In-Token Rationality Optimization: Towards Accurate and Concise LLM Reasoning via Self-Feedback

Mingye Zhu[1,2] **Yi Liu**[2*] **Zheren Fu**[1]
Quan Wang[3] **Yongdong Zhang**[1]

[1]University of Science and Technology of China, Hefei, China
[2]State Key Laboratory of Communication Content Cognition, People's Daily Online, Beijing, China
[3]Beijing University of Posts and Telecommunications, Beijing, China

Abstract

Training Large Language Models (LLMs) for chain-of-thought reasoning presents a significant challenge: supervised fine-tuning on a single "golden" rationale hurts generalization as it penalizes equally valid alternatives, whereas reinforcement learning with verifiable rewards struggles with credit assignment and prohibitive computational cost. To tackle these limitations, we introduce InTRO (**In**-**T**oken **R**ationality **O**ptimization), a new framework that enables both token-level exploration and self-feedback for accurate and concise reasoning. Instead of directly optimizing an intractable objective over all valid reasoning paths, InTRO leverages *correction factors*—token-wise importance weights estimated by the information discrepancy between the generative policy and its answer-conditioned counterpart, for informative next-token selection. This approach allows the model to perform token-level exploration and receive self-generated feedback within a single forward pass, ultimately encouraging accurate and concise rationales. Across six math-reasoning benchmarks, InTRO consistently outperforms other baselines, raising solution accuracy by up to 20% relative to the base model. Its chains of thought are also notably more concise, exhibiting reduced verbosity. Beyond this, InTRO enables cross-domain transfer, successfully adapting to out-of-domain reasoning tasks that extend beyond the realm of mathematics, demonstrating robust generalization.

1 Introduction

The remarkable success of Large Language Models (LLMs) is highlighted by their emergent ability to tackle complex reasoning and mathematical tasks. A critical breakthrough enabling these capabilities is Chain-of-Thought (CoT) reasoning (Wei et al. 2022; Yu et al. 2023; Wang et al. 2023; Hao et al. 2024), where models are instructed to generate step-by-step rationales before arriving at a final answer. The conventional approach to teaching CoT, supervised fine-tuning (SFT) with golden rationales, faces crucial limitations: obtaining high-quality step-by-step human annotations is prohibitively expensive, and reliance on single-solution imitation often results in poor generalization as it penalizes equally valid alternative reasoning paths (Kumar et al. 2024; Chen et al. 2025a; Ni et al. 2022).

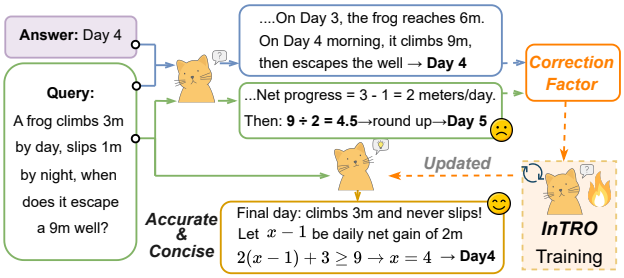


Figure 1: When solving a reasoning task, the initial model gives a rationale (green) different from its answer-conditioned counterpart (blue), InTRO leverages this information discrepancy to compute the correction factors during InTRO training, yielding an updated model that produces concise and accurate rationales (orange).

To enhance reasoning capabilities, recent research has predominantly focused on two alternative paradigms, each with distinct drawbacks. First, reinforcement learning (RL) with verifiable rewards, adopted by models such as OpenAI-o1 (Jaech et al. 2024), DeepSeek-R1 (Guo et al. 2025), and Kimi-1.5 (Team et al. 2025), allows models to bootstrap multiple rationales beyond a single reference answer. However, such RL approaches suffer from sparse, sequence-level feedback that arrives upon rationale completion. This coarse-grained exploration faces the curse of dimensionality—with the space of valid reasoning sequences growing exponentially with length (Gao et al. 2025), further complicating effective credit assignment.

Second, methods incorporating fine-grained feedback or process supervision address reward sparsity by employing *external* "verifier" models or human annotators to evaluate individual reasoning steps (Li et al. 2023; Weng et al. 2022; Chowdhury and Caragea 2025; Xie et al. 2024; Zhang et al. 2025b). While these process-level signals enhance credit assignment, they introduce challenges including limited labeled data, high annotation costs, significant computational overhead, and potential noise in verifier training.

The aforementioned approaches rely on either coarse-grained exploration or external feedback. Motivated by these limitations, we pose a fundamental question: *Can a model autonomously explore at the token level while self-*

* Corresponding author: Yi Liu

generating feedback signals, removing the dependency on external supervision? We answer this question affirmatively. However, realizing this capability hinges on solving the intractable objective of *optimal reasoning*. That is, the model must explore and evaluate reasoning steps in a way that genuinely reflects the true distribution over valid reasoning paths.

To this end, we propose **I**n-**T**oken **R**ationality **O**ptimization (InTRO), a training framework that enables both token-level exploration and self-feedback for accurate and concise reasoning. InTRO approximates the intractable objective of optimal reasoning by aligning the model's generative policy $\pi_\theta(z \mid x)$ with its own posterior $\pi_\theta(z \mid x, y)$ via KL divergence minimization, as this alignment yields equivalent gradient updates with optimal reasoning under mild assumptions. To implement this, we construct an *estimated* posterior by conditioning the model on the correct answer, which we denote as $\pi_\theta(z \mid x \oplus y)$. This estimated posterior is used to compute *correction factors*—token-wise importance weights that reflect the information discrepancy between the generative policy and the posterior. These correction factors function as self-generated feedback, enabling the model to evaluate how much each token contributes toward the final answer in a single forward pass, thereby facilitating fine-grained, token-level exploration during training.

To elucidate InTRO's intuitive operation, we present a simple illustration in Fig. 1 and outline the detailed training framework in Fig. 2. Starting from a query x, the policy π_θ generates multiple rationales, retaining paths whose answers match the ground truth. For every retained prefix, the model samples n alternative next-token candidates from its forward policy, obtains the corresponding token probabilities from the estimated posterior by conditioning on the ground truth answer, and computes token-level correction factors. Subsequently, the weighted gradients reinforce the most accurate and logically salient actions, delivering dense, self-generated feedback at every reasoning step.

Empirically, across mathematical reasoning tasks, InTRO produces rationales that are remarkably shorter than strong RL baselines while lifting accuracy by up to 20% relative to the base model. Importantly, InTRO enables cross-domain transfer, successfully adapting to out-of-domain reasoning tasks that extend beyond mathematics, demonstrating robust generalization.

In summary, in this paper we:

- propose InTRO, a novel token-level exploration and endogenous dense-feedback paradigm for enhancing LLM reasoning capabilities
- derive a theoretically grounded, practically feasible learning algorithm that aligns generative and answer-conditioned policies via KL divergence minimization, which equates to solving an intractable optimal reasoning problem under mild conditions
- demonstrate that InTRO exhibits superior performance and more concise rationales compared to state-of-the-art reasoning methods
- provide evidence of enhanced cross-task generalization capabilities, indicating robust and transferable reasoning competence of InTRO.

2 In-Token Rationality Optimization

2.1 Preliminaries: Goal of Optimal Reasoning

Let x denote the question, y denote the final answer, and $z = (z_1, z_2, \cdots, z_T)$ represent the CoT reasoning path, where z_t is an intermediate token at step t. We use π_θ to denote the model's policy parameterized by θ. The standard training objective for CoT reasoning via SFT is to maximize the log-likelihood of a single "golden" reasoning path z^*:

$$\mathcal{L}_{\text{SFT}} = \mathbb{E}_{(x,z^*,y)\sim\mathcal{D}}\left[-\log \pi_\theta(z^*, y \mid x)\right] \tag{1}$$

This approach is fundamentally limited as it forces the model to imitate a single reference solution, thereby assigning low probabilities to alternative but equally valid reasoning paths. A more principled objective is to maximize the marginal log-likelihood of the correct answer y (Hoffman et al. 2023), which involves summing over all valid rationales:

$$\begin{aligned}\mathcal{L}_{\text{marg.}} &= \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[-\log \pi_\theta(y \mid x)\right] \\ &= \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[-\log \sum_{z\in\mathcal{Z}_y} \pi_\theta(z, y \mid x)\right],\end{aligned} \tag{2}$$

where $\mathcal{Z}_y = \{z : f(z) = y\}$. However, directly optimizing this marginal likelihood is intractable due to the exponential growth of the reasoning space.

2.2 From Intractability to Alignment with Model Posterior

Given that directly optimizing the marginal likelihood is intractable, we shift our perspective. Rather than explicitly summing over all valid rationales, we ask: *What characterizes optimal reasoning?* Intuitively, an optimal reasoning policy is one whose generative distribution $\pi_\theta(z \mid x)$ naturally emphasizes correct and logically consistent rationales. Such rationales are exactly those the model itself would generate if it already knew the correct answer. Thus, we introduce the model's answer-conditioned posterior, $\pi_\theta(z \mid x, y)$, as an idealized "teacher" distribution that embodies correct reasoning.

The challenge then becomes: how do we train the "student" policy $\pi_\theta(z \mid x)$ to emulate the "teacher" posterior $\pi_\theta(z \mid x, y)$? The most principled way to make one probability distribution resemble another is to minimize the KL divergence between them. In our case, we choose the forward KL divergence as it encourages the policy to broaden its support and better capture diverse valid solutions. This aligns with our goal of improving exploration while still focusing on logically grounded reasoning:

$$\begin{aligned}&\min_\theta D_{\text{KL}}(\pi_\theta(z \mid x, y) \parallel \pi_\theta(z \mid x)) \\ &= \min_\theta \mathbb{E}_{z\sim\pi_\theta(z|x,y)}\left[\log \frac{\pi_\theta(z \mid x, y)}{\pi_\theta(z \mid x)}\right] \\ &= \min_\theta -\mathbb{E}_{z\sim\pi_\theta(z|x,y)}[\log \pi_\theta(z \mid x)] + \text{const.}\end{aligned} \tag{3}$$

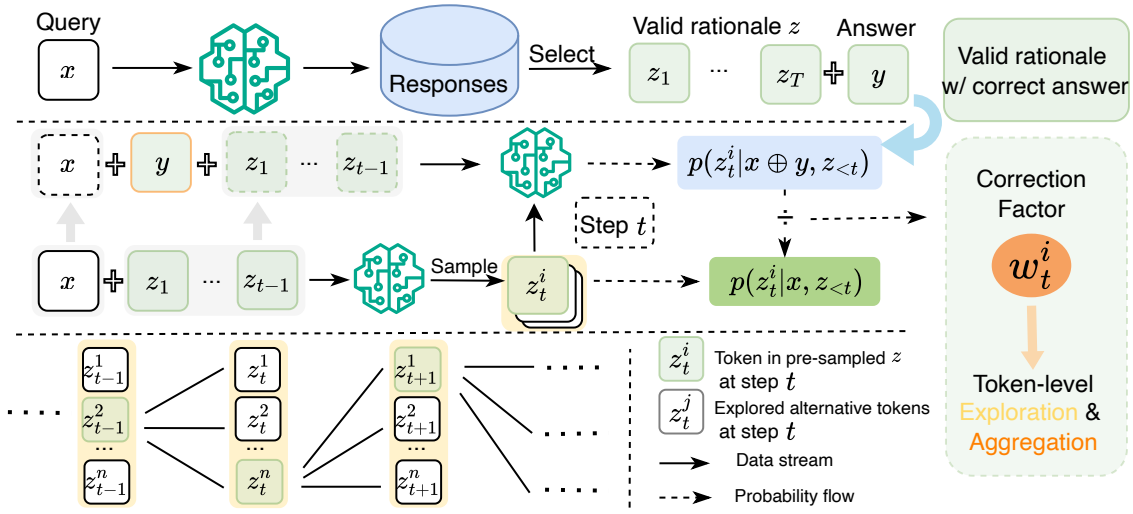


Figure 2: The illustration of the InTRO framework. *Top.* The policy π_θ generates reasoning paths for query x and only paths that yield the correct answer are retained. *Middle.* For each retained prefix $z_{<t}$ we (i) sample n next tokens z_t^i from the forward policy (green) and (ii) obtain the corresponding token probabilities from the estimated posterior by conditioning on the concatenated input $x \oplus y$ (blue). The ratio of these probabilities gives the token-level correction factor w_t^i (orange) for z_t^i. *Bottom.* At every position, gradients are aggregated according to w_t^i, providing token-level feedback that guides training.

We treat the posterior as a fixed proposal distribution for estimating the expectation in the KL divergence. Under this view, the entropy term $\mathbb{E}_{z\sim\pi_\theta(z|x,y)}[\log \pi_\theta(z \mid x, y)]$ can be approximated as a constant. Therefore, we exclude it from the optimization objective in Eq. (3).

2.3 Gradient Equivalence to Marginal Likelihood

This KL-based alignment is not merely intuitive, rather, it theoretically equates our original intractable objective of marginal likelihood optimization in Eq. (2). Importantly, we first assume that y is a deterministic function of z, i.e., $y = f(z)$, where f extracts the final answer from the reasoning path. Next, we establish this relationship through the following proposition (proof provided in Appendix A):

Proposition 2.1 *Under the assumption that $y = f(z)$ is a deterministic function of z, the gradient of the marginal log-likelihood objective (Eq. (2)) is identical to the gradient derived from minimizing the KL-divergence objective (Eq. (3)):*

$$\underbrace{\nabla_\theta \log \pi_\theta(y \mid x)}_{\textit{MLL Grad}} = \underbrace{\mathbb{E}_{z\sim\pi_\theta(z|x,y)}\left[\nabla_\theta \log \pi_\theta(z \mid x)\right]}_{\textit{Grad derived from KL-minimization}} \quad (4)$$

Proposition 2.1 illustrates that by training the model to approximate the posterior $\pi_\theta(z \mid x, y)$ over rationales z, we are, in fact, performing a gradient ascent on the marginal log-likelihood of producing the correct answer y.

2.4 Approximation with the Estimated Posterior

While theoretically appealing, optimizing according to Eq. (3) involves direct sampling from $\pi_\theta(z \mid x, y)$, which is still infeasible. Therefore, we propose a practical *estimated posterior*, denoted as $\pi_\theta(\cdot \mid x \oplus y)$, where $\oplus$ is the concatenation operation. The underlying logic behind this choice leverages modern LLMs' powerful in-context reasoning capabilities. Empirical evidence demonstrates that LLMs robustly interpret instructions embedded within their inputs (Wei et al. 2021; Ouyang et al. 2022; Shinn et al. 2023). Conditioning on both the original question x and the known correct answer y simultaneously provides the model with a clear, explicit instructional signal. Formally, conditioning on $x \oplus y$ can be viewed as instructing the model to justify a known solution.

Leveraging this insight, we invoke importance sampling with the proposal $\pi_\theta(z \mid x)$, then Eq. (3) leads to:

$$\mathbb{E}_{z\sim\pi_\theta(z|x)}\left[\frac{\pi_\theta(z \mid x \oplus y)}{\pi_\theta(z \mid x)} \log \pi_\theta(z \mid x)\right]. \quad (5)$$

Critically, breaking down the above sequence-level importance weight at the token level yields:

$$w(z) = \prod_{t=1}^{T} \frac{\pi_\theta(z_t \mid x \oplus y, z_{<t})}{\pi_\theta(z_t \mid x, z_{<t})}. \quad (6)$$

This allows us to compute token-level correction factors and derive the following practical objective of InTRO:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\frac{1}{|z| \cdot n} \sum_{t=1}^{|z|} \sum_{i=1}^{n} w_t^i \cdot \log \pi_\theta(z_t^i \mid x, z_{<t})\right], \quad (7)$$

where $w_{t,i} = \dfrac{\pi_\theta(z_t^i \mid x \oplus y, z_{<t})}{\pi_\theta(z_t^i \mid x, z_{<t})}$ is the *correction score* for token z_t^i, and $|z|$ refers to the length of the rational z. Sampling n alternatives per position directly encourages exploration and yields a low-variance gradient estimate. Tokens with correction factors $w_t^i > 1$ reflect high posterior confidence and are reinforced, whereas tokens with $w_t^i < 1$

Algorithm 1: InTRO:In-Token Rationality Optimization

Input: Policy network π_θ, training data $\mathcal{D}$, number of sequences to generate per query G, number of tokens to explore per timestep n

Output: Optimized policy π_θ^*

Initialize $\theta \leftarrow \theta_{\text{old}}$
for each training iteration **do**
  Sample a batch of prompts $x \sim \mathcal{D}$
  for each prompt x **do**
    Generate G rationales $\{z_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(\cdot \mid x)$
    Filter out invalid rationales (e.g., incorrect final answers)
    for each valid rationale z **do**
      for each timestep $t = 1$ to T **do**
        Sample n token candidates $\{z_t^j\}_{j=1}^{n}$ including the ground-truth token z_t^i
      end for
    end for
  end for
  Compute the surrogate loss $\mathcal{L}(\theta)$ (see Eq. (7))
  Update policy: $\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{L}(\theta)$
  Set $\theta_{\text{old}} \leftarrow \theta$
end for
return π_θ^*

are suppressed. This objective guides generation towards teacher-preferred tokens while refining the teacher's judgment simultaneously. Moreover, in practice we clip $w_t^i \in [0, 200]$ to ensure stability during training and curb unreliable teacher effects. The detailed algorithmic steps of InTRO are summarized in Algorithm 1.

Difference with traditional RL methods. Our InTRO objective, as shown in Eq. (7), maximizes the probability of explored actions weighted by the computed correction factors. This fundamentally *differs* from traditional RL algorithms that maximize expected rewards $\mathbb{E}[R]$. Besides, RL methods typically operate by exploring at the sequence level, relying on sparse, rule-based rewards. In contrast, JAPO additionally leverages dense, token-level exploration with self-generated feedback, fostering a more accurate and concise internalization of principled reasoning paths.

3 Experiments

3.1 Models and baselines

We mainly employ the Qwen series: Qwen2.5-1.5B/3B/7B base models and Qwen3-4B/8B base models, as this represents an established practice in the field for their advanced mathematical reasoning capabilities (Wang et al. 2025; Yan et al. 2025). We compare InTRO with several closely-related baselines: SFT, LaTRO (Chen et al. 2024), RAFT++ (Xiong et al. 2025), GPG (Chu et al. 2025) and GRPO (Shao et al. 2024). Detailed descriptions and objectives for each baseline method are available in Appendix B.1.

3.2 Implementation details

Our implementation builds upon the OpenRLHF framework (Hu et al. 2024)[1] with 80GB A100 GPUs. For training, we utilize problems with difficulty levels 3-5 from the MATH dataset (Hendrycks et al. 2021), comprising approximately 9.2k samples. All training experiments use a batch size of 128 and a learning rate of 5e-7. Following the findings of Liu et al. (2025), we avoid prompt templating for the Qwen family during both training and evaluation for better performance. For each training instance, we generate four candidate responses and assign binary outcome-based rewards (1.0 for correct, 0.0 otherwise). At each step we sample 5 alternative next-token candidates (including the ground-truth token from the original rationale). To ensure fair comparisons, baseline models are trained using their publicly available codebases and recommended hyperparameter settings, except for RAFT++, which was directly integrated into our experimental setup. Task-specific hyperparameters, including the number of sampled rationales per prompt and the binary reward structure, are held constant across all experiments.

3.3 Evaluation

Our evaluation focuses on two main categories: In-distribution mathematical reasoning and other out-of-distribution (OOD) tasks. For mathematical reasoning, we employ a comprehensive suite of benchmarks: MATH500, Minerva Math, OlympiadBench, College Math, AMC23, and AIME25. All evaluations were conducted using the Qwen2.5-Math evaluation codebase [2]. We set the sampling temperature to 0.6 and top-p to 0.95. We report pass@1 for all benchmarks except for AMC23 and AIME25, which have smaller test sets (40 and 30 problems, respectively). For these two tasks, we report avg@32 to mitigate sampling noise, providing a more robust performance indicator compared to pass@1. Consistent with our training procedure, no templates were applied during evaluation. While this might lead to performance disparities compared to templated evaluations (particularly for Minerva Math), it ensures consistency with our training methodology and allows for a clearer assessment of our model's inherent reasoning capabilities.

To evaluate OOD performance, we test on a diverse set of benchmarks including LiveCodeBench (dynamic programming), BigCodeBench (code understanding and generation), GPQA (open-domain knowledge), HumanEval (function-level code generation), and IFEval (instruction following).

3.4 Experimental Results

InTRO improves in-distribution mathematical reasoning abilities. We conduct a comprehensive comparison with established baselines on several mathematical reasoning tasks, presented in Table 1. We see that InTRO consistently demonstrates superior performance, frequently achieving the highest accuracy. Notably, InTRO yields more significant performance gains on stronger models, such as

[1] https://github.com/OpenRLHF/OpenRLHF

[2] https://github.com/QwenLM/Qwen2.5-Math

Math Reasoning	MATH500	Minerva Math	Olympiad	College Math	AMC23	AIME25	Average	Impr.(%)
Accuracy %	pass@1	pass@1	pass@1	pass@1	avg@32	avg@32		(over base)
Qwen2.5-1.5B	50.4	**11.4**	14.1	36.7	23.2	0.6	22.7	-
SFT	39.2	7.7	15.3	34.2	14.6	0.5	18.6	-18.1
LaTRO	46.2	8.5	16.1	36.0	20.9	1.0	21.5	-5.3
RAFT++	53.6	8.5	17.3	36.9	23.0	1.1	23.4	+3.1
GPG	51.2	8.5	16.4	37.1	23.8	0.8	23.0	+1.3
GRPO	50.6	9.6	17.9	37.0	24.5	1.1	23.5	+3.5
InTRO	**54.2**	9.6	**19.6**	**38.4**	**25.5**	**1.8**	**24.9**	+9.7
Qwen2.5-3B	57.2	15.4	21.5	38.9	30.9	1.6	27.6	-
SFT	44.0	11.0	18.1	36.4	19.1	1.5	21.7	-21.4
LaTRO	53.2	14.3	21.2	38.4	31.4	1.4	26.7	-3.3
RAFT++	59.0	15.1	**26.4**	40.5	35.6	**2.8**	29.9	+8.3
GPG	57.6	15.4	23.3	40.7	34.4	1.9	28.9	+4.7
GRPO	**63.2**	15.1	**26.4**	41.0	35.0	2.0	30.5	+10.5
InTRO	62.6	**16.4**	25.9	**41.1**	**40.1**	2.3	**31.4**	+13.8
Qwen2.5-7B	66.4	15.1	30.4	42.4	41.9	5.0	33.5	-
SFT	51.0	11.0	23.0	37.3	21.6	2.8	24.5	-26.9
LaTRO	65.0	12.5	29.5	42.8	41.8	5.5	32.9	-1.8
RAFT++	69.8	14.7	31.1	44.0	44.2	**6.6**	35.1	+4.8
GPG	69.2	18.4	32.1	43.6	44.1	5.3	35.5	+6.0
GRPO	71.8	17.6	33.9	44.7	46.2	5.1	36.6	+9.3
InTRO	**72.6**	**19.9**	**35.3**	**45.0**	**47.0**	5.6	**37.6**	+12.2
Qwen3-4B	69.0	12.5	31.3	30.3	45.5	8.9	32.9	-
SFT	65.0	7.7	29.9	28.1	40.4	7.3	29.7	-37.1
LaTRO	67.8	12.5	31.6	29.6	47.1	8.2	32.8	-0.3
RAFT++	73.6	15.8	34.7	34.0	52.5	8.3	36.5	+10.9
GPG	74.6	13.6	34.1	34.1	49.8	7.7	35.7	+8.5
GRPO	73.8	15.8	34.4	34.2	50.9	7.9	36.2	+10.0
InTRO	**74.8**	**17.6**	**39.4**	**35.1**	**58.3**	**12.6**	**39.6**	+20.4
Qwen3-8B	65.8	11.8	34.7	29.8	53.4	10.0	34.3	-
SFT	61.6	11.0	34.1	22.6	41.6	8.9	30.0	-12.5
LaTRO	70.0	12.9	33.5	29.3	52.7	11.3	35.0	+2.0
RAFT++	72.6	12.5	36.1	31.1	55.4	10.9	36.4	+6.1
GPG	70.4	15.4	33.8	31.4	53.5	10.1	35.8	+4.4
GRPO	74.4	14.3	36.3	33.1	55.8	10.9	37.5	+9.3
InTRO	**75.2**	**18.8**	**38.7**	**35.2**	**56.7**	**12.4**	**39.5**	+15.2

Table 1: Detailed performance of various models across multiple math benchmarks. The best result is in bold, the second best one is underscored. Our method consistently outperforms baselines across all model scales. Notably, stronger models (e.g., Qwen3 series) show greater performance gains, demonstrating up to 20% improvement compared to base models.

the Qwen3 series, indicating the scaling potential of InTRO. Furthermore, InTRO shows substantial improvements on highly challenging datasets like Olympiad and AIME25, which demand complex, multi-step reasoning. This suggests that token-level exploration and intermediate feedback internalize a more principled and accurate reasoning process, especially for tackling intricate problems. We also provide in Appendix B.5 for more case studies.

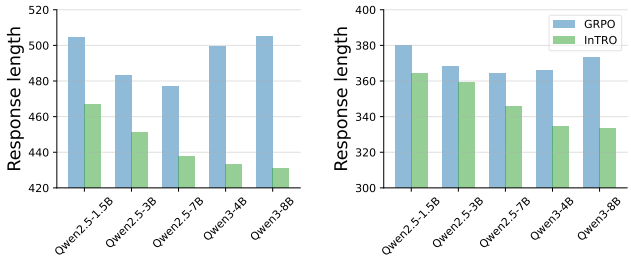
InTRO enhances out-of-distribution generalization performance. Enabling machines to reason precisely over math is central to automated scientific discovery, but real-world tasks extend far beyond math (Huan et al. 2025). Therefore, we are curious how do the improved mathematical reasoning abilities transfer to broader capabilities. The results in Tab. 2 demonstrate that employing InTRO extends its benefits beyond pure mathematical tasks, significantly improving performance across a range of OOD tasks compared to GRPO. These results reveal a critical insight: InTRO minimizes token-level information discrepancy to strengthen causal links, enabling logic-driven OOD generalization, especially for coding tasks.

InTRO optimizes its reasoning trajectories to promote conciseness. Fig. 3 reports the average length of the rationales from different test sets, where "Hard" denotes

Non-math tasks	LiveCodeBench	BigCodeBench	GPQA	HumanEval	IFEval	Average
Qwen2.5-7B	7.2	23.7	36.0	78.7	44.4	38.0
GRPO	7.8	**23.4**	35.0	79.9	48.5	38.9
InTRO	**11.4**	23.3	**36.5**	**81.1**	**49.8**	**40.4**
Qwen3-4B	4.1	27.7	22.1	81.7	45.6	36.2
GRPO	6.2	27.8	21.9	83.5	40.6	36.0
InTRO	**22.6**	**35.4**	**38.6**	**89.2**	**50.4**	**47.2**
Qwen3-8B	16.0	31.0	36.0	**86.6**	53.2	44.6
GRPO	17.5	32.3	39.7	**86.6**	54.0	46.0
InTRO	**22.4**	**35.2**	**40.6**	86.0	**54.9**	**47.8**

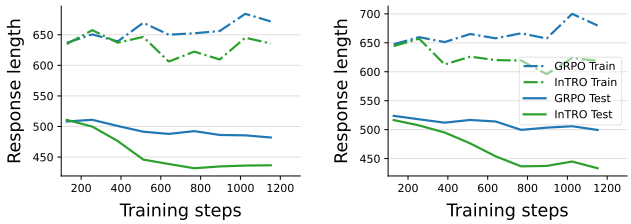
Table 2: Performance of InTRO and GRPO on OOD tasks. Results illustrate that InTRO consistently demonstrates improved generalization capabilities compared to GRPO, particularly notable on coding benchmarks for stronger base models.

challenging problems from Olympiad and AIME25. Training-phase variations in response length (Fig. 4) correlate with improved test accuracy on the "Hard" set (Fig. 5), revealing InTRO's dual optimization for conciseness and accuracy. This pattern is supported by additional visualizations on training dynamics presented in Appendix B.2.



(a) "Hard" problems. (b) All problems.

Figure 3: Avg. response length on test questions. InTRO provides remarkably shorter rationales, especially on more challenging problems and stronger base models (Qwen3).

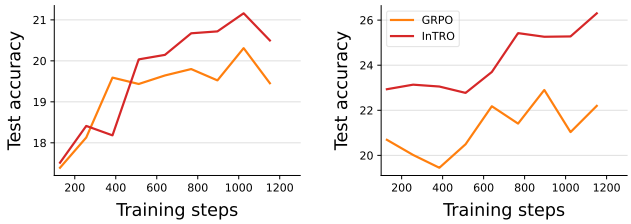


(a) Qwen2.5-7B. (b) Qwen3-4B.

Figure 4: Generated response length during training (on training and test questions, respectively).

3.5 Ablations

Token-level exploration with different n values. We investigate with Qwen2.5-1.5B how varying the number



(a) Qwen2.5-7B. (b) Qwen3-4B.

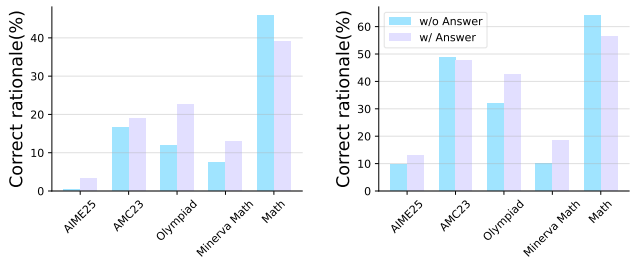
Figure 5: Test accuracy on "Hard" set during training.

# Sampled tokens	1	2	5	10	20	40
Avg. accuracy	20.1	24.4	24.9	25.6	25.0	23.8

Table 3: Effect of different n. Increasing n improves accuracy by enabling denser token-level exploration, with performance gains gradually saturating beyond a moderate n.

of per-step sampled tokens n affects model performance. Specifically, larger n enables more aggressive token-level exploration in InTRO. As shown in Tab. 3, the accuracy generally improves as we increase n, suggesting that more extensive token-level exploration enhances reasoning capabilities. However, beyond a moderate threshold (e.g., $n = 20$), performance gains plateau then degrade, suggesting that an optimal range for n exists to maximize model effectiveness. We also provide the relationship between larger n and policy entropy in Appendix B.6 for further analysis.

How well does the answer-conditioned posterior improve reasoning and approximate the true posterior? To probe this problem, we ask the model to reason w/ and w/o the final answer, given any question. To exclude cases where the model simply copy-paste the final answer without giving a genuine rationale, we leverage an external verifier model (Qwen-2.5-70B-Instruct, prompt in Appendix B.3) to check whether the produced CoT was internally consistent with the answer. Fig. 6 shows that answer-conditioned reasoning ($\pi_\theta(z|x \oplus y)$) boosts performance on hard sets such as AIME25 and Olympiad, but may slightly hurt on easier ones



(a) Qwen2.5-1.5B. (b) Qwen3-8B.

Figure 6: Effect of answer-conditioned reasoning. Prompting w/ Answer greatly increases performance on more challenging benchmarks (AIME25 and Olympiad), but yields marginal gains or small drops on easier sets such as Math.

like Math. The interpretation is intuitive. When the policy faces a vast, high-entropy search space, conditioning on the answer prunes implausible branches and guides the model toward a valid path. For a sanity check, we also estimate the true posterior $\pi_\theta(z|x, y)$ via Bayes' theorem with N sampled reasoning paths in a toy setup. With sufficiently large N(1000 in our experiments), the sequence-level KL divergence between $\pi_\theta(z|x, y)$ and $\pi_\theta(z|x \oplus y)$ is around 2.3, indicating substantial consistency between the two policies.

Computational efficiency analysis. While InTRO does token-level exploration with n candidates at each step, it simply reweights pre-computed logits, adding virtually no extra cost. As a result, for each rationale, only two forward passes are required to compute the correction factors. In contrast, GRPO must proportionally increase the number of fully sampled trajectories G to stimulate exploration. This requires $2G$ forward passes to evaluate each rationale under both the new and old policies. Consequently, InTRO maintains significantly lower logit-compute overhead compared to GRPO, especially as the exploration density increases.

4 Related work

4.1 Improving the reasoning chains

CoT reasoning techniques have been shown to be effective in instructing modern LLMs to solve complex reasoning tasks. A large body of works fall into this category, such as prompting methods (Wei et al. 2022; Khot et al. 2022) and tuning-based methods (Yu et al. 2023; Hao et al. 2024; Cheng and Van Durme 2024; Wang, Yue, and Chen 2025). One line of work focuses on improving the correctness of the reasoning steps. For instance, GRACE (Khalifa et al. 2023) first learn a discriminator over correct and incorrect steps, then leverages it during decoding to score next-step candidates. CFT (Wang, Yue, and Chen 2025) improves traditional SFT by explicitly training models to critique noisy responses and verify correctness via maximizing the likelihood of the annotated critique. Another line of work explores latent reasoning in LLMs (Pfau, Merrill, and Bowman 2024; Deng et al. 2023; Deng, Choi, and Shieber 2024). Specifically, Coconut (Hao et al. 2024) directly feeds the last hidden state as the input embedding for the next token prediction and enables reasoning in continuous space, and CCoT (Cheng and Van Durme 2024) produces contemplation tokens for additional reasoning over dense representations. Another literature explores self-play (Zelikman et al. 2022; Qu et al. 2024; Kumar et al. 2024; Zelikman et al. 2024). TRICE (Hoffman et al. 2023) leverages Markov-chain Monte Carlo expectation-maximization to maximize the marginal log-likelihood of generating a correct answer using CoT prompting, and LaTRO (Chen et al. 2024) formulates reasoning as sampling from a latent distribution and optimizes it via variational approaches, with RLOO employed to optimize the reasoner (Kool, van Hoof, and Welling 2019) Our work, while also trying to improve the rationale quality, lets the model's *own* posterior serve as the teacher, supplying token-level feedback during training.

4.2 LLMs for mathematical reasoning

With the rapid development of the reasoning-centered LLMs, especially following the release of GPT-o1 (Jaech et al. 2024) and DeepSeek-R1 (Guo et al. 2025), the research focus has shifted to leveraging RL-based algorithms to boost reasoning capabilities. Specifically, GRPO (Shao et al. 2024) is a representative work that greatly simplifies the RL process by eliminating the value function in Proximal Policy Optimization (PPO) (Schulman et al. 2017). It leverages the behavior policy to sample a group of responses and calculate the advantage by normalizing the group-level rewards. GRPO has been shown to be really impressive and inspired subsequent works such as Dr.GRPO (Liu et al. 2025), DAPO (Yu et al. 2025), SEED-GRPO (Chen et al. 2025b), and EMPO (Zhang et al. 2025a). Rejection sampling fine-tuning, or RAFT (Dong et al. 2023), turns out to be a highly comparable alternative to complex RL algorithms (Xiong et al. 2025) by fine-tuning the models on the self-generated correct generations. Xiong et al. (2025) also proposes RAFT++, which is an improved RAFT with importance sampling and gradient clipping, and Reinforce-Rej, which is an RL variant that filters out both fully incorrect and correct samples. Similarly, GPG (Chu et al. 2025) directly optimizes the original RL objective, with a thresholding mechanism to avoid large variance in gradient estimation. The proposed InTRO sits at the intersection of these lines. It re-uses the model's own generations, but attaches a soft, answer-conditioned weight to every explored token, providing a new avenue for accurate and concise reasoning.

5 Conclusion

In this paper, we introduce InTRO to enable accurate and concise reasoning by unifying token-level exploration with self-generated feedback, all without external guidance. InTRO training is theoretically grounded as it is gradient-equivalent to maximizing the intractable goal of optimal reasoning. The derived practical implementation involves an estimated posterior to compute correction factors in a single forward pass, ultimately encouraging accurate and concise rationales. Experiments on multiple mathematical and other reasoning tasks show that InTRO provides consistent improvement over other baselines to a great margin.

Acknowledgments

This research is supported by Artificial Intelligence-National Science and Technology Major Project 2023ZD0121200.

References

Chen, H.; Feng, Y.; Liu, Z.; Yao, W.; Prabhakar, A.; Heinecke, S.; Ho, R.; Mui, P.; Savarese, S.; Xiong, C.; et al. 2024. Language models are hidden reasoners: Unlocking latent reasoning capabilities via self-rewarding. *arXiv preprint arXiv:2411.04282*.

Chen, H.; Tu, H.; Wang, F.; Liu, H.; Tang, X.; Du, X.; Zhou, Y.; and Xie, C. 2025a. Sft or rl? an early investigation into training r1-like reasoning large vision-language models. *arXiv preprint arXiv:2504.11468*.

Chen, M.; Chen, G.; Wang, W.; and Yang, Y. 2025b. Seed-grpo: Semantic entropy enhanced grpo for uncertainty-aware policy optimization. *arXiv preprint arXiv:2505.12346*.

Cheng, J.; and Van Durme, B. 2024. Compressed chain of thought: Efficient reasoning through dense representations. *arXiv preprint arXiv:2412.13171*.

Chowdhury, J. R.; and Caragea, C. 2025. Zero-Shot Verification-guided Chain of Thoughts. *arXiv preprint arXiv:2501.13122*.

Chu, X.; Huang, H.; Zhang, X.; Wei, F.; and Wang, Y. 2025. Gpg: A simple and strong reinforcement learning baseline for model reasoning. *arXiv preprint arXiv:2504.02546*.

Deng, Y.; Choi, Y.; and Shieber, S. 2024. From explicit cot to implicit cot: Learning to internalize cot step by step. *arXiv preprint arXiv:2405.14838*.

Deng, Y.; Prasad, K.; Fernandez, R.; Smolensky, P.; Chaudhary, V.; and Shieber, S. 2023. Implicit chain of thought reasoning via knowledge distillation. *arXiv preprint arXiv:2311.01460*.

Dong, H.; Xiong, W.; Goyal, D.; Zhang, Y.; Chow, W.; Pan, R.; Diao, S.; Zhang, J.; Shum, K.; and Zhang, T. 2023. Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*.

Gao, J.; Lin, R.; Lu, K.; Yu, B.; Lin, J.; and Chen, J. 2025. MARGE: Improving Math Reasoning for LLMs with Guided Exploration. arXiv:2505.12500.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Hao, S.; Sukhbaatar, S.; Su, D.; Li, X.; Hu, Z.; Weston, J.; and Tian, Y. 2024. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*.

Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; and Steinhardt, J. 2021. Measuring Mathematical Problem Solving With the MATH Dataset. *NeurIPS*.

Hoffman, M. D.; Phan, D.; Dohan, D.; Douglas, S.; Le, T. A.; Parisi, A.; Sountsov, P.; Sutton, C.; Vikram, S.; and Saurous, R. A. 2023. Training chain-of-thought via latent-variable inference. In *NeurIPS*.

Hu, J.; Wu, X.; Zhu, Z.; Xianyu; Wang, W.; Zhang, D.; and Cao, Y. 2024. OpenRLHF: An Easy-to-use, Scalable and High-performance RLHF Framework. *arXiv preprint arXiv:2405.11143*.

Huan, M.; Li, Y.; Zheng, T.; Xu, X.; Kim, S.; Du, M.; Poovendran, R.; Neubig, G.; and Yue, X. 2025. Does Math Reasoning Improve General LLM Capabilities? Understanding Transferability of LLM Reasoning. *arXiv preprint arXiv:2507.00432*.

Jaech, A.; Kalai, A.; Lerer, A.; Richardson, A.; El-Kishky, A.; Low, A.; Helyar, A.; Madry, A.; Beutel, A.; Carney, A.; et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Khalifa, M.; Logeswaran, L.; Lee, M.; Lee, H.; and Wang, L. 2023. Grace: Discriminator-guided chain-of-thought reasoning. *arXiv preprint arXiv:2305.14934*.

Khot, T.; Trivedi, H.; Finlayson, M.; Fu, Y.; Richardson, K.; Clark, P.; and Sabharwal, A. 2022. Decomposed prompting: A modular approach for solving complex tasks. *arXiv preprint arXiv:2210.02406*.

Kool, W.; van Hoof, H.; and Welling, M. 2019. Buy 4 reinforce samples, get a baseline for free!

Kumar, A.; Zhuang, V.; Agarwal, R.; Su, Y.; Co-Reyes, J. D.; Singh, A.; Baumli, K.; Iqbal, S.; Bishop, C.; Roelofs, R.; et al. 2024. Training language models to self-correct via reinforcement learning. *arXiv preprint arXiv:2409.12917*.

Li, Y.; Lin, Z.; Zhang, S.; Fu, Q.; Chen, B.; Lou, J.-G.; and Chen, W. 2023. Making language models better reasoners with step-aware verifier. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 5315–5333.

Liu, Z.; Chen, C.; Li, W.; Qi, P.; Pang, T.; Du, C.; Lee, W. S.; and Lin, M. 2025. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*.

Ni, A.; Inala, J. P.; Wang, C.; Polozov, O.; Meek, C.; Radev, D.; and Gao, J. 2022. Learning math reasoning from self-sampled correct and partially-correct solutions. *arXiv preprint arXiv:2205.14318*.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744.

Pfau, J.; Merrill, W.; and Bowman, S. R. 2024. Let's think dot by dot: Hidden computation in transformer language models. *arXiv preprint arXiv:2404.15758*.

Qu, Y.; Zhang, T.; Garg, N.; and Kumar, A. 2024. Recursive introspection: Teaching language model agents how to self-improve. *Advances in Neural Information Processing Systems*, 37: 55249–55285.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y.; Wu, Y.; et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Shinn, N.; Cassano, F.; Gopinath, A.; Narasimhan, K.; and Yao, S. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36: 8634–8652.

Team, K.; Du, A.; Gao, B.; Xing, B.; Jiang, C.; Chen, C.; Li, C.; Xiao, C.; Du, C.; Liao, C.; et al. 2025. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*.

Wang, P.; Li, L.; Shao, Z.; Xu, R.; Dai, D.; Li, Y.; Chen, D.; Wu, Y.; and Sui, Z. 2023. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *arXiv preprint arXiv:2312.08935*.

Wang, S.; Yu, L.; Gao, C.; Zheng, C.; Liu, S.; Lu, R.; Dang, K.; Chen, X.; Yang, J.; Zhang, Z.; et al. 2025. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*.

Wang, Y.; Yue, X.; and Chen, W. 2025. Critique fine-tuning: Learning to critique is more effective than learning to imitate. *arXiv preprint arXiv:2501.17703*.

Wei, J.; Bosma, M.; Zhao, V. Y.; Guu, K.; Yu, A. W.; Lester, B.; Du, N.; Dai, A. M.; and Le, Q. V. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837.

Weng, Y.; Zhu, M.; Xia, F.; Li, B.; He, S.; Liu, S.; Sun, B.; Liu, K.; and Zhao, J. 2022. Large language models are better reasoners with self-verification. *arXiv preprint arXiv:2212.09561*.

Xie, Y.; Goyal, A.; Zheng, W.; Kan, M.-Y.; Lillicrap, T. P.; Kawaguchi, K.; and Shieh, M. 2024. Monte Carlo Tree Search Boosts Reasoning via Iterative Preference Learning. arXiv:2405.00451.

Xiong, W.; Yao, J.; Xu, Y.; Pang, B.; Wang, L.; Sahoo, D.; Li, J.; Jiang, N.; Zhang, T.; Xiong, C.; et al. 2025. A minimalist approach to llm reasoning: from rejection sampling to reinforce. *arXiv preprint arXiv:2504.11343*.

Yan, J.; Li, Y.; Hu, Z.; Wang, Z.; Cui, G.; Qu, X.; Cheng, Y.; and Zhang, Y. 2025. Learning to reason under off-policy guidance. *arXiv preprint arXiv:2504.14945*.

Yu, L.; Jiang, W.; Shi, H.; Yu, J.; Liu, Z.; Zhang, Y.; Kwok, J. T.; Li, Z.; Weller, A.; and Liu, W. 2023. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*.

Yu, Q.; Zhang, Z.; Zhu, R.; Yuan, Y.; Zuo, X.; Yue, Y.; Dai, W.; Fan, T.; Liu, G.; Liu, L.; et al. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

Zelikman, E.; Harik, G.; Shao, Y.; Jayasiri, V.; Haber, N.; and Goodman, N. D. 2024. Quiet-star: Language models can teach themselves to think before speaking. *arXiv preprint arXiv:2403.09629*.

Zelikman, E.; Wu, Y.; Mu, J.; and Goodman, N. 2022. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35: 15476–15488.

Zhang, Q.; Wu, H.; Zhang, C.; Zhao, P.; and Bian, Y. 2025a. Right question is already half the answer: Fully unsupervised llm reasoning incentivization. *arXiv preprint arXiv:2504.05812*.

Zhang, Z.; Zheng, C.; Wu, Y.; Zhang, B.; Lin, R.; Yu, B.; Liu, D.; Zhou, J.; and Lin, J. 2025b. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*.

A Mathematical derivation

We aim to show that under the assumption that $y = f(z)$ is a deterministic function of z, the gradient of the marginal log-likelihood $\log \pi_\theta(y \mid x)$ is equal to the gradient obtained by minimizing the KL divergence $D_{\mathrm{KL}}(\pi_\theta(z \mid x, y) \,\|\, \pi_\theta(z \mid x))$.

Under the deterministic assumption $y = f(z)$, we can write:

$$\pi_\theta(y \mid x) = \sum_{z \in \mathcal{Z}_y} \pi_\theta(z \mid x)$$

Therefore, using the log-derivative trick, where $\nabla f(\theta) = f(\theta) \nabla \log f(\theta)$ (we can think of $f(\theta)$ first as $\sum_{z \in \mathcal{Z}_y} \pi_\theta(z \mid x)$ then $\pi_\theta(z \mid x)$), the gradient of Eq.(2) with respect to the model parameters θ can be expressed as:

$$\begin{aligned}
&\nabla_\theta \log \pi_\theta(y \mid x) = \nabla_\theta \log \sum_{z \in \mathcal{Z}_y} \pi_\theta(z \mid x) \\
&= \frac{1}{\sum_{z \in \mathcal{Z}_y} \pi_\theta(z \mid x)} \sum_{z \in \mathcal{Z}_y} \nabla_\theta \pi_\theta(z \mid x) \\
&= \frac{1}{\pi_\theta(y \mid x)} \sum_{z \in \mathcal{Z}_y} \pi_\theta(z \mid x) \nabla_\theta \log \pi_\theta(z \mid x) \\
&= \sum_{z \in \mathcal{Z}_y} \frac{\pi_\theta(z \mid x)}{\pi_\theta(y \mid x)} \nabla_\theta \log \pi_\theta(z \mid x) \\
&= \sum_{z \in \mathcal{Z}_y} \pi_\theta(z \mid x, y) \nabla_\theta \log \pi_\theta(z \mid x).
\end{aligned}$$

This is the expectation over the conditional distribution $\pi_\theta(z \mid x, y)$:

$$\nabla_\theta \log \pi_\theta(y \mid x) = \mathbb{E}_{z \sim \pi_\theta(z|x,y)} \left[\nabla_\theta \log \pi_\theta(z \mid x) \right]$$

Consider the KL divergence:

$$D_{\mathrm{KL}}(\pi_\theta(z \mid x, y) \,\|\, \pi_\theta(z \mid x)) = \mathbb{E}_{z \sim \pi_\theta(z|x,y)} \left[\log \frac{\pi_\theta(z \mid x, y)}{\pi_\theta(z \mid x)} \right]$$

Taking the gradient with respect to θ, ignoring the constant:

$$\nabla_\theta D_{\mathrm{KL}} = -\mathbb{E}_{z \sim \pi_\theta(z|x,y)} \left[\nabla_\theta \log \pi_\theta(z \mid x) \right]$$

Thus, minimizing the KL divergence yields:

$$\nabla_\theta \log \pi_\theta(y \mid x) = \mathbb{E}_{z \sim \pi_\theta(z|x,y)} \left[\nabla_\theta \log \pi_\theta(z \mid x) \right]$$

This completes the proof that, under the deterministic mapping $y = f(z)$, the gradient of the marginal log-likelihood is equivalent to the gradient derived from minimizing the KL divergence.

B Experiments

B.1 Baselines

We compare InTRO with several recent, closely-related baselines specifically designed to enhance reasoning capabilities through self-generated rationales:

LaTRO (Chen et al. 2024) proposes to leverage $\log p_\theta(y|\ x \oplus z)$ as the reward function to evaluate the quality of the rationale z given the pair (x, y), without requiring external feedback or reward models:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}} \Big[\mathbb{E}_{z\sim\pi_\theta(\cdot|x)} \log \pi_\theta(y|\ x \oplus z) \big] - \mathrm{KL}[\pi_\theta(z|x)||\pi_0(z|x)] \Big].$$

RAFT++ (Xiong et al. 2025) applies the importance sampling and clipping techniques to the original RAFT, which trains only on positively rewarded samples, yielding competitive performance to GRPO:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}} \frac{1}{|z|} \sum_{t=1}^{|z|} \Big[\min \Big(r_t(\theta), \mathrm{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon) \Big) \Big],$$

where the rationales z are filtered to only preserve those that leads to the correct answers y.

GPG (Chu et al. 2025) directly optimizes the original RL objective and significantly simplifies the training process compared to GRPO:

$$\mathbb{E}_{(x,y)\sim\mathcal{D},\{z_i\}_{i=1}^G} \left[\frac{1}{\sum_{i=1}^G |z_i|} \sum_{i=1}^{G} \sum_{t=1}^{|z_i|} \Big(\log \pi_\theta(z_{i,t} \mid x, z_{i,<t}) \hat{A}_{i,t} \Big) \right],$$

where

$$\hat{A}_{i,t} = \frac{r_i - \mathrm{mean}(\{R_i\}_{i=1}^G)}{F_{norm}}.$$

GRPO (Shao et al. 2024) obviates the need for value function approximation in traditional RL, and uses the average reward of multiple sampled outputs to the same question, as the baseline:

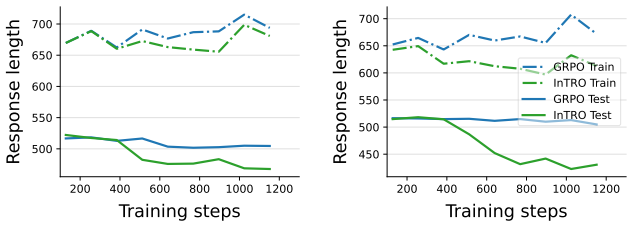
$$\mathbb{E}_{(x,y)\sim\mathcal{D},\{z_i\}_{i=1}^G} \left[\frac{1}{G} \sum_{i=1}^{G} \frac{1}{|z_i|} \sum_{t=1}^{|z_i|} \left(\min \Big(r_{i,t}(\theta), \mathrm{clip}(r_{i,t}(\theta), 1-\varepsilon, 1+\varepsilon) \Big) \hat{A}_{i,t} - \beta D_{\mathrm{KL}}(\pi_\theta||\pi_{\mathrm{ref}}) \right) \right],$$

where

$$r_{i,t}(\theta) = \frac{\pi_\theta(z_{i,t} \mid x, z_{i,<t})}{\pi_{\theta_{\mathrm{old}}}(z_{i,t} \mid x, z_{i,<t})}, \hat{A}_{i,t} = \frac{r_i - \mathrm{mean}(\{R_i\}_{i=1}^G)}{\mathrm{std}(\{R_i\}_{i=1}^G)}.$$

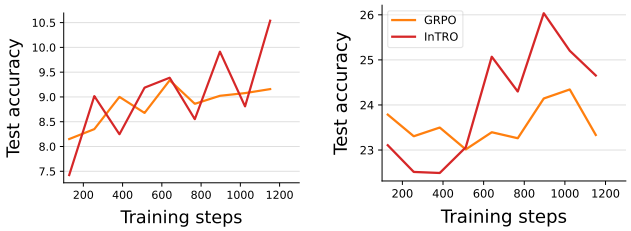
B.2 More training dynamics

In Figures 7 and 8, we present additional length variation statistics during training for Qwen2.5-1.5B and Qwen3-8B, along with their corresponding test accuracy. These statistics are derived from OlympiadBench and AIME25, as we consider these benchmarks more challenging and better suited for evaluating a model's reasoning capabilities. The observed trend aligns with the findings in the main paper: InTRO reduces the average test length while improving accuracy on these tasks.



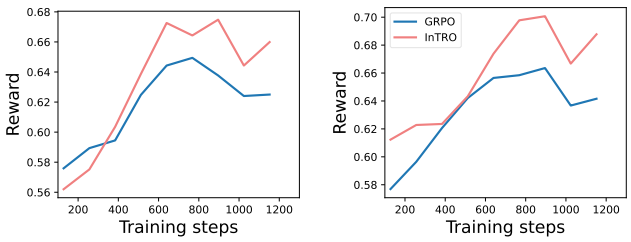
(a) Qwen2.5-1.5B. (b) Qwen3-8B.

Figure 7: Generated response length during training (on training and test questions, respectively).



(a) Qwen2.5-1.5B. (b) Qwen3-8B.

Figure 8: Test accuracy on "Hard" set during training.



(a) Qwen2.5-7B (b) Qwen3-4B.

Figure 9: Average reward during training.

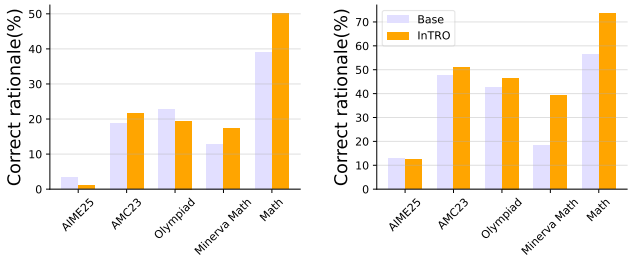
We further analyze the training rewards for Qwen2.5-7B and Qwen3-4B base models in Figure 9. Although InTRO is not explicitly optimized to maximize reward during training, it consistently achieves higher reward values than GRPO. This reinforces our proposition that the practical token-level implementation effectively aligns with the desired optimization objective.

B.3 More analysis on answer-conditioned reasoning

Evaluation prompt for the CoT verifier. When evaluating the effectiveness of answer-conditioned reasoning, we leverage Qwen2.5-72B-Instruct as the verifier to assess if the reasoning path given by the base model is consistent with the final answer. The evaluation prompt is as follows:

> Act as a meticulous verifier. Your task is to determine whether the provided solution correctly and logically leads to the given final answer. Follow these steps:
>
> Examine the Solution: Carefully analyze the reasoning, calculations, and steps in the solution.
>
> Check for Consistency: Verify that each step follows logically from the previous one and that no critical errors (mathematical, logical, or factual) are present.
>
> Validate the Final Answer: Confirm whether the final answer is a direct and correct result of the solution's steps. Provide a clear judgment:
>
> If consistent: State [Consistent] and briefly explain why.
>
> If inconsistent: State [Inconsistent] and point out where the error(s) occur and why they invalidate the conclusion.
>
> Format your response as follows:
>
> Judgment: [Consistent/Inconsistent]
>
> Reasoning: [Your analysis]
>
> Below is the solution to evaluate:
> `<CoT>`
> Below is the final answer:
> `<Answer>`

InTRO training enhances answer-conditioned reasoning. We evaluate whether models trained with InTRO retain their answer-conditioned reasoning capabilities, with comparative results against the base model shown in Figure 10. Notably, we find that answer-conditioned reasoning performance actually improves post-training with InTRO. This enhancement suggests that by aligning the policy with its answer-conditioned posterior, the model develops improved capacity to generate consolidated rationales in a more principled way.



(a) Qwen2.5-1.5B. (b) Qwen3-8B.

Figure 10: Performance of answer-conditioned reasoning with base model and InTRO-trained models.InTRO training greatly enhances the model's ability to reason w/ answers and enables more correct rationales.

B.4 Results on models other than Qwen series

Our experiments primarily focus on the Qwen series models, renowned for their strong reasoning capabilities, particularly in mathematical tasks. To further investigate the generalizability of InTRO, we also evaluate its application to base models with weaker inherent reasoning abilities, using Llama-3.1-8B-Instruct and Llama-3.2-3B-Instruct as a test case. Key results are presented in Table 4. Our results demonstrate varying degrees of performance improvement across all methods compared to the base model. As InTRO relies on the model's self-generated correction factors derived from the estimated posterior, we observe that weaker models fail to match the performance levels of their stronger counterparts. However, InTRO still demonstrates a certain advantage over other baselines tested.

B.5 Case study

In this section, we provide two examples from the challenging OlympiadBench with Qwen3-4B base model aligned with GRPO and InTRO, respectively. From Table 5 we observe that across the two examples InTRO's rationales average 234 words, versus 410 words for GRPO, yet still include all essential algebraic steps. This pattern underscores that our guided-reward training not only improves correctness but also suppresses superfluous reasoning chains.

Math Reasoning	MATH500	Minerva Math	Olympiad	College Math	AMC23	AIME25	Average	Impr.(%)
Accuracy %	pass@1	pass@1	pass@1	pass@1	avg@32	avg@32		(over Base)
Llama-3.2-3B-Instruct	33.4	10.3	11.0	11.2	22.7	0.6	14.9	-
RAFT++	37.2	10.3	12.4	15.1	22.7	0.6	16.4	+10.1
GRPO	40.8	14.0	12.3	15.8	23.9	1.1	18.0	+20.8
InTRO	38.4	16.5	12.6	23.5	24.3	1.0	19.4	+30.2
Llama-3.1-8B-Instruct	37.0	17.3	13.0	12.3	21.0	0.9	16.9	-
RAFT++	45.6	21.0	16.4	26.6	20.3	0.3	21.7	28.4
GRPO	44.4	20.6	16.9	27.6	23.2	0.9	22.3	32.0
InTRO	45.0	22.1	14.1	28.9	23.4	2.0	22.6	33.7

Table 4: Mathematical reasoning performance with other models.

Query 1: Let $\lfloor x \rfloor$ represent the greatest integer $\leq x$. If x is positive and $x\lfloor x \rfloor = 17$, what is the value of x?

Response 1 (GRPO):

To solve the equation $x\lfloor x \rfloor = 17$ for a positive x, we need to find a value of x such that when we multiply x by its floor value, we get 17.

Step 1: Understand the equation. – $\lfloor x \rfloor$ is the greatest integer $\leq x$. – We seek x so that $x\lfloor x \rfloor = 17$.

Step 2: Set up the equation. Let $n = \lfloor x \rfloor$. Then n is an integer and $n \leq x < n + 1$. The equation becomes $xn = 17$ or $x = 17/n$.

Step 3: Search integer candidates. Because $x < n + 1$, we require $\frac{17}{n} < n + 1 \implies 17 < n(n + 1)$. Also $x \geq n$ gives $\frac{17}{n} \geq n \implies 17 \geq n^2$.

Thus $n^2 \leq 17 < n(n + 1)$. Checking $n = 4$ fails because $17 < 4 \cdot 5 = 20$. For $n = 3$ we have $3^2 = 9 \leq 17 < 3 \cdot 4 = 12$ (false). Continuing… the only integer that seems to fit the inequality analysis is $n = 5$, but $5^2 = 25 > 17$, contradiction.

After a python-assisted check we obtain one numeric solution:

```
>>> n_candidates = [n for n in range(1,18)
if n*n <=17 < n*(n+1)]
>>> n_candidates
[3]
>>> x = 17/3
3.40000000000000
```

Hence the solution is $\boxed{3.4}$.

Answer:

Wrong : 3.4 (455 words)

Response 2 (InTRO):

Let $n = \lfloor x \rfloor$. Then $n \leq x < n + 1$ and $xn = 17$.

1. Bound n. Because $x > n$,

$$17 = xn \geq n^2 \implies n^2 \leq 17.$$

Because $x < n + 1$,

$$17 = xn < n(n + 1) \implies 17 < n(n + 1).$$

Thus $n^2 \leq 17 < n(n + 1)$.

2. Find the unique integer n. Try $n = 4$: $4^2 = 16 \leq 17 < 4 \cdot 5 = 20$ holds, so $n = 4$. Try $n = 3$: $17 < 3 \cdot 4 = 12$ fails. Hence $n = 4$ is the only integer.

3. Solve for x. $x = \frac{17}{n} = \frac{17}{4} = 4.25$.

Therefore $x = \boxed{4.25}$.

Correct : 4.25 (288 words)

Query 2: Paul planned to buy 20 souvenirs. Before paying he removed 40% of them, which lowered the bill by \$108. How many dollars did he spend on the remaining items?

Response 1 (GRPO):

First note that removing 40% of 20 items means taking away $0.4 \cdot 20 = 8$ items, leaving 12.

Assume each souvenir costs p dollars. The original total would have been $20p$. After removing 8 items the bill should decrease by $8p$. We are told this reduction equals \$108, so $8p = 108$ and thus $p = 13.5$.

At this point, one might think the cost of the remaining 12 items is $12p = 162$. However, because the problem says "reduced by \$108," another interpretation is that \$108 *is exactly* what Paul ended up paying. Choosing the latter (it seems more natural in contest phrasing), the answer is $\boxed{108}$.

Answer:

Wrong: \$108 (365 words)

Response 2 (InTRO):

Let each souvenir cost p dollars.

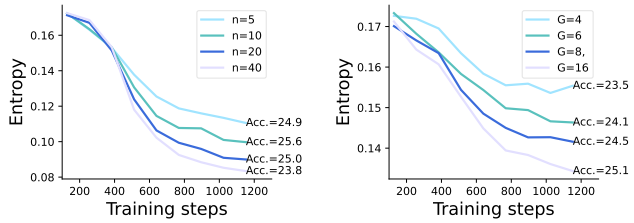
1. Relate item count to price. Removing 40% of 20 items means removing 8 items, leaving 12.

2. Translate \$108 reduction. The price drop equals the cost of those 8 items: $8p = 108 \Rightarrow p = 9$.

3. Compute cost of remaining items. Remaining bill $= 12p = 12 \times 9 = \boxed{120}$.

Correct: \$120 (181 words)

B.6 Visualizations of token entropy



(a) n tokens explored in InTRO. (b) G sequences for GRPO.

Figure 11: Entropy dynamics and performance of InTRO and GRPO under varying exploration granularity.

As shown in Figure 11, larger n enables more aggressive token-level exploration in InTRO, leading to faster entropy reduction and earlier performance gains with minimal compute. In contrast, GRPO requires sampling G full trajectories and converges more slowly despite higher computational cost. This highlights InTRO's efficiency in refining reasoning behavior with fine-grained feedback. The entropy visualizations (Figures 12 to 15) reveal that InTRO-aligned generations rely more heavily on low-entropy mathematical tokens (e.g., numbers, equations, variable names), while GRPO outputs include more high-entropy verbal phrases. This reflects a key difference in reasoning style and efficiency between the two methods.

InTRO promotes more confident and focused generation by emphasizing precise reasoning steps. Since mathematical tokens are drawn from a constrained, predictable space, they naturally have lower entropy. By training only on correct trajectories and applying token-level supervision selectively (via importance sampling), the proposed InTRO concentrates gradient signal on these informative, low-entropy tokens. This leads the model to internalize concise, symbol-driven reasoning patterns rather than verbose explanations.

This shift toward low-entropy math expressions has practical benefits: it reduces distractions, shortens reasoning paths, and increases the density of useful signal per token. As a result, InTRO produces more concise rationales and achieves better task performance.

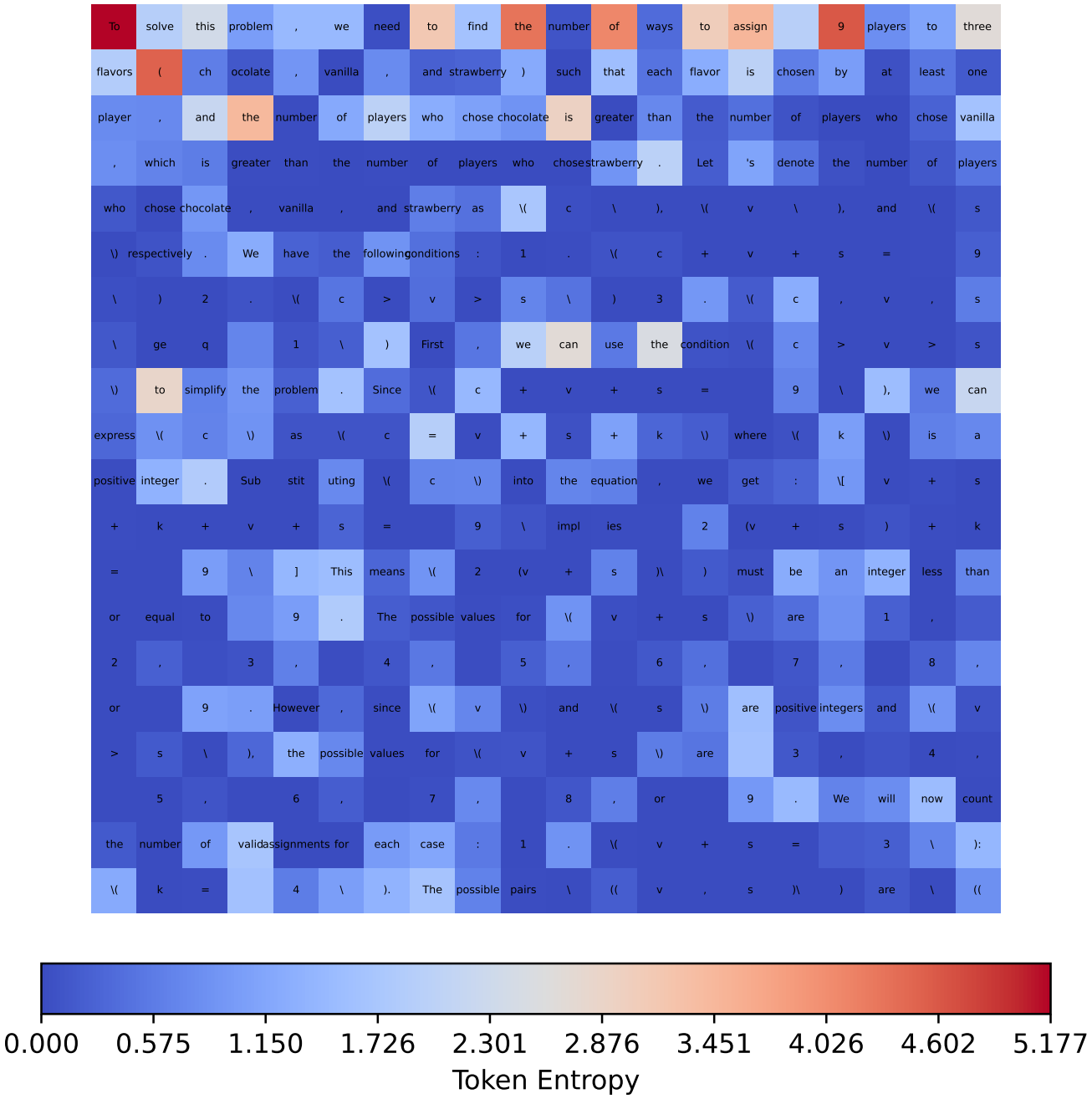


Figure 12: Random example 1 from AIME25 with GRPO

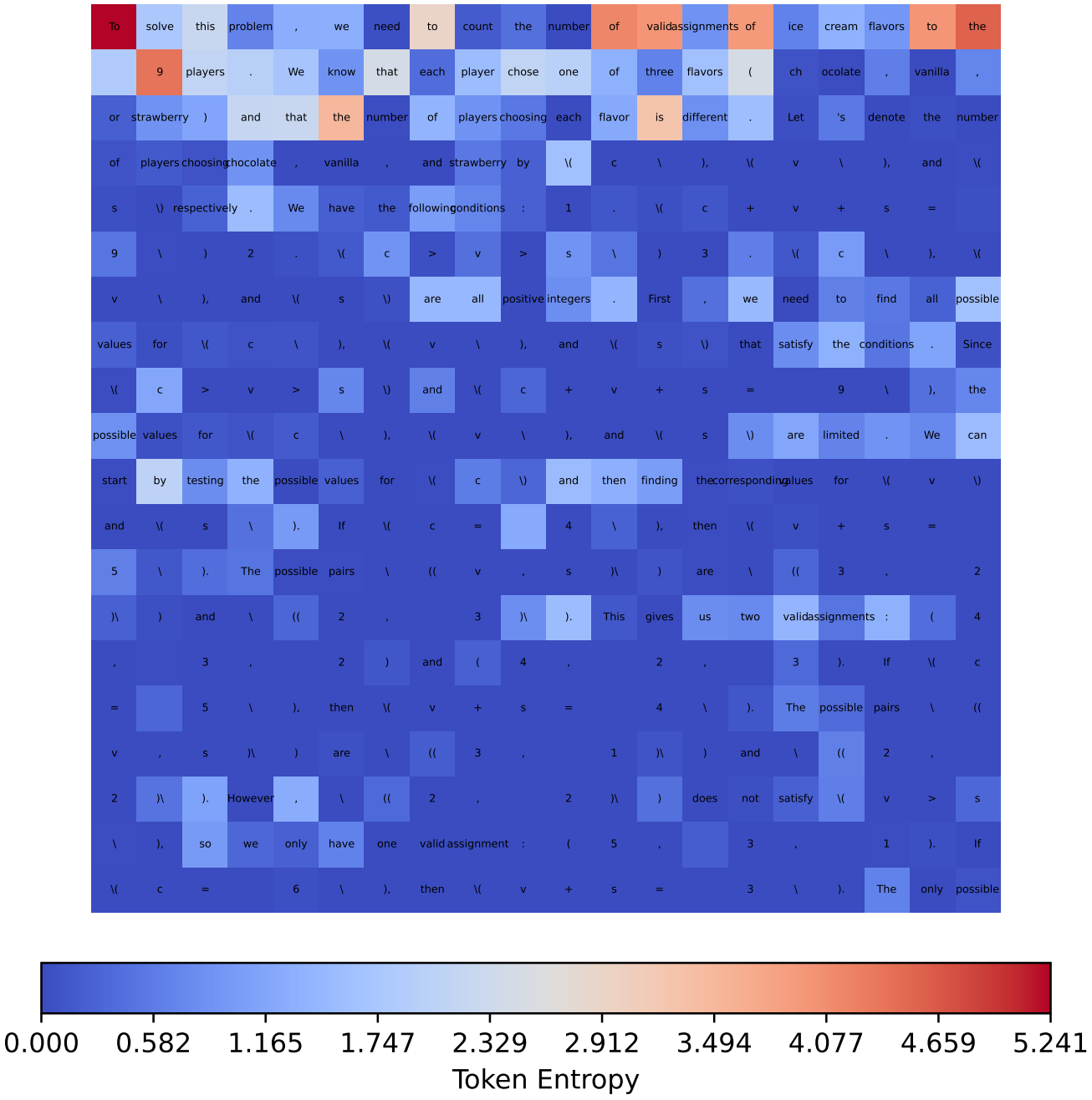


Figure 13: Random example 1 from AIME25 with InTRO

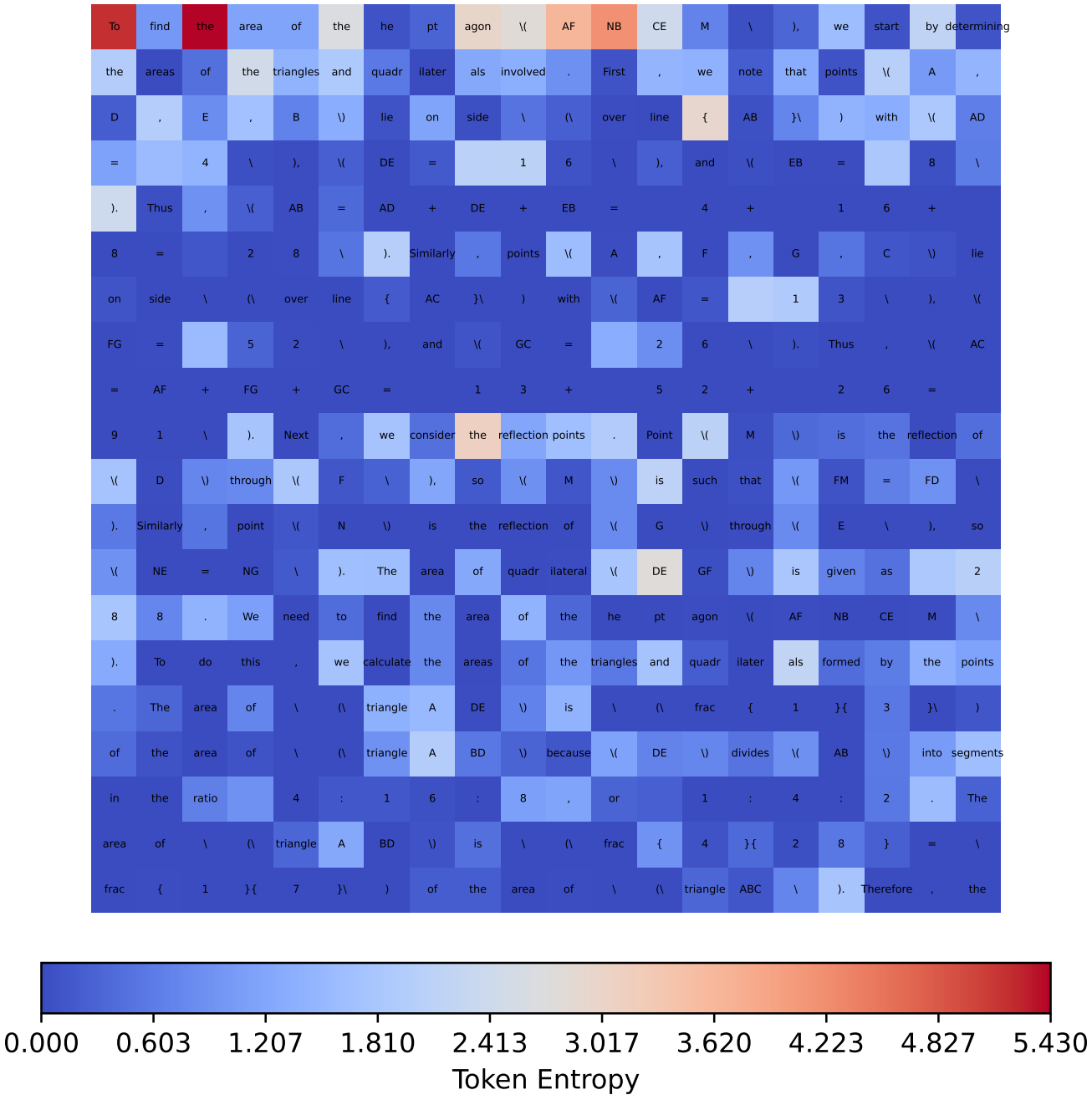


Figure 14: Random example 2 from AIME25 with GRPO

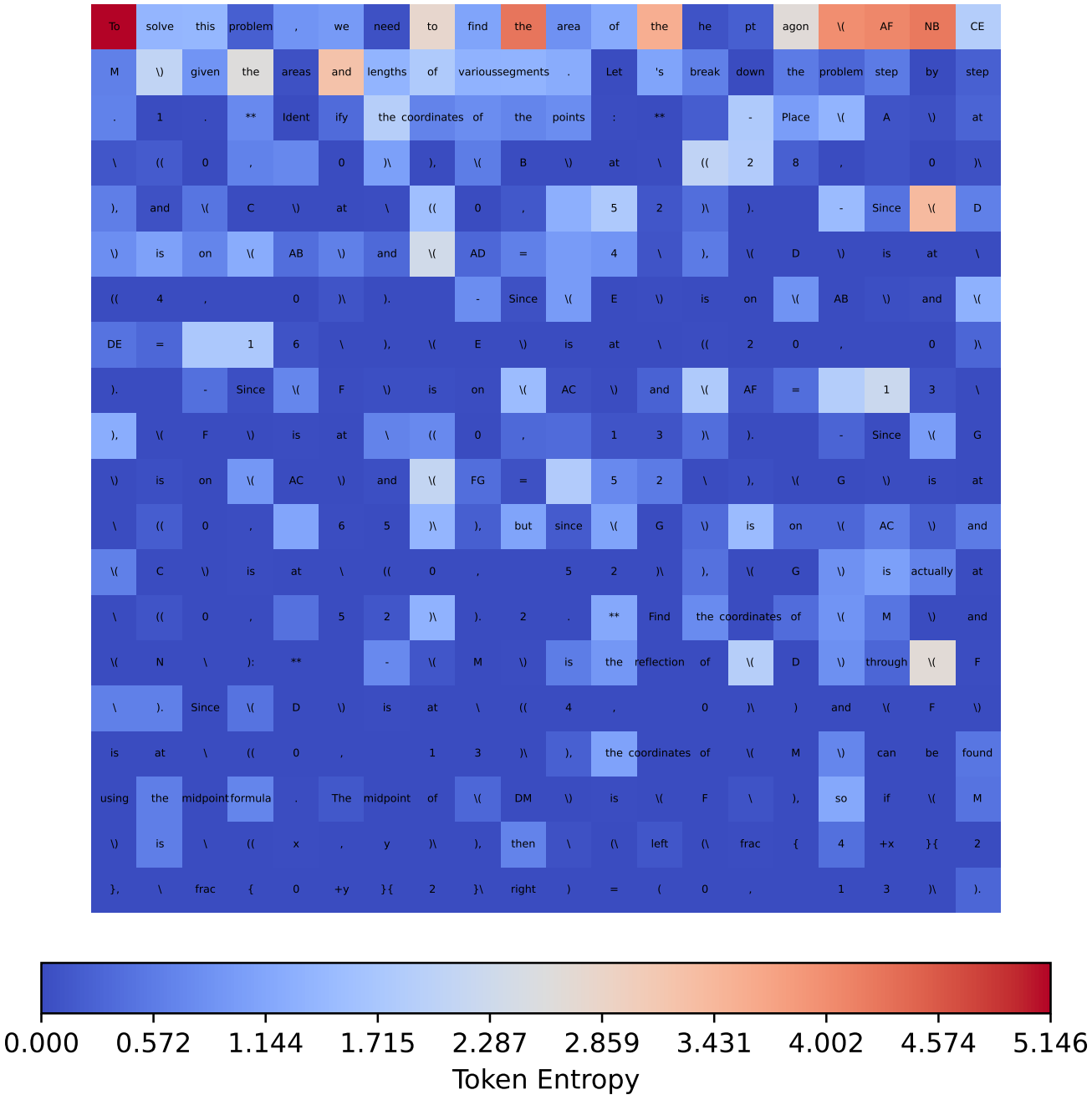


Figure 15: Random example 2 from AIME25 with InTRO