Thoughts (s^τ). A reasoning trajectory T is represented as an ordered sequence of thoughts:

$$T := \langle s^1, s^2, \ldots, s^\tau, \ldots, s^m \rangle, \tag{9}$$

where each s^τ corresponds to a contiguous segment of reasoning that follows a coherent line of argument or problem-solving strategy. Transitions between consecutive thoughts, $s^\tau \rightarrow s^{\tau+1}$, are identified using discourse-level linguistic markers (e.g., "Alternatively", "Wait"; see § B.5), which heuristically indicate a shift in reasoning direction.

Discourse Markers Used for Segmenting Thoughts

But, Wait, Alternatively, However, Hmm, Hmmm, Not sure, Going back, Backtrack, Trace back, Another

Steps (s_k^τ). Each thought s^τ is further decomposed into a sequence of intermediate steps:

$$s^\tau := \langle s_1^\tau, s_2^\tau, \ldots, s_{k_\tau}^\tau \rangle. \tag{10}$$

The superscript τ indexes the thought, while the subscript k indexes the step within that thought. In practice, steps correspond to fine-grained inference units and are delimited using formatting cues in the generated trace, such as double newline characters (`\n\n`).

Under this formulation, a reasoning trajectory admits the following hierarchical structure:

$$T = \langle \langle s_1^1, \ldots, s_{k_1}^1 \rangle, \ldots, \langle s_1^\tau, \ldots, s_{k_\tau}^\tau \rangle, \ldots \rangle. \tag{11}$$

This hierarchical decomposition is used solely as an analysis tool to describe the structure of reasoning trajectories.

B.6 ACCURACY-COST EFFICIENCY

We introduce *Accuracy-Cost Efficiency (ACE)*, a metric that measures how efficiently a model's generation leads to a correct final answer under hybrid SYSTEM 1/2 reasoning. ACE extends prior efficiency metrics, including *Outcome Efficiency* (Chen et al., 2025b), *VT* (Cheng et al., 2026), and *Efficiency* (Li et al., 2025), to hybrid reasoning settings. Unlike these measures, ACE accommodates optional or partially expressed reasoning and does not assume that correct answers are embedded within a single continuous reasoning trace.

Let N denote the number of test instances. For each instance i, let $\sigma_i \in \{0, 1\}$ indicate whether the final answer is correct, τ_i denote the total number of generated tokens, and $\hat{\tau}_i$ denote the position of the first token where the correct answer appears. Let $m_i^{\text{Think}} \in \{0, 1\}$ indicate whether the model produces explicit reasoning.

In hybrid reasoning scenarios, the correct answer may not appear explicitly in the reasoning trace or may be produced without any explicit reasoning. In such cases, we set $\hat{\tau}_i = \tau_i$, corresponding to maximal efficiency under ACE, since no redundant reasoning tokens are generated beyond the final answer.

Per-Instance Efficiency. Based on the above definitions, the per-instance efficiency is computed:

$$\eta_i = 1 - m_i^{\text{Think}} \, \mathbb{I}[\hat{\tau}_i < \tau_i] \left(1 - \frac{\hat{\tau}_i}{\tau_i}\right), \tag{12}$$

where:

- $\eta_i = 1$ if either no explicit reasoning is generated ($m_i^{\text{Think}} = 0$) or the correct answer first appears at the end of the trajectory ($\hat{\tau}_i = \tau_i$),
- $\eta_i < 1$ if explicit reasoning continues beyond the first correct token, decreasing proportionally to the fraction of redundant reasoning tokens.

Overall ACE Score. Finally, the overall ACE score is computed:

$$\xi_{\text{ACE}} = \frac{1}{N} \sum_{i=1}^{N} \sigma_i \, \eta_i. \tag{13}$$

This score is bounded, $\xi_{\text{ACE}} \in [0, 1]$, and is used solely as an evaluation metric. Incorrect predictions contribute zero to the score, while correct predictions achieve maximal efficiency if no redundant reasoning occurs. If explicit reasoning continues beyond the earliest correct answer, the score decreases proportionally to the fraction of redundant reasoning tokens, $1 - \hat{\tau}_i / \tau_i$.

C Additional Experimental Results

C.1 Analysis of Token Distribution

Figure 5 shows the distributions of token usage by reasoning mode and response correctness. For correct instances (Left), NoThink peaks sharply at low token counts, reflecting System 1 efficiency, whereas Think consumes more tokens even when correct, revealing overthinking. For incorrect instances (Right), Think exhibits a heavy-tailed distribution reaching the context limit, demonstrating the high cost of failed System 2 reasoning.

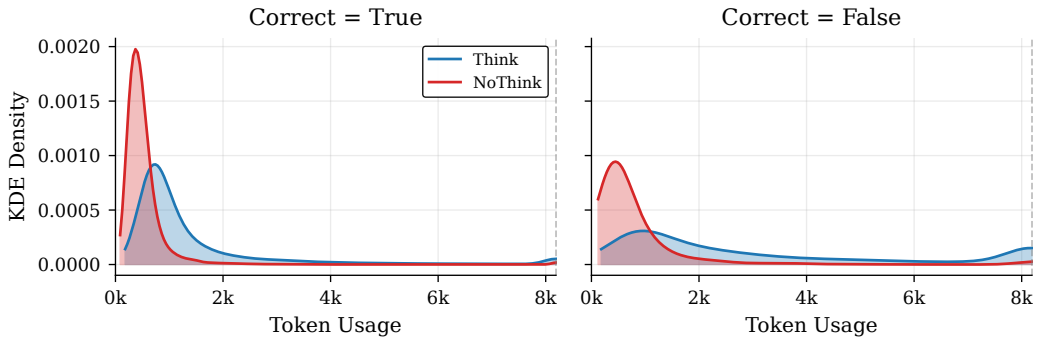


Figure 5: **When2Think token usage distributions by reasoning mode.** Kernel Density Estimation (KDE) of generated token counts for Think (blue) and NoThink (red), stratified by response correctness. **Left:** Correct responses–NoThink peaks at low token usage. **Right:** Incorrect responses–Think exhibits a broader distribution with a long tail, indicating higher token consumption during failed reasoning.

C.2 Reasoning Mode Performance

Table 4: **Detailed Performance by Reasoning Mode on GSM-Plus.** The table reports `Pass@3` accuracy (%), average token usage (↓ `Tokens`), and step counts (↓ `Steps`) for the Think mode, alongside accuracy, token usage, and selection ratio for the NoThink mode. All metrics are reported as mean ± standard deviation over five runs.

Model \ Benchmark	GSM-Plus					
	Think			NoThink		
	↑ Pass@3(%)	↓ Tokens(#)	↓ Steps(#)	↑ Pass@3(%)	↓ Tokens(#)	Ratio(%)
1.5B Models						
AdaptThink-delta0.05	67.4 ± 1.6	1,807 ± 59	10 ± 1	85.9 ± 0.2	561 ± 16	87.5 ± 0.1
Thinkless-RL	84.5 ± 0.2	1,872 ± 30	14 ± 0	93.7 ± 0.8	1,378 ± 62	14.7 ± 0.5
When2Think (Ours)	80.8 ± 0.3	1,401 ± 27	21 ± 1	93.0 ± 0.5	538 ± 15	40.3 ± 0.3

Table 5: **Detailed Performance by Reasoning Mode on Minerva.** The table reports `Pass@3` accuracy (%), average token usage (↓ `Tokens`), and step counts (↓ `Steps`) for the Think mode, alongside accuracy, token usage, and selection ratio for the NoThink mode. All metrics are reported as mean ± standard deviation over five runs.

Model \ Benchmark	Minerva					
	Think			NoThink		
	↑ Pass@3(%)	↓ Tokens(#)	↓ Steps(#)	↑ Pass@3(%)	↓ Tokens(#)	Ratio(%)
1.5B Models						
AdaptThink-delta0.05	36.6 ± 1.6	3,012 ± 285	34 ± 6	57.3 ± 1.6	1,154 ± 96	58.7 ± 2.6
Thinkless-RL	52.5 ± 2.0	4,982 ± 200	74 ± 7	47.4 ± 10.1	6,487 ± 1,603	5.5 ± 1.1
When2Think (Ours)	50.3 ± 2.5	3,282 ± 82	92 ± 3	78.8 ± 4.3	811 ± 192	9.0 ± 0.6

Table 6: **Detailed Performance by Reasoning Mode on OlympiadBench.** The table reports `Pass@3` accuracy (%), average token usage (↓ `Tokens`), and step counts (↓ `Steps`) for the THINK mode, alongside accuracy, token usage, and selection ratio for the NOTHINK mode. All metrics are reported as mean ± standard deviation over five runs.

Model \ Benchmark	OlympiadBench-Math					
	THINK			NOTHINK		
	↑ Pass@3(%)	↓ Tokens(#)	↓ Steps(#)	↑ Pass@3(%)	↓ Tokens(#)	Ratio(%)
1.5B Models						
AdaptThink-delta0.05	41.5 ±2.1	7,223 ±215	69 ±4	62.5 ±1.3	2,003 ±54	80.5 ±0.9
Thinkless-RL	63.6 ±0.9	6,839 ±116	117 ±4	59.4 ±9.0	9,785 ±979	4.0 ±0.7
When2Think (Ours)	57.5 ±1.2	6,635 ±57	204 ±3	90.5 ±2.1	1,086 ±93	14.7 ±0.5

Table 7: **Detailed Performance by Reasoning Mode on AIME24.** The table reports `Pass@3` accuracy (%), average token usage (↓ `Tokens`), and step counts (↓ `Steps`) for the THINK mode, alongside accuracy, token usage, and selection ratio for the NOTHINK mode. All metrics are reported as mean ± standard deviation over five runs.

Model \ Benchmark	AIME24					
	THINK			NOTHINK		
	↑ Pass@3(%)	↓ Tokens(#)	↓ Steps(#)	↑ Pass@3(%)	↓ Tokens(#)	Ratio(%)
1.5B Models						
AdaptThink-delta0.05	44.0 ±11.9	8,121 ±678	338 ±72	47.6 ±12.8	3,227 ±552	48.0 ±11.7
Thinkless-RL	47.7 ±4.6	10,861 ±245	221 ±31	0.0 ±0.0	18,897 ±1,606	2.0 ±1.8
When2Think (Ours)	56.0 ±2.8	10,236 ±438	399 ±27	–	–	0.0 ±0.0

Table 8: **Detailed Performance by Reasoning Mode on AIME25.** The table reports `Pass@3` accuracy (%), average token usage (↓ `Tokens`), and step counts (↓ `Steps`) for the THINK mode, alongside accuracy, token usage, and selection ratio for the NOTHINK mode. All metrics are reported as mean ± standard deviation over five runs.

Model \ Benchmark	AIME25					
	THINK			NOTHINK		
	↑ Pass@3(%)	↓ Tokens(#)	↓ Steps(#)	↑ Pass@3(%)	↓ Tokens(#)	Ratio(%)
1.5B Models						
AdaptThink-delta0.05	29.7 ±12.6	9,114 ±828	345 ±30	32.7 ±3.3	3,000 ±585	36.7 ±4.1
Thinkless-RL	35.2 ±8.9	10,667 ±729	210 ±35	0.0 ±0.0	19,497 ±2,328	4.7 ±5.1
When2Think (Ours)	40.0 ±0.0	9,549 ±719	424 ±65	–	–	0.0 ±0.0

C.3 PERFORMANCE ACROSS BENCHMARKS

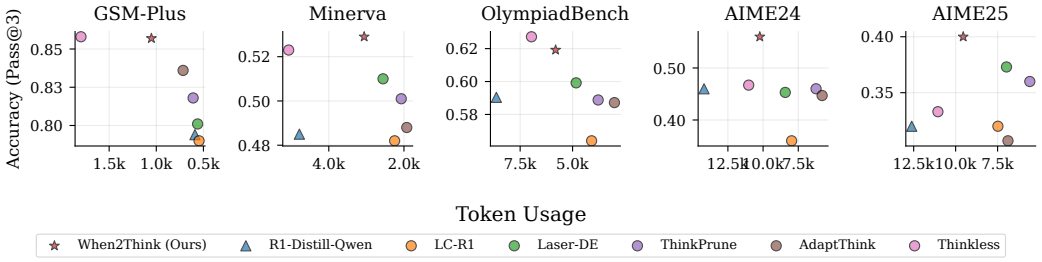


Figure 6: **Accuracy-Efficiency trade-off across reasoning benchmarks.** Performance of ***When2Think*** **(Ours, ⋆)** is compared with the base model **R1-Distill-Qwen (Triangle)** on five benchmarks of varying difficulty: GSM-Plus, Minerva, OlympiadBench, AIME24, and AIME25. The x-axis shows **Token Usage** (reversed; further right indicates fewer tokens/more efficient), and the y-axis shows **Pass@3 Accuracy**. Across benchmarks, *When2Think* points generally occupy the top-right region relative to the baseline, indicating higher accuracy with lower token usage.

Table 9: **Extended performance comparison on reasoning benchmarks.** The table reports `Pass@3` accuracy (%), average tokens per response (↓ `Tokens`), and Accuracy-Cost Efficiency (↑ `ACE`) on GSM-Plus, Minerva, OlympiadBench-Math, AIME24, and AIME25 under a unified evaluation protocol. Results are obtained using a sampling-based `Pass@k` procedure with matched decoding settings and are reported as mean ± standard deviation over five runs. Values in parentheses indicate differences relative to R1-Distill-Qwen (green: improvement; red: degradation).

Model \ Benchmark	GSM-Plus			Minerva			OlympiadBench-Math			AIME24			AIME25		
	↑ Pass@3(%)	↓ Tokens(#)	↑ ACE(%)	↑ Pass@3(%)	↓ Tokens(#)	↑ ACE(%)	↑ Pass@3(%)	↓ Tokens(#)	↑ ACE(%)	↑ Pass@3(%)	↓ Tokens(#)	↑ ACE(%)	↑ Pass@3(%)	↓ Tokens(#)	↑ ACE(%)
1.5B Models															
Qwen2.5-Instruct	73.6 ±0.3	346 ±2.9	–	30.1 ±1.5	716 ±53.2	–	29.6 ±0.9	976 ±31.0	–	8.0 ±3.8	1,373 ±343.1	–	1.3 ±3.0	995 ±163.4	–
Qwen2.5-Math-Instruct	84.3 ±0.3	360 ±1.8	–	44.8 ±0.5	671 ±14.7	–	53.2 ±0.8	842 ±8.3	–	22.7 ±1.5	1,057 ±57.7	–	19.3 ±2.8	923 ±64.8	–
R1-Distill-Qwen (Base)	79.4 ±0.3	590 ±11.6	79.2 ±0.3	48.5 ±1.9	4,782 ±124.1	36.4 ±2.1	58.8 ±2.0	8,634 ±107.4	39.6 ±1.6	46.0 ±2.8	14,195 ±1,026.7	26.2 ±3.7	32.0 ±5.1	12,616 ±688.4	17.7 ±3.1
DeepScaleR-Preview	85.4 ±0.3	1,358 ±13.4	83.7 ±0.1	55.4 ±1.1	4,130 ±56.4	43.7 ±1.0	62.8 ±0.8	5,215 ±43.3	42.9 ±0.5	58.0 ±5.6	8,473 ±479.5	38.4 ±4.4	39.3 ±6.8	8,074 ±288.6	27.0 ±5.7
LC-R1	79.0 ±0.3	546 ±9.9	79.0 ±0.3	48.2 ±1.7	2,248 ±71.8	42.7 ±1.8	55.5 ±1.1	4,092 ±113.0	39.9 ±1.1	36.0 ±4.9	7,972 ±1,024.3	23.0 ±3.4	32.0 ±3.8	6,971 ±278.7	18.5 ±2.7
	(- 0.4)	(- 44)	(- 0.2)	(- 0.3)	(- 2,534)	(+ 0.3)	(- 3.3)	(- 4,542)	(+ 0.3)	(- 10.0)	(- 6,223)	(- 3.2)	(± 0)	(- 5,645)	(+ 0.8)
ThinkPrune-iter3k	81.8 ±0.3	609 ±5.5	79.3 ±0.3	50.1 ±1.9	2,072 ±62.2	32.9 ±1.5	58.6 ±0.7	3,781 ±66.5	35.7 ±0.6	46.0 ±4.3	6,236 ±555.4	27.0 ±2.0	36.0 ±3.7	5,582 ±273.3	21.5 ±1.9
	(+ 2.4)	(+ 19)	(+ 0.1)	(+ 1.6)	(- 2,710)	(- 3.5)	(+ 0.2)	(- 4,853)	(- 3.9)	(± 0)	(- 7,959)	(+ 0.8)	(+ 4.0)	(- 7,034)	(+ 3.8)
Laser-DE-L4096	80.1 ±0.2	562 ±4.1	79.4 ±0.3	51.0 ±1.2	2,556 ±105.5	38.7 ±0.7	59.9 ±1.1	4,830 ±43.4	39.1 ±1.0	45.3 ±3.8	8,425 ±392.3	27.0 ±2.6	37.3 ±8.9	6,971 ±1,024.3	24.1 ±6.6
	(+ 0.7)	(- 28)	(+ 0.2)	(+ 2.5)	(- 2,226)	(+ 2.3)	(+ 1.1)	(- 3,804)	(- 0.5)	(- 0.7)	(- 5,770)	(+ 0.8)	(+ 5.3)	(- 5,645)	(+ 6.4)
AdaptThink-delta0.05	83.6 ±0.2	716 ±17.5	83.5 ±0.2	48.8 ±1.3	1,925 ±110.9	48.0 ±1.5	58.4 ±0.8	3,004 ±76.2	56.9 ±0.9	44.7 ±4.5	5,806 ±438.8	40.1 ±4.9	30.7 ±8.0	6,883 ±440.2	25.1 ±5.7
	(+ 3.2)	(+ 126)	(+ 4.3)	(+ 0.3)	(- 2,857)	(+ 11.6)	(- 0.4)	(- 5,630)	(+ 17.3)	(- 1.3)	(- 8,389)	(+ 13.9)	(- 1.3)	(- 5,733)	(+ 7.4)
Thinkless-RL	85.8 ±0.2	1,799 ±16.2	82.7 ±0.2	52.3 ±1.4	5,067 ±166.0	44.3 ±1.4	63.4 ±1.1	6,960 ±76.2	47.5 ±1.5	46.7 ±4.1	11,023 ±209.4	30.4 ±4.4	33.3 ±7.5	11,056 ±641.7	21.4 ±4.7
	(+ 6.4)	(+ 1,209)	(+ 3.5)	(+ 3.8)	(+ 285)	(+ 7.9)	(+ 4.6)	(- 1,674)	(+ 7.9)	(+ 0.7)	(- 3,172)	(+ 4.2)	(+ 1.3)	(- 1,560)	(+ 3.7)
When2Think (Ours)	85.7 ±0.3	1,052 ±17.9	80.7 ±0.4	52.9 ±2.1	3,058 ±58.6	40.6 ±1.7	62.4 ±0.9	5,909 ±56.4	43.9 ±1.2	56.0 ±2.8	10,236 ±438.0	32.1 ±1.8	40.0 ±0.0	9,549 ±719.4	24.6 ±1.1
	(+ 0.3)	(+ 462)	(+ 1.5)	(+ 4.4)	(- 1,724)	(+ 4.2)	(+ 3.6)	(- 2,725)	(+ 4.3)	(+ 10.0)	(- 3,959)	(+ 5.9)	(+ 8.0)	(- 3,067)	(+ 6.9)
7B Models															
Qwen2.5-Instruct	88.8 ±0.2	342 ±0.2	–	54.0 ±1.4	634 ±1.4	–	52.3 ±1.4	819 ±11.0	–	18.0 ±3.8	1,087 ±107.0	–	20.7 ±7.2	937 ±13.5	–
Qwen2.5-Math-Instruct	89.1 ±0.2	421 ±3.1	–	51.6 ±1.2	833 ±11.3	–	55.8 ±0.5	1,110 ±13.4	–	24.0 ±3.8	1,614 ±140.6	–	22.0 ±4.5	1,512 ±71.5	–
R1-Distill-Qwen	87.6 ±0.1	521 ±1.1	87.5 ±0.1	63.6 ±1.1	3,517 ±99.5	50.0 ±1.7	65.6 ±0.5	4,984 ±65.0	49.3 ±1.1	60.0 ±8.5	7,266 ±504.0	39.3 ±6.3	43.3 ±4.7	6,611 ±473.3	29.1 ±4.6
Laser-DE-L4096	90.4 ±0.1	849 ±1.3	87.0 ±0.2	61.9 ±1.0	1,767 ±29.4	51.8 ±1.0	71.1 ±0.6	3,358 ±21.7	46.8 ±0.6	66.7 ±2.4	5,563 ±283.0	42.2 ±2.2	52.0 ±5.6	5,553 ±176.2	35.0 ±4.1
	(+ 2.8)	(+ 328)	(- 0.5)	(- 1.7)	(- 1,750)	(+ 1.8)	(+ 5.5)	(- 1,626)	(- 2.5)	(+ 6.7)	(- 1,703)	(+ 2.9)	(+ 8.7)	(- 1,058)	(+ 5.9)
AdaptThink-delta0.05	88.8 ±0.2	394 ±2.0	88.8 ±0.2	62.0 ±1.6	1,983 ±67.2	56.6 ±2.1	67.2 ±1.0	4,080 ±79.7	56.4 ±0.6	68.0 ±3.8	6,682 ±762.7	48.8 ±2.2	46.7 ±2.4	6,599 ±108.6	33.8 ±2.3
	(+ 1.2)	(- 127)	(+ 1.3)	(- 1.6)	(- 1,534)	(+ 6.6)	(+ 1.6)	(- 904)	(+ 7.1)	(+ 8.0)	(- 584)	(+ 9.5)	(+ 3.4)	(- 12)	(+ 1.8)
When2Think (Ours)	89.3 ±0.1	460 ±2.3	88.7 ±0.1	63.2 ±1.1	2,451 ±47.5	54.1 ±0.8	69.1 ±0.7	4,250 ±70.6	55.4 ±0.8	68.7 ±4.5	7,855 ±397.9	45.4 ±3.7	48.0 ±1.8	7,347 ±799.1	31.6 ±1.4
	(+ 1.7)	(- 61)	(+ 1.2)	(- 0.4)	(- 1,066)	(+ 4.1)	(+ 3.5)	(- 734)	(+ 6.1)	(+ 8.7)	(+ 589)	(+ 6.1)	(+ 4.7)	(+ 736)	(+ 2.5)

Table 10: **Performance comparison on MATH-500 stratified by problem difficulty (Levels 1–5).** The table reports `Pass@1` accuracy (%), average tokens per response (↓ `Tokens`), and Accuracy–Cost Efficiency (↑ `ACE`) for each difficulty level under the same evaluation protocol as the main results. Results are reported as mean ± standard deviation over five runs. Values in parentheses indicate differences relative to R1-Distill-Qwen (green: improvement; red: degradation).

Model \ Benchmark	MATH-500															
	Level 1			Level 2			Level 3			Level 4			Level 5			
	↑ Pass@1(%)	↓ Tokens(#)	↑ ACE(%)	↑ Pass@1(%)	↓ Tokens(#)	↑ ACE(%)	↑ Pass@1(%)	↓ Tokens(#)	↑ ACE(%)	↑ Pass@1(%)	↓ Tokens(#)	↑ ACE(%)	↑ Pass@1(%)	↓ Tokens(#)	↑ ACE(%)
1.5B Models																
Qwen2.5-Instruct	79.5 ±0.9	342 ±11.9	–	71.1 ±4.2	429 ±15.8	–	57.0 ±1.4	768 ±144.7	–	37.5 ±4.0	828 ±213.7	–	21.3 ±2.9	1,037 ±113.9	–	
Qwen2.5-Math-Instruct	94.0 ±1.3	346 ±6.1	–	90.2 ±0.5	422 ±4.4	–	87.8 ±3.0	499 ±13.7	–	71.2 ±2.3	644 ±26.0	–	51.8 ±3.6	774 ±38.5	–	
R1-Distill-Qwen (Base)	92.1 ±2.7	1,199 ±150.9	89.3 ±2.1	91.8 ±2.9	1,724 ±136.2	83.7 ±2.9	85.7 ±2.7	2,530 ±64.1	74.7 ±3.3	79.4 ±2.0	4,006 ±197.3	65.0 ±3.6	71.0 ±2.3	6,830 ±208.3	56.6 ±2.3	
DeepScaleR-Preview	94.0 ±2.1	1,250 ±62.6	86.3 ±2.5	95.8 ±0.5	1,533 ±48.4	85.2 ±0.6	93.9 ±1.4	2,135 ±94.4	78.3 ±1.7	87.7 ±2.0	2,913 ±138.3	71.5 ±1.0	75.8 ±2.1	4,703 ±115.7	62.3 ±1.9	
LC-R1	90.2 ±3.8	626 ±23.2	85.6 ±3.5	85.8 ±4.0	916 ±74.5	78.9 ±3.4	81.9 ±1.8	1,491 ±116.7	71.0 ±2.5	77.3 ±3.5	2,265 ±202.1	66.0 ±4.2	59.3 ±2.4	3,880 ±313.9	47.3 ±2.0	
	(- 1.9)	(- 573)	(- 3.7)	(- 6.0)	(- 808)	(- 4.8)	(- 3.8)	(- 1,039)	(- 3.7)	(- 2.1)	(- 1,741)	(+ 1.0)	(- 11.7)	(- 2,950)	(- 9.3)	
ThinkPrune-iter3k	93.0 ±5.5	770 ±20.3	80.9 ±5.5	91.3 ±2.9	990 ±27.3	72.8 ±4.0	90.3 ±3.4	1,421 ±68.5	67.9 ±3.4	83.3 ±4.0	2,106 ±138.9	60.9 ±3.0	66.6 ±4.3	3,068 ±68.4	48.5 ±3.1	
	(+ 0.9)	(- 429)	(- 8.9)	(- 0.5)	(- 734)	(- 10.9)	(+ 4.6)	(- 1,109)	(- 6.8)	(+ 3.9)	(- 1,900)	(- 4.1)	(- 4.4)	(- 3,762)	(- 8.1)	
Laser-DE-L4096	91.2 ±1.9	898 ±68.1	82.7 ±2.7	90.2 ±1.8	1,211 ±23.1	78.1 ±2.2	92.2 ±2.1	1,763 ±59.8	72.0 ±2.6	81.6 ±0.4	2,551 ±133.8	63.0 ±1.4	69.6 ±3.3	4,056 ±90.1	53.7 ±2.3	
	(- 0.9)	(- 301)	(- 6.6)	(- 1.6)	(- 513)	(- 5.6)	(+ 6.5)	(- 767)	(- 2.7)	(+ 2.2)	(- 1,455)	(- 2.0)	(- 1.4)	(- 2,774)	(- 2.9)	
AdaptThink-delta0.05	94.4 ±2.1	512 ±156.9	94.4 ±2.1	92.0 ±1.4	510 ±29.7	91.8 ±1.6	89.3 ±1.2	1,029 ±87.5	88.0 ±1.1	80.0 ±3.2	1,477 ±63.3	79.2 ±3.2	65.2 ±3.2	2,525 ±112.2	63.9 ±2.7	
	(+ 2.3)	(- 687)	(+ 5.1)	(+ 0.2)	(- 1,214)	(+ 8.1)	(+ 3.6)	(- 1,501)	(+ 13.3)	(- 0.6)	(- 2,529)	(+ 14.2)	(- 5.8)	(- 4,305)	(+ 7.3)	
Thinkless-RL	95.8 ±3.4	1,374 ±196.8	90.8 ±4.1	94.7 ±2.7	1,807 ±146.3	89.2 ±2.3	91.6 ±2.1	2,669 ±218.2	82.3 ±3.1	82.2 ±3.2	3,837 ±221.7	71.2 ±1.8	71.3 ±3.4	5,668 ±146.3	58.2 ±3.1	
	(+ 3.7)	(+ 175)	(+ 1.5)	(+ 2.9)	(+ 143)	(+ 5.5)	(+ 5.9)	(+ 139)	(+ 7.6)	(+ 2.8)	(- 169)	(+ 6.2)	(+ 0.3)	(- 1,102)	(+ 1.6)	
When2Think (Ours)	95.8 ±1.9	619 ±11.0	94.3 ±1.9	92.9 ±2.6	862 ±76.1	89.2 ±2.4	90.5 ±0.7	1,447 ±42.6	82.8 ±0.6	84.5 ±2.2	2,277 ±157.7	76.6 ±1.9	70.7 ±2.9	4,554 ±177.5	60.3 ±2.0	
	(+ 3.7)	(- 580)	(+ 5.0)	(+ 1.1)	(- 862)	(+ 5.5)	(+ 4.8)	(- 1,083)	(+ 8.1)	(+ 5.1)	(- 1,729)	(+ 11.6)	(- 0.3)	(- 2,276)	(+ 3.7)	
7B Models																
Qwen2.5-Instruct	96.3 ±1.3	342 ±4.9	–	92.4 ±1.4	440 ±4.2	–	87.4 ±2.8	506 ±7.4	–	72.7 ±3.0	623 ±8.6	–	53.9 ±3.9	775 ±7.1	–	
Qwen2.5-Math-Instruct	96.7 ±2.7	370 ±16.9	–	93.8 ±1.0	449 ±7.7	–	93.9 ±1.1	571 ±18.8	–	81.6 ±0.9	767 ±13.6	–	64.6 ±2.7	1,013 ±44.0	–	
R1-Distill-Qwen	93.5 ±1.0	948 ±83.4	90.1 ±2.2	93.3 ±1.1	1,306 ±33.5	86.5 ±1.2	95.6 ±1.6	1,815 ±80.9	84.1 ±2.3	90.5 ±1.7	2,800 ±24.8	77.7 ±1.4	84.8 ±1.5	4,449 ±163.9	71.7 ±2.0	
Laser-DE-L4096	96.7 ±1.3	953 ±10.9	82.9 ±1.5	97.3 ±1.3	1,080 ±21.8	78.4 ±0.9	98.9 ±1.0	1,326 ±32.3	75.7 ±1.5	94.4 ±1.0	1,782 ±41.2	71.7 ±0.6	86.9 ±1.1	2,717 ±23.4	66.3 ±1.1	
	(+ 3.2)	(+ 5)	(- 7.2)	(+ 4.0)	(- 226)	(- 8.1)	(+ 3.3)	(- 489)	(- 10.4)	(+ 3.9)	(- 1,018)	(- 6.0)	(+ 2.1)	(- 1,732)	(- 5.4)	
AdaptThink-delta0.05	95.3 ±1.6	420 ±12.7	95.2 ±1.7	95.6 ±1.8	568 ±16.8	95.1 ±1.7	97.5 ±1.1	929 ±24.6	95.2 ±1.8	92.0 ±1.7	1,416 ±63.8	89.0 ±1.6	85.2 ±1.8	3,186 ±171.3	78.8 ±1.8	
	(+ 1.8)	(- 528)	(+ 5.1)	(+ 2.3)	(- 738)	(+ 8.6)	(+ 1.9)	(- 886)	(+ 11.1)	(+ 1.5)	(- 1,384)	(+ 11.3)	(+ 0.4)	(- 1,263)	(+ 7.1)	
When2Think (Ours)	96.7 ±2.1	603 ±23.9	95.7 ±2.3	98.7 ±0.9	904 ±33.8	93.1 ±1.2	97.0 ±1.4	1,247 ±67.0	91.4 ±1.8	93.4 ±1.2	2,082 ±58.8	83.2 ±1.0	87.6 ±2.0	3,825 ±101.6	76.9 ±1.4	
	(+ 3.2)	(- 345)	(+ 5.6)	(+ 5.4)	(- 402)	(+ 6.6)	(+ 1.4)	(- 568)	(+ 7.3)	(+ 2.9)	(- 718)	(+ 5.5)	(+ 2.8)	(- 624)	(+ 5.2)	

C.4 Cross-Domain Transfer on MMLU-Pro Stratified

MMLU-Pro (Wang et al., 2024) extends MMLU (Hendrycks et al., 2021a) by filtering trivial or noisy questions, incorporating more reasoning-intensive problems, and expanding the answer set from four to typically ten options. We evaluate on MMLU-Pro Stratified (Shi et al., 2025), a balanced 1,300-question subset spanning 13 disciplines and 10 empirical difficulty levels. Each discipline–difficulty combination contains ten questions, and the difficulty levels are derived from mean item-level accuracy across ten strong models rather than from native MMLU-Pro annotations. We aggregate the levels into Easy (Levels 1–3), Medium (Levels 4–7), and Hard (Levels 8–10). The 13 disciplines are grouped into four supercategories: STEM (Biology, Chemistry, Computer Science, Engineering, Mathematics, and Physics), Humanities (History and Philosophy), Social Science (Economics and Psychology), and Professional disciplines (Business, Health, and Law).

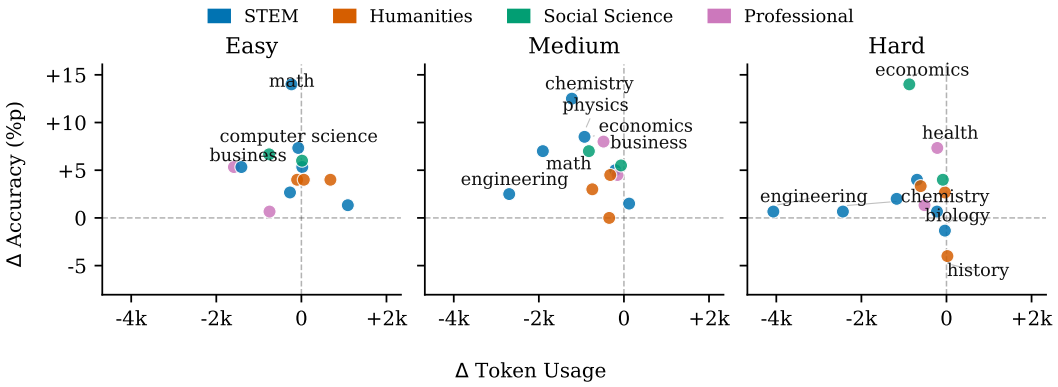


Figure 7: **Discipline-level accuracy and computation changes relative to R1-Distill on MMLU-Pro Stratified.** Each point represents one discipline, colored by supercategory, and the three panels correspond to the Easy, Medium, and Hard groups. The horizontal axis shows the change in average token usage, and the vertical axis shows the change in Pass@3 accuracy. The upper-left quadrant indicates simultaneous accuracy improvement and computation reduction. Accuracy gains and token reductions exhibit heterogeneous patterns across disciplines and difficulty groups.

Cross-domain transfer. Relative to R1-Distill, When2Think improves macro-average accuracy by 5.1, 5.3, and 2.7 percentage points on the Easy, Medium, and Hard groups, respectively (Table 11). Average token usage decreases by 255, 752, and 842 tokens, corresponding to relative reductions of 8.1%, 14.8%, and 16.0%. The corresponding changes in THINK ratio are -4.0, -1.3, and -0.4 percentage points. At the discipline level, When2Think improves Easy-set accuracy in all 13 disciplines and simultaneously improves accuracy and reduces token usage on the Hard set in 11 of 13 disciplines (Figure 7). The form of transfer nevertheless varies across disciplines. On the Hard set, Economics exhibits the largest accuracy gain at +14.0 percentage points, whereas the largest absolute token reductions occur in Engineering (-4,070), Chemistry (-2,440), Mathematics (-1,170), and Computer Science (-690). Token reduction is limited in Physics, while History shows no simultaneous Hard-set improvement in accuracy and computation. Thus, accuracy gains and computation reductions do not follow the same disciplinary distribution.

Difficulty-conditioned behavior. The absolute behavior of When2Think also varies with empirical difficulty (Table 12). Macro-average Pass@3 decreases from 54.9% on Easy questions to 41.6% and 29.6% on Medium and Hard questions, respectively. Over the same groups, average token usage increases from 2,891 to 4,343 and 4,421 tokens, while the THINK ratio increases from 94.6% to 96.4% and 97.1%. Thus, increasing difficulty is accompanied by greater computation and more frequent use of explicit reasoning.

Mode selection versus computation allocation. Reasoning-mode selection and computation allocation exhibit distinct cross-domain patterns. Direct answering transfers most strongly to Mathematics, where the THINK ratio is 64.0%, 64.0%, and 70.0% on Easy, Medium, and Hard questions, respectively (Table 12). Despite identical THINK ratios on Easy and Medium questions, average token usage increases from 1,721 to 3,602 tokens, showing that mode frequency alone does not determine computation allocation. Outside Mathematics, explicit reasoning remains predominant, with non-Math average THINK ratios of 97.2%, 99.0%, and 99.3% across the three groups. Several disciplines, including Economics, Health, History, Law, and Philosophy, retain a 100% THINK ratio while exhibiting substantial variation in token usage.

The baseline-relative comparison reinforces this distinction. On Hard questions, the macro-average THINK ratio changes by only -0.4 percentage points, while average token usage decreases by 842 tokens (Table 11). The Hard-set computation reduction therefore is not primarily accompanied by increased NOTHINK selection. The large token reductions in Engineering and Chemistry likewise occur despite THINK ratios at or near 100% (Figure 7, Table 12). Together, these observations are consistent with domain-selective transfer of direct answering and broader transfer of computation-allocation behavior. Because IDAC, BWS, and hybrid exploration are trained jointly, this interpretation constitutes behavioral attribution rather than a controlled causal decomposition of individual components.

Table 11: **Macro-average cross-domain transfer relative to R1-Distill on MMLU-Pro Stratified.** Accuracy and THINK-ratio changes are reported in percentage points, while token changes are reported in tokens per response. Relative token changes are computed with respect to R1-Distill.

Metric	Easy	Medium	Hard
Δ Accuracy (%p)	$+5.1$	$+5.3$	$+2.7$
Δ Tokens	-255	-752	-842
Relative token change	-8.1%	-14.8%	-16.0%
Δ THINK ratio (%p)	-4.0	-1.3	-0.4

Table 12: **Discipline-level behavior of When2Think on MMLU-Pro Stratified.** Disciplines are grouped into four supercategories. Pass@3 accuracy, average token usage, and THINK ratio are reported for the Easy, Medium, and Hard groups. Macro Avg. includes all 13 disciplines, while Non-Math Avg. excludes Mathematics.

		Pass@3 (%)			Tokens			THINK ratio (%)		
Supercategory	Discipline	Easy	Medium	Hard	Easy	Medium	Hard	Easy	Medium	Hard
STEM	Biology	41.3	35.0	36.0	3269	4819	4267	99.3	99.5	99.3
	Chemistry	71.3	51.0	35.3	4741	8755	7496	94.0	97.0	96.7
	Computer Science	75.3	40.0	32.0	2472	4782	4851	88.0	96.0	100.0
	Engineering	65.3	35.5	22.7	6656	10360	10765	97.3	99.5	100.0
	Mathematics	**90.0**	**79.5**	**37.3**	**1721**	**3602**	**5379**	**64.0**	**64.0**	**70.0**
	Physics	72.7	58.0	36.0	5775	6870	7872	92.7	99.5	99.3
Humanities	History	31.3	24.0	20.0	2009	1298	2202	100.0	100.0	100.0
	Philosophy	39.3	33.5	21.3	1425	2610	1764	100.0	100.0	100.0
Social Science	Economics	59.3	39.0	38.0	2458	1972	1913	100.0	100.0	100.0
	Psychology	41.3	39.5	29.3	820	1446	1338	100.0	98.5	100.0
Professional	Business	57.3	44.0	24.0	3205	6905	6084	94.7	98.5	96.7
	Health	36.0	33.5	30.0	1279	1526	1719	100.0	100.0	100.0
	Law	32.7	28.0	23.3	1752	1512	1827	100.0	100.0	100.0
Macro Avg.		**54.9**	**41.6**	**29.6**	**2891**	**4343**	**4421**	**94.6**	**96.4**	**97.1**
Non-Math Avg.		**51.9**	**38.4**	**29.0**	**2988**	**4405**	**4342**	**97.2**	**99.0**	**99.3**

D Ablation Studies

D.1 Reward and Advantage Landscapes under IDAC and BWS

BWS versus GRPO in the Advantage Landscape. We analyze why Batch-Wise Standardization (BWS) is critical for optimizing IDAC by comparing three reward formulations: the raw THINK reward, BWS advantages, and GRPO-style group-relative advantages. We construct a Monte Carlo simulation using offline statistics from R1-Distill-Qwen-1.5B on DeepScaleR. For each problem, empirical accuracy α captures difficulty and reference length τ approximates typical reasoning cost. We then generate THINK trajectories by perturbing τ with log-normal noise and sampling correctness according to α. This setup isolates how different normalization schemes allocate credit to reasoning trajectories as a function of problem difficulty and token usage.

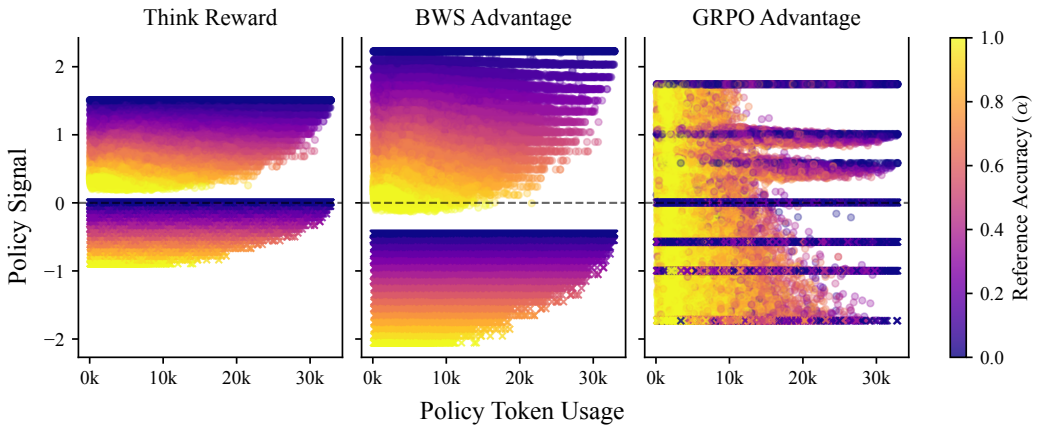


Figure 8: **Empirically Grounded Advantage Landscape of Explicit Reasoning.** Policy signals for simulated THINK trajectories constructed from R1-Distill-Qwen-1.5B statistics on DeepScaleR. Colors represent reference accuracy α (hard problems in blue, $\alpha \to 0$, to easy problems in yellow, $\alpha \to 1$), and markers indicate correctness (circles for correct, crosses for incorrect). **Left:** Raw IDAC THINK reward with absolute length-sensitive bonus. **Middle:** BWS advantage (ours), which preserves positive advantages for long successful trajectories on hard problems. **Right:** GRPO-style advantage, where problem instance normalization yields fragmented signals and horizontal banding.

BWS: Preserving Global Difficulty Signals. BWS converts raw IDAC rewards into relative advantages computed over the full mini-batch, rather than normalizing within each instance. This batch-wise formulation preserves cross-instance difficulty structure that is lost under instance-local normalization. As a result, trajectories that solve difficult problems can remain above the batch expectation and receive positive advantages, even when their absolute rewards are reduced. As shown in Figure 8, this effect is most evident for low-α instances, where successful long trajectories retain positive optimization signals despite high token usage. Thus, BWS does not merely reduce variance; it preserves global difficulty signals in the advantage function, enabling credit assignment for deliberate but necessary long-horizon reasoning.

GRPO: Removing Cross-Instance Reward Structure. In contrast, GRPO-style normalization computes advantages relative only to rollouts of the same problem, removing any cross-instance reward comparisons. While this eliminates instance-level scale variation, it also discards the global difficulty structure encoded in IDAC. As a consequence, the advantage landscape becomes fragmented, manifesting as horizontal bands in Figure 8. Because normalization is restricted to within-problem rollouts, credit assignment becomes sensitive to the composition of sampled trajectories. In particular, long but successful reasoning traces on difficult problems may receive weak or unstable signals when competing against short incorrect or prematurely terminated rollouts. These results highlight a key limitation of instance-local normalization, and motivate BWS as a mechanism for preserving cross-instance difficulty structure in critic-free optimization.

D.2 Analysis of BWS Training Dynamics

We analyze training dynamics to evaluate the effectiveness of our optimization strategy, focusing on the role of *Batch-Wise Standardization* (BWS). Figure 9 reports both task performance and reasoning behavior throughout post-training.

Improved Convergence and Stability. BWS substantially stabilizes policy optimization. Without BWS, raw reward signals exhibit high variance across instances, leading to noisy gradient estimates and unstable learning trajectories. By normalizing rewards using batch-level statistics (μ_k, σ_k), BWS reduces gradient variance and disentangles instance difficulty from policy quality in the optimization signal. This effect is particularly pronounced on AIME24, where problem difficulty varies sharply; under BWS, the model consistently improves performance even on the hardest instances.

Controlled Reasoning Depth. BWS also plays a critical role in regulating reasoning length. Without proper normalization, baseline policies tend to drift toward degenerate behaviors–either collapsing to minimal reasoning or producing excessively verbose outputs–due to unstable credit assignment. In contrast, *When2Think* maintains a stable *Think Ratio* and *Response Length*, gradually adapting reasoning depth to optimize the accuracy–efficiency trade-off. Notably, this behavior emerges without relying on an explicit value function, indicating that BWS effectively stabilizes critic-free policy optimization.

Sample Efficiency and Difficulty Scaling. Our approach exhibits improved sample efficiency. On AIME24, *When2Think* reaches peak performance in approximately half the training steps required by the baseline. Moreover, the performance gap widens as task difficulty increases, highlighting robustness under high-entropy reasoning regimes. Finally, BWS decouples response length from accuracy, preventing the reasoning bloat observed in the baseline, where additional tokens fail to yield meaningful performance gains.

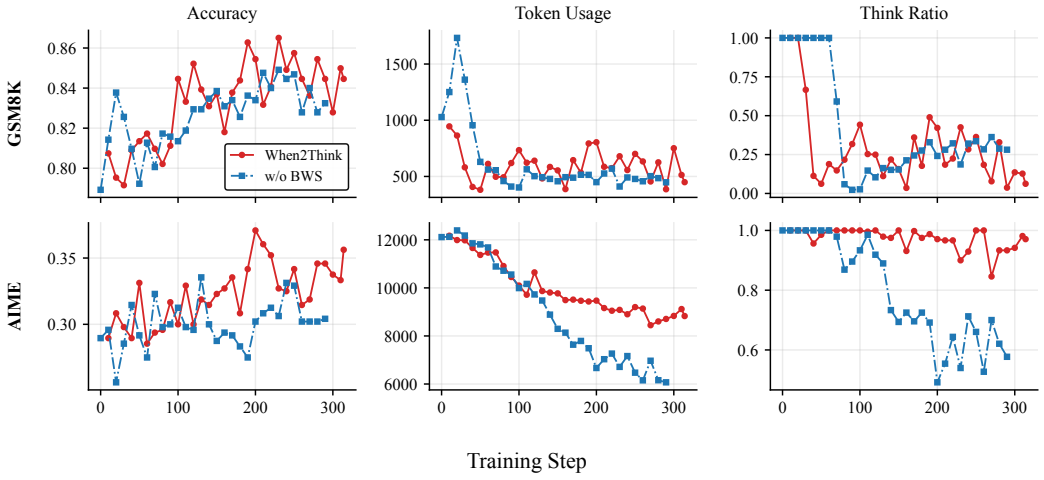


Figure 9: **Training Dynamics under Batch-Wise Standardization Ablations.** *When2Think* (red, with batch-wise standardized advantage) and *Baseline* (blue, without standardization) on DeepScaleR. **Top:** GSM8K (easy). **Bottom:** AIME24 (hard). Each row reports validation accuracy, average token usage, and Think Ratio over training steps.

D.3 Analysis of IS & IDAC Training Dynamics

We further analyze the roles of Importance Sampling (IS) and Instance-level Difficulty-Aware Control (IDAC) during training. We compare three variants: the full *When2Think* model, a variant without IS (retaining IDAC and BWS), and a variant without IDAC (retaining IS and BWS). This ablation isolates whether the observed accuracy–efficiency trade-offs are driven by balanced THINK/NOTHINK exploration (via IS) or by difficulty-aware reward shaping and normalized optimization signals (via IDAC and BWS).

IDAC and BWS provide the main depth-control signal. The *IDAC+BWS* variant removes the balanced exploration distribution induced by IS, while retaining difficulty-aware reward shaping and batch-wise standardized advantages. As shown in Figure 10, this variant remains competitive in validation accuracy, especially on the harder AIME24 benchmark. This suggests that the main depth-control signal is primarily driven by IDAC-based reward shaping together with BWS-based advantage standardization, rather than by Importance Sampling alone.

IS shifts the policy toward a more efficient hybrid operating point. Although IS is not the primary source of the depth-control signal, it changes the operating point of the learned policy. On the easier GSM8K benchmark, the full *When2Think* model achieves comparable validation accuracy while reducing both token usage and Think Ratio relative to the *IDAC+BWS* variant. This indicates that IS helps expose and exploit the NOTHINK mode during training, encouraging more efficient behavior when explicit reasoning is unnecessary.

IDAC is necessary for difficulty-aware depth control. The *IS+BWS* variant retains balanced mode exploration but removes IDAC. While this variant can still learn to switch between THINK and NOTHINK, it lacks the difficulty-aware signal that determines how much reasoning should be allocated once THINK is selected. As a result, IS alone can encourage mode switching, but does not provide the same control over reasoning depth across easy and hard instances.

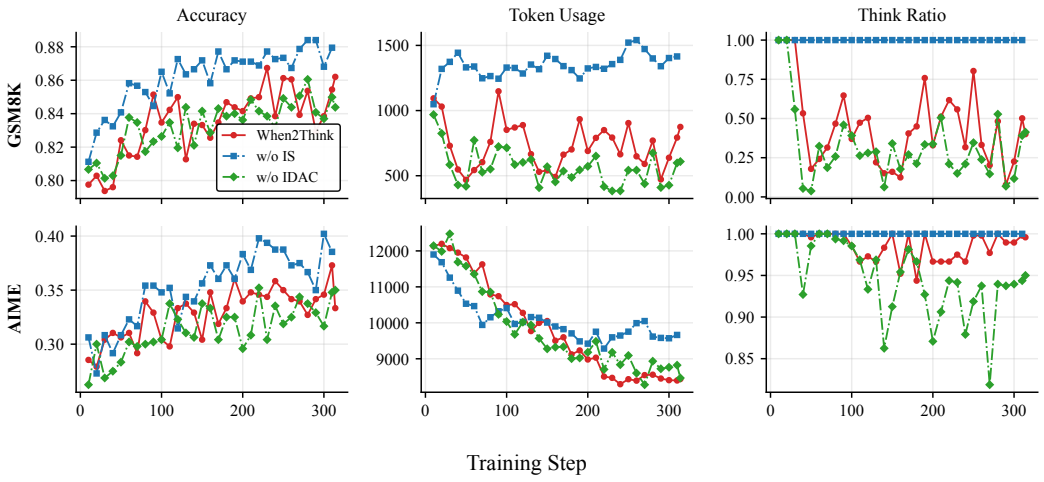


Figure 10: **Training Dynamics under IS, IDAC Ablations.** Training dynamics of three variants on DeepScaleR: the full *When2Think* model (IS + IDAC + BWS), a variant without IS (IDAC + BWS), and a variant without IDAC (IS + BWS). **Top:** GSM8K (easy). **Bottom:** AIME24 (hard). Each row reports validation accuracy, token usage, and Think Ratio over training steps.

D.4 Ablation on Hybrid Bonus Design

Separate bonuses provide limited benefit. As shown in Figure 11, varying the NoThink bonus η does not yield a clear improvement in the accuracy–efficiency trade-off. In particular, configurations with larger η tend to produce lower validation accuracy, suggesting that overly strong incentives for NoThink can bias the policy toward direct answering even when explicit reasoning remains useful. This behavior is consistent with hard-instance under-allocation: encouraging NoThink too aggressively can suppress necessary long-horizon reasoning.

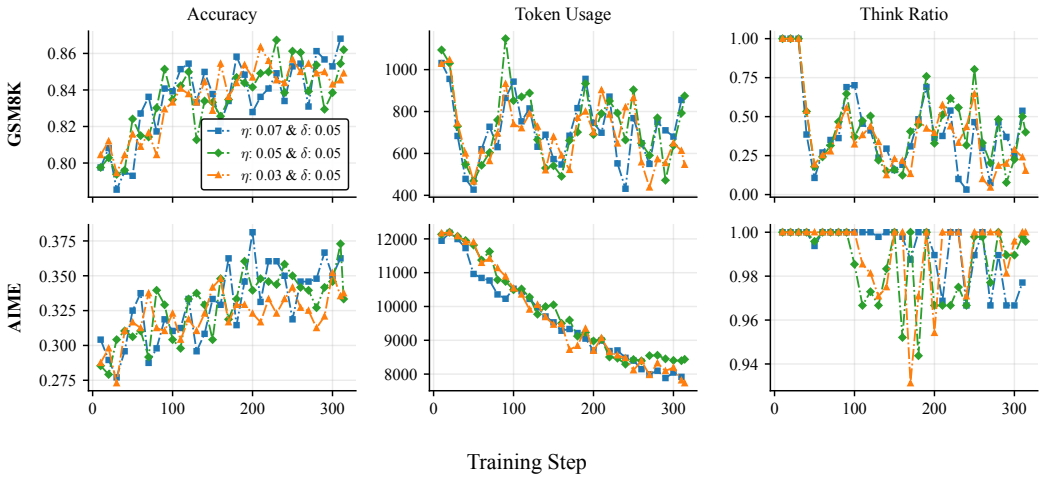


Figure 11: **Training Dynamics under η and δ Variations.** Training dynamics of *When2Think* under different η on DeepScaleR, with δ fixed to 0.05. **Top:** GSM8K (easy). **Bottom:** AIME24 (hard). Each row reports validation accuracy, token usage, and Think Ratio over training steps.

Unified bonus yields more stable training. We therefore simplify the reward by using a single correctness-gated efficiency coefficient δ for both modes, with IDAC controlling how strongly the bonus is applied to Think trajectories. The training dynamics in Figure 12 show that smaller unified bonus values lead to more stable behavior, avoiding both excessive NoThink collapse and overly verbose Think trajectories. This suggests that the reward does not require separate mode-specific bonus coefficients.

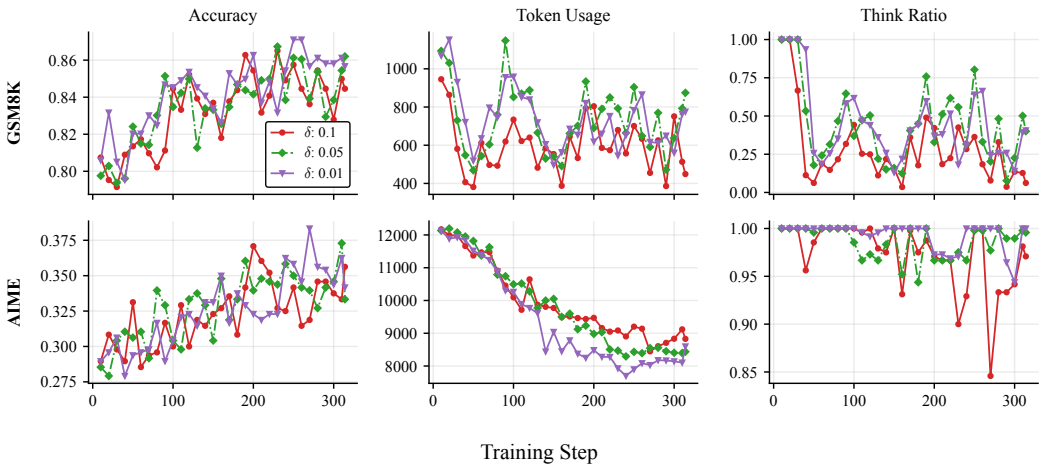


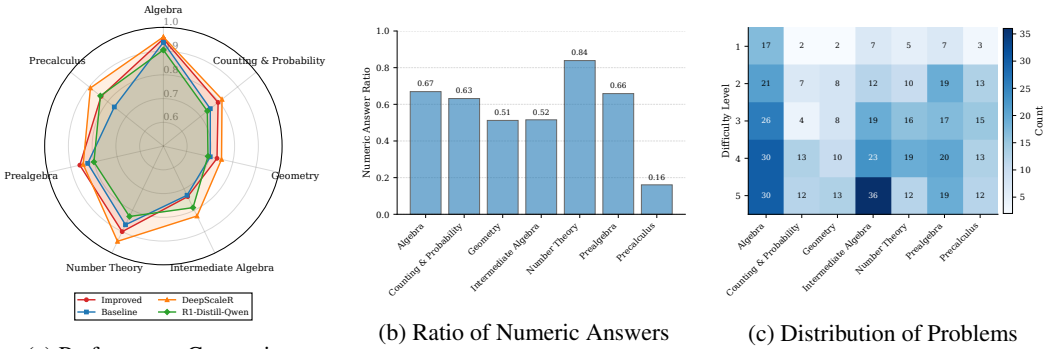
Figure 12: **Training Dynamics under δ Variations.** Training dynamics of *When2Think* under different efficiency-bonus coefficients δ on DeepScaleR. **Top:** GSM8K (easy). **Bottom:** AIME24 (hard). Each row reports validation accuracy, token usage, and Think Ratio over training steps.

D.5 Effectiveness of the Dual Math Verifier

We analyze the contribution of the Dual Math Verifier through subject-level performance on MATH-500. Figure 13 compares the Base Model (R1-Distill), the Baseline with standard verification, and our proposed method (Improved). Figure 13(c) further confirms that each subject exhibits sufficient coverage in both problem count and difficulty distribution, enabling meaningful comparison.

Robustness to Non-Numeric Answers. Subjects such as *Geometry* and *Precalculus* contain a high proportion of symbolic, structured, or text-heavy answers. In these categories, the Baseline exhibits notable performance degradation due to the inherent limitations of string-based verification, which fails to recognize semantically equivalent but syntactically distinct solutions (e.g., reordered vector coordinates or algebraically equivalent expressions). By incorporating symbolic equivalence checking, the Dual Math Verifier effectively resolves this mismatch, substantially improving robustness to non-numeric answer formats and yielding consistent gains across these subjects.

Trade-offs in Intermediate Algebra. Performance gains in *Intermediate Algebra* are comparatively smaller. This behavior is not attributable to data scarcity, as models such as DeepScaleR–trained on the same dataset–achieve stronger results, but rather reflects a trade-off between verification strictness and recall. In particular, complex multi-step derivations involving alternative but equivalent forms (e.g., factored versus expanded polynomials) may be falsely rejected under strict symbolic parsing. Moreover, efficiency constraints imposed by IDAC can limit the allocation of reasoning steps required for such transformations. These observations suggest that subject-aware verification tolerance or adaptive length modulation may further improve performance in algebra-intensive domains.



(a) Performance Comparison

(b) Ratio of Numeric Answers

(c) Distribution of Problems

Figure 13: **Subject-wise analysis of MATH-500. (a)** Performance across subjects. The proposed method (Improved) recovers baseline drops in geometry and precalculus. **(b)** Fraction of numeric-answer problems; low numeric ratios align with baseline failures. **(c)** Heatmap of problem counts across difficulty levels (1–5), confirming sufficient sample coverage for each subject.

E CASE STUDIES

E.1 OVERTHINKING ON EASY PROBLEM

[GSM-Plus; R1-Distill-Qwen-1.5B] Correct Overthinking Response (32,206 tokens)

Comet Halley, which is approximately 15 kilometers wide, orbits the sun every 75 years. Bill's dad saw the Comet when he was 30 years old. Bill, who has a brother 5 years younger than him, saw the comet a second time when he was three times the age his father was when he saw the Comet. Despite the fact that Bill's favorite number is 7, how old was Bill when he saw the Comet for the first time?

<THINK>
But according to the problem, the second appearance is at 90, which is 60 years after the first, which is inconsistent with the period of 75 years. Therefore, perhaps the problem is incorrect, or perhaps I'm misinterpreting the relationship between the father's age and Bill's age.
...
Wait, perhaps I'm overcomplicating it.
Wait, let's think: If the first appearance is on a 75-year interval, ... Therefore, T1 + 75 = 90, so T1 = 15.
Therefore, the first appearance is at 15, which is a 75-year interval. The second appearance is at 90, which is 75 years after 15.
...
Wait, but 15 years after the start would be the first appearance. But the problem says that the first appearance is on a 75-year interval. So, the first appearance is on a 75-year interval, which is the period.
Therefore, the first appearance is at 15, which is 15 years after the start, which is the first occurrence.
Therefore, the first appearance is at 15, which is 15 years after the start, and the second is at 90, which is 75 years after the first.
Thus, the first appearance is at 15, which is a 75-year interval.
Therefore, the first appearance is at 15.
Thus, the first appearance is at 15, which is 15 years after the start, and the second is at 90, which is 75 years after the first.
Therefore, the first appearance is at 15, which is a 75-year interval.
Hence, the first appearance is at 15.
</THINK> Think tokens: 31,919; Steps: 492; Thoughts: 439

Final Answer
The first appearance was on a 75-year interval at $\boxed{15}$.
The problem involves determining the first appearance of a bird on a 75-year interval. The key steps to solve this are as follows:
1. **Understanding the problem**: The first appearance of the bird on a 75-year interval is the starting point. The second appearance is noted as being 15 years after the first appearance.
2. **Setting up the equations**: - Let T_1 be the first appearance. - The second appearance is given as $T_2 = T_1 + 15$. - The second appearance is noted as 90 years after the first appearance, so $T_2 = 90$.
3. **Solving the equations**: - From the second appearance, we have $90 = T_1 + 15$. - Solving this gives $T_1 = 90 - 15 = 75$.
4. **Conclusion**: - The first appearance is at $T_1 = 75$, which is 75 years after the start. - The second appearance is at $T_2 = 90$, which is 15 years after the first appearance.
Thus, the first appearance of the bird was on a 75-year interval at $\boxed{15}$.

Figure 14: R1-Distill-1.5B GSM-Plus correct overthinking

E.2 UNDERTHINKING ON HARD PROBLEM

[AIME24; R1-Distill-Qwen-1.5B] Incorrect Underthinking Response (17,075 tokens)

Alice and Bob play the following game. A stack of n tokens lies before them. The players take turns with Alice going first. On each turn, the player removes either 1 token or 4 tokens from the stack. Whoever removes the last token wins. Find the number of positive integers n less than or equal to 2024 for which there exists a strategy for Bob that guarantees that Bob will win the game regardless of Alice's play.

<THINK>
Okay, so I have this problem where Alice and Bob are playing a game with a stack of n tokens. They take turns removing either 1 or 4 tokens, and the person who takes the last token wins. I need to find how many positive integers n $\leq$ 2024 allow Bob to have a winning strategy, no matter what Alice does.
Hmm, let me think about this. It sounds like a combinatorial game theory problem. Maybe I can model the winning and losing positions for each n. A winning position is one where the current player can force a win, and a losing position is one where no matter what the current player does, the opponent can force a win.
So, if I can figure out the losing positions, then the number of losing positions up to 2024 would be the number of n where Bob can win. That makes sense.
Let me try to list out some small values of n and see which are winning (W) or losing (L) positions.
...
Wait, that seems contradictory. Let me check.
...
Hmm, not sure.
...
Therefore, the losing positions are n where n mod5 is 0,1,2,3,4, and n $\neq$ 5,10,15,20,...
But that would mean losing positions are all n, which is not the case.
Therefore, I think I have to give up.
</THINK> Think tokens: 16,482; Steps: 778; Thoughts: 256

Final Answer
$\boxed{2022}$
To determine the number of positive integers $n \leq 2024$ for which there exists a winning strategy for Bob, we need to analyze the game where Alice and Bob alternately remove 1 or 4 tokens from a stack of n tokens. The player who cannot make a move loses.
Key steps:
...
After detailed analysis, the number of losing positions is found to be 2022. Therefore, the number of positive integers $n \leq 2024$ for which there exists a winning strategy for Bob is:

$$\boxed{2022}$$

Figure 15: R1-Distill-1.5B GSM-Plus incorrect underthinking

E.3 Comparison with When2Think and Base Model

[GSM-Plus] Problem

Nate's dog can dig six holes a day. Each hole takes 15 minutes. He digs for 14 days while Nate is on vacation. When Nate gets home, he starts filling in 9 holes a day, but the dog keeps digging 6 new holes every night. How many weeks does it take him to fill in all the holes?

[GSM-Plus; When2Think-1.5B] Correct NoThink Response (625 tokens)

<THINK> </THINK>

Let's break down the problem step by step.
Given:
1. **Nate's dog digs** 6 holes **a day**.
2. **Each hole takes** 15 minutes to dig.
3. **Nate digs for 14 days** while he is on vacation.
4. **When he gets home, he starts filling holes** at a rate of 9 holes per day.
5. **The dog continues digging** 6 new holes every night.
Objective:
Determine how many **weeks** it takes for Nate to fill all the holes.
Step-by-Step Solution:
1. **Calculate the total number of holes the dog digs while Natie is on vacation:**
- **Dogs digging time per day:** 6 holes
- **Days Natie digs:** 14 days
- **Total holes dug by the dog:** 6 holes/day $\times$ 14 days $=$ 84 holes
2. **Calculate the total time Natie spends digging:**
- **Each hole takes:** 15 minutes
- **Total time Natie digs:** 84 holes $\times$ 15 minutes/hole $=$ 1260 minutes
- **Convert minutes to days:** $\frac{1260 \text{ minutes}}{1440 \text{ minutes/day}} = 0.875$ days
3. **Calculate the number of holes Natie starts filling:**
- **Nate starts filling holes at:** 9 holes/day
- **Total holes dug by the dog during Natie's vacation:** 84 holes
- **Total holes Natie needs to fill:** 84 holes
4. **Determine the net rate at which holes are being filled:**
- **Dog digs:** 6 holes/night
- **Nate fills:** 9 holes/day
- **Net rate per day:** 9 holes/day $-$ 6 holes/day $=$ 3 holes/day
5. **Calculate the number of days needed to fill all holes:**
- **Total holes to fill:** 84 holes
- **Net rate:** 3 holes/day
- **Days needed:** $\frac{84 \text{ holes}}{3 \text{ holes/day}} = 28$ days
6. **Convert days into weeks:**
- **Days needed:** 28 days
- **Number of weeks:** $\frac{28 \text{ days}}{7 \text{ days/week}} = 4$ weeks
Final Answer

$\boxed{4}$

Figure 16: When2Think correct NoThink

[GSM-Plus; R1-Distill-Qwen-1-5B] Incorrect Response (8,837 tokens)

<THINK>
Okay, so I need to figure out how long it takes for Nate to fill all the holes his dog digs. Let me try to break this down step by step.
First, the dog digs six holes each day. Each hole takes 15 minutes to dig. So, I guess I should find out how much time the dog spends digging in total over the 14 days he's away. But wait, the dog also starts digging every night, right? So, each day, the dog digs six holes in the morning and six more in the evening. That means each full day, including night, the dog digs 12 holes.
Let me write that down:
...Wait, actually, no. Let me think again. ...Wait, I think I'm mixing up the time frames. Let me clarify. ...Wait, but this is confusing. Let me think about it step by step. ...Wait, that seems off. Wait, maybe I'm overcomplicating it. Let me think of it as the dog digging 12 holes per day, and the holes are being filled at 9 per day, but the dog is also digging 6 holes every night. ...Wait, maybe I'm misunderstanding the problem. Let me read it again. ...Wait, that seems off. I'm stuck. Maybe I need to think about the problem in a different way. ...So, even though the dog is also digging 6 holes every night, the net gain is 3 holes per day, so it takes 56 days to fill all the holes, which is 8 weeks.
So, the final answer is 8 weeks.
</THINK> Think tokens: 8,500; Steps: 322; Thoughts: 261

Final Answer
It takes Nate and his dog $\boxed{8}$ weeks to fill all the holes.
Nate's dog can dig 6 holes a day, and each hole takes 15 minutes to dig. During his vacation, the dog digs for 14 days.
First, we calculate the total number of holes the dog digs during the 14 days:

$$\text{Total holes dug} = 12\,\text{holes/day} \times 14\,\text{days} = 168\,\text{holes}$$

Next, we consider the rate at which the holes are being filled. The dog fills 9 holes every night. The net gain in the number of holes each day (after accounting for the holes dug and filled) is:

$$12\,\text{holes/day} - 9\,\text{holes/day} = 3\,\text{holes/day}$$

To find the total number of days required to fill all 168 holes, we divide the total number of holes by the net gain per day:

$$\text{Total days} = \frac{168\,\text{holes}}{3\,\text{holes/day}} = 56\,\text{days}$$

Converting the total days into weeks:

$$\text{Total weeks} = \frac{56\,\text{days}}{7\,\text{days/week}} = 8\,\text{weeks}$$

Thus, it takes Nate and his dog $\boxed{8}$ weeks to fill all the holes.

Figure 17: R1-Distill-1.5B incorrect overthinking

F Problem Characterization

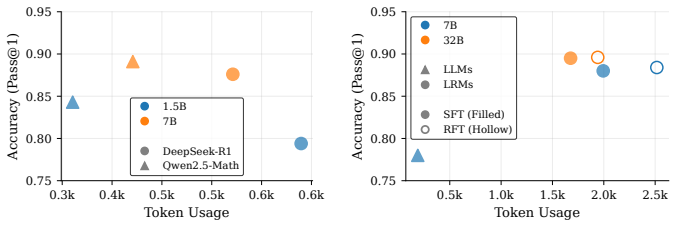
F.1 Impact of Model Scale and Supervision on Reasoning Efficiency

We examine reasoning efficiency across model scale and supervision using the Olmo-3 (Olmo et al., 2025) family (7B and 32B). Figure 18 shows the accuracy–token usage trade-off across model scales, post-training strategies (SFT vs. RFT), and reasoning paradigms (standard LLMs vs. LRMs). The results highlight systematic inefficiencies in current post-training approaches and illustrate how model capacity and supervision affect reasoning cost.

The Paradox of Scale: Smaller Models Overthink. Figure 18 reveals a clear *paradox of scale* in reasoning efficiency. Smaller 7B models are concentrated in the high token-usage regime relative to 32B models, despite achieving substantially lower accuracy. This indicates that limited model capacity is associated with inefficient and prolonged reasoning trajectories, a phenomenon referred to as *capability-induced overthinking*. In contrast, 32B models achieve higher accuracy with fewer tokens, exhibiting superior *reasoning economy*. A similar trend is observed in the R1-distill family (Table 9), suggesting that stronger base capabilities enable more direct and efficient problem-solving paths.

The Cost of Reinforcement Learning. Figure 18 shows a consistent *efficiency tax* under Reinforcement Fine-Tuning (RFT). Within the same model scale, RFT models (hollow markers) are systematically shifted toward higher token usage compared to their Supervised Fine-Tuned (SFT) counterparts (filled markers). These shifts are accompanied by only modest accuracy gains, resulting in a less favorable accuracy–token trade-off. This behavior is consistent with outcome-based reinforcement learning objectives that do not explicitly penalize token usage, leading to longer chain-of-thought generations across instances. Overall, RFT models occupy a Pareto-inferior region relative to SFT baselines.

The Mechanics of Inefficiency: Why RLVR Inflates Tokens. The token inflation observed in RFT models–particularly those trained with RLVR (Lambert et al., 2025), such as OlmoRL (Olmo et al., 2025)–follows directly from the optimization objective. Unlike SFT, which imitates concise human-curated reasoning traces, RLVR optimizes only for outcome correctness and imposes no explicit constraint on token usage. The absence of a KL regularization term further permits substantial deviation from the SFT initialization. Consequently, longer reasoning trajectories become a favorable strategy for increasing success probability. Without explicit difficulty-aware control, RLVR therefore tends to produce inefficient and overly verbose reasoning behavior.



(a) Qwen2.5-Math & R1-Distill (b) Olmo-3

Figure 18: **Efficiency landscape across model scales and tuning methods on GSM-Plus. (a) Model Scale and Reasoning Cost:** Reasoning-oriented models (LRMs, circles) consistently achieve higher accuracy than standard LLMs (triangles) but at the cost of significantly higher token usage. Notably, smaller LRMs (1.5B, blue) consume more tokens than larger LRMs (7B, orange), suggesting that smaller models require longer reasoning chains to compensate for limited capacity. **(b) Impact of Tuning Strategies (SFT vs. RFT):** Across both model sizes, RFT models (hollow markers) are systematically shifted to the right compared to SFT models (filled markers), indicating that reinforcement fine-tuning induces higher computational overhead regardless of model scale.