

Logit Arithmetic Elicits Long Reasoning Capabilities Without Training

Yunxiang Zhang* Muhammad Khalifa Lechen Zhang Xin Liu
Ayoung Lee Xinliang Frederick Zhang Farima Fatahi Bayat Lu Wang
University of Michigan

Abstract

Large reasoning models exhibit long chain-of-thought reasoning with complex strategies such as backtracking and self-verification. Yet, these capabilities typically require resource-intensive post-training. We investigate whether such behaviors can be elicited in large models without *any* gradient updates. To this end, we propose a decoding-time approach, **THINKLOGIT**, which utilizes logit arithmetic (Liu et al., 2024) to transfer these capabilities from a substantially smaller reasoning guider to a large non-reasoning target. We further show that we can boost performance by training the guider to correct the target’s errors using preference optimization over mixed model outputs, a setup we refer to as **THINKLOGIT-DPO**. We evaluate these methods across six reasoning benchmarks spanning math, science, and coding domains using the Qwen2.5-32B guided by R1-Distill-Qwen-1.5B, a model 21x smaller. Our experiments demonstrate that THINKLOGIT and THINKLOGIT-DPO achieve a relative improvement of 21.5% and 24.2%, respectively, over the target model. Moreover, THINKLOGIT remains effective even when the guider and target come from different model families. Crucially, our method requires zero training for the large model and would incur minimal inference overhead when logits are computed in parallel, presenting a practical solution for enabling long reasoning at scale.¹

1 Introduction

Large reasoning models (LRMs), such as OpenAI o1 (OpenAI, 2024) and DeepSeek-R1 (DeepSeek-AI et al., 2025), have significantly advanced reasoning by leveraging inference-time compute (Snell et al., 2024; Brown et al., 2024). These models generate very long chain-of-thought (CoT)

traces involving planning, reflection, and self-correction (Gandhi et al., 2025). It is widely believed that such behaviors require specialized training, either through reinforcement learning (RL) with verifiable rewards (DeepSeek-AI et al., 2025; Lambert et al., 2024; Shao et al., 2024) or supervised distillation (Muennighoff et al., 2025; Li et al., 2025) from other LRMs. However, such training is costly for large models with long generations and many parameters. Meanwhile, existing *training-free* long CoT elicitation methods (Pang et al., 2025; Muennighoff et al., 2025; Zou et al., 2023; Tang et al., 2025; Zhao et al., 2025) still remain limited, as they often lengthen outputs without reliably inducing complex reasoning behaviors and further require domain-specific supervision or white-box access. While the training costs are often *prohibitive* for large models, small models can be trained with modest compute (Dang and Ngo, 2025; Luo et al., 2025b). This observation motivates our central research question: *Can a small reasoning model elicit long CoT behavior in a large non-reasoning model at inference time, without training the large model?*

We address this question with a decoding-time method, **THINKLOGIT**, which elicits long CoT reasoning in a large non-reasoning model as the target. At each decoding step, we use logit arithmetic (Liu et al., 2024) by computing the logit difference between a small guider model trained for long reasoning and its base version, and add the resulting shift to the target logits. This token-by-token guidance transfers long reasoning signals from the small model to the large one without requiring any gradient updates of the target.

Furthermore, since the output distributions of long and short CoTs differ substantially, we align them by training the small guider to correct errors made by the target model while maintaining the strengths of the target model. This training process uses Direct Preference Optimiza-

* Correspondence to yunxiang@umich.edu

¹Our code is publicly available at <https://github.com/yunx-z/think-logit>.

tion (DPO; Rafailov et al., 2023) on mixed preference pairs sampled from both the guider and target models, thereby making THINKLOGIT more *on-policy* (Agarwal et al., 2024; Lu and Lab, 2025), and then applies logit arithmetic using the fine-tuned guider. We call this approach **THINKLOGIT-DPO** and show that it further boosts performance compared to THINKLOGIT.

We evaluate our methods on six challenging benchmarks covering mathematical, scientific, and code reasoning. Our results show that fusing the logits of a small reasoning model (R1-Distill-Qwen-1.5B) with those of a large target (Qwen2.5-32B) yields 21.5% and 24.2% relative performance gains with THINKLOGIT and THINKLOGIT-DPO, respectively, over the frozen target baseline. We further demonstrate that our method generalizes effectively across model families. Notably, a Qwen-based guider successfully drives long reasoning in Llama (Dubey et al., 2024), OLMo (OLMo et al., 2025), and EXAONE (Research et al., 2024) targets. Crucially, our approach requires zero training for the large model and incurs minimal inference latency overhead by computing the guider and target logits in parallel.

Our approach is particularly favorable when training the target is infeasible: *extreme-scale* models beyond practical fine-tuning budgets (Dettmers et al., 2023), *black-box* models limited to logit access (Ormazabal et al., 2023),² or *privacy-preserving* settings where an on-device guider steers a centralized model without exposing private data (McMahan et al., 2017; Xu et al., 2024). In these important scenarios, it enables efficient reasoning transfer without full-model fine-tuning.

To summarize, our main contributions are:

- We propose **THINKLOGIT**, a *training-free* decoding method that transfers long reasoning capabilities from a small guider to a large target using logit arithmetic (Section 3.1).
- We introduce **THINKLOGIT-DPO**, which further boosts performance by training the guider to correct target errors via preference optimization on mixed outputs (Section 3.2).
- We demonstrate up to 24.2% relative improvement across diverse domains (math, science, coding) and effective generalization to heterogeneous model families (Section 5.1 & 5.2).

²For example, the OpenAI API exposes a `logit_bias` parameter that lets users alter token probabilities at inference by boosting or suppressing specific tokens (OpenAI, 2025).

2 Related Work

Eliciting Long Chain-of-Thought Reasoning.

Standard approaches rely on **reinforcement learning** (Lambert et al., 2024; Shao et al., 2024; Yu et al., 2025; Liu et al., 2025) or **supervised fine-tuning** (Muennighoff et al., 2025; Xu et al., 2025a; Ye et al., 2025; Li et al., 2025) to elicit long CoT capabilities. However, applying these to large models incurs prohibitive computational costs. Alternatively, **training-free** methods exploit the fact that pretrained LLMs already exhibit latent long CoT behaviors (Liu et al., 2025; Gandhi et al., 2025). While techniques such as representation engineering (Zou et al., 2023; Tang et al., 2025) or neuron amplification (Zhao et al., 2025) attempt to surface these capabilities, they typically require white-box access and domain-specific supervision. In contrast, our **THINKLOGIT** framework operates without parameter access or task-specific constraints, effectively emulating the benefits of costly training in a flexible, black-box compatible manner.

Logit Arithmetic. Logit arithmetic blends output distributions from multiple models to steer generation (Liu et al., 2021; Ormazabal et al., 2023; Fan et al., 2024; Shi et al., 2024), and has been applied to emulating pretraining (Mitchell et al., 2024), task-specific fine-tuning (Liu et al., 2024; Fan et al., 2024), knowledge unlearning (Huang et al., 2025), and overriding safety filters (Zhao et al., 2024). However, prior applications typically target *local* attributes like vocabulary or style. In contrast, our setting demands modeling *long-range dependencies* to sustain complex cognitive behaviors like backtracking and verification over extended sequences (Gandhi et al., 2025). While *concurrent* work (Ouyang et al., 2025) also applies logit arithmetic to reasoning, they rely on direct fusion within the same model family. We advance this by introducing DPO alignment for targeted error correction and demonstrating robustness in *cross-family* transfer.

3 Methodology

In this section, we introduce **THINKLOGIT** (Section 3.1) and **THINKLOGIT-DPO** (Section 3.2) to elicit long CoT in frozen non-reasoning models via logit arithmetic and DPO.

3.1 THINKLOGIT

Let $\mathbf{z}_{1:t} = z_1, \dots, z_t$ be the partially decoded sequence of reasoning tokens at step t . For any lan-

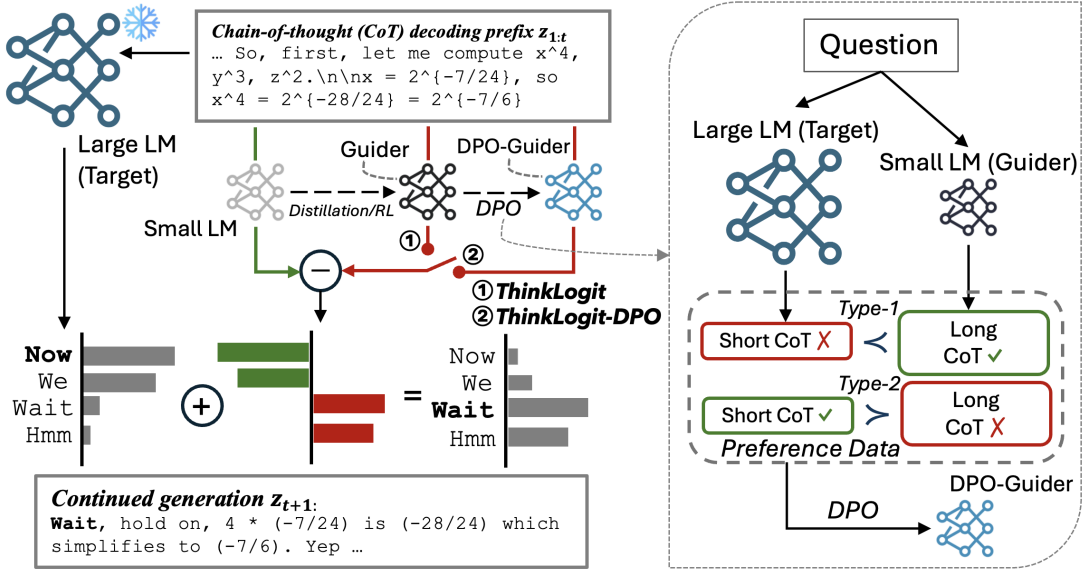


Figure 1: Overview of our proposed THINKLOGIT and THINKLOGIT-DPO approaches to elicit long chain-of-thought reasoning from a large non-reasoning model that is frozen.

guage model f , denote its pre-softmax logits at the next step by $\ell_{t+1}^{(f)} = f(\mathbf{z}_{1:t}) \in \mathbb{R}^{|\mathcal{V}|}$, where \mathcal{V} is the vocabulary. We assume three models during inference:

- **large (target) L** , a pre-trained LLM lacking long CoT capability;
- **small base S_0** , a pre-trained model without long reasoning fine-tuning;
- **small reasoning (guider) S** , obtained via long CoT post-training to S_0 .

As shown in Figure 1, at decoding step $t+1$, the fused logits are computed as $\tilde{\ell}_{t+1} = \ell_{t+1}^{(L)} + \alpha \cdot (\ell_{t+1}^{(S)} - \ell_{t+1}^{(S_0)})$, where $\alpha \geq 0$ controls the guidance strength. The delta term $\ell^{(S)} - \ell^{(S_0)}$ encodes the probability shift that turns a short-CoT model into a long-CoT one. In the illustrated example, while the target model favors shallow continuations like “Now” or “We,” THINKLOGIT shifts the top prediction to “Wait” by fusing the reasoning-oriented delta logits. This intervention triggers a reflective self-correction step (“Wait, hold on...”). In Appendix A, we provide a theoretical analysis justifying that our logit arithmetic formulation behaves approximately linearly across tasks.

3.2 THINKLOGIT-DPO

The effectiveness of THINKLOGIT can be constrained by distribution mismatches between the guider and target. To address this, we further train the small model as a stronger guider that corrects target reasoning errors while retaining the target

strengths, using a mixture of two types of preference pairs sampled from both the target L and the guider S outputs (see Figure 1):

Type-1: $(x, y^{L^\checkmark}, y^{S^\times})$ — The *large* model’s correct (short) CoT is preferred over the *small* model’s incorrect (long) one. This encourages the guider to preserve the correctness of the target model and avoid introducing new errors.

Type-2: $(x, y^{S^\checkmark}, y^{L^\times})$ — The *small* model’s correct (long) CoT is preferred over the *large* model’s incorrect (short) one, teaching the guider to be more confident at fixing the large model’s reasoning errors.

We gather these pairs from training queries x by independently sampling CoTs from L and S and labeling correctness based on the final answer. Let θ denote the parameters of the preference-optimized guider, initialized from S . We train θ with a DPO objective function that mixes the two pair types:

$$\mathcal{L}_{\text{DPO}}(\theta) = \lambda \mathbb{E}_{(x, y^{L^\checkmark}, y^{S^\times}) \sim \mathcal{D}_1} \ell_\theta(x; y^{L^\checkmark}, y^{S^\times}) + (1 - \lambda) \mathbb{E}_{(x, y^{S^\checkmark}, y^{L^\times}) \sim \mathcal{D}_2} \ell_\theta(x; y^{S^\checkmark}, y^{L^\times}), \quad (1)$$

where $\ell_\theta(x; y^+, y^-) = \log \sigma(r_\theta(x, y^+) - r_\theta(x, y^-))$, σ is the sigmoid function, $r_\theta(x, y) = \beta[\log \pi_\theta(y | x) - \log \pi_{\text{ref}}(y | x)]$ is the implicit reward of output y , and $\lambda \in [0, 1]$ balances the two datasets \mathcal{D}_1 (Type-1) and \mathcal{D}_2 (Type-2). We use $\lambda = \frac{|\mathcal{D}_1|}{|\mathcal{D}_1| + |\mathcal{D}_2|}$ by default, directly concatenating

Model	# Trainable Params	AIME 2024	AIME 2025	AMC 23	MATH Level 5	GPQA Diamond	LiveCode Bench v5
(Guider) R1-Distill-Qwen-1.5B	-	16.2	18.8	51.2	47.5	28.9	36.2
(Target) Qwen2.5-32B	-	14.6	8.3	57.2	44.7	36.9	48.3
No Fine-tuning of the Target							
Target + THINKLOGIT	0	22.5 +6.3	19.2 +0.3	62.2 +5.0	55.3 +7.8	41.8 +4.9	54.1 +5.8
Target + THINKLOGIT-DPO	78M	22.1 +5.9	21.7 +2.9	63.7 +6.5	58.5 +11.0	42.4 +5.5	52.4 +4.1
Full Fine-tuning of the Target							
s1.1-32B	32B	32.9	25.4	70.0	72.2	51.9	58.0
R1-Distill-Qwen-32B	32B	45.8	35.0	76.9	72.7	55.6	75.4

Table 1: Comparison of avg@8 performance across six reasoning benchmarks covering math, science, and coding domains. The yellow cells highlight the improvement of our methods over the stronger baseline model (**Target** or **Guider**). We show that THINKLOGIT and THINKLOGIT-DPO provide substantial gains over the baselines and partially recovers the benefits of full-model fine-tuning without any training of the large target model.

two datasets as DPO training data without further rebalancing. After fine-tuning, we replace S in THINKLOGIT with the optimized guider to obtain THINKLOGIT-DPO.

4 Experimental Setup

Benchmarks. We evaluate models on six widely used reasoning benchmarks for LRMs. Four of them are competition math problems sources from AIME2024, AIME2025, AMC23, and Level 5 hard problems from MATH-500 (Lightman et al., 2024). For scientific reasoning, we evaluate on GPQA Diamond (Rein et al., 2023), consisting of 198 PhD-level science questions in Biology, Chemistry, and Physics. For code reasoning, we use a 200-problem subset from the v5 release of LiveCodeBench (Jain et al., 2025). For each dataset, we independently sample 8 completions with a decoding temperature of 0.6 and maximum output length of 8192, and then compute their average accuracy (for math and science reasoning) or pass rate (for code reasoning) as **Avg@8** for our primary metric.

Models. Our primary setup pairs a **Qwen2.5-32B** (Yang et al., 2024a) target with an **R1-Distill-Qwen-1.5B** (DeepSeek-AI et al., 2025) guider. The guider is initialized from Qwen2.5-Math-1.5B (Yang et al., 2024b) and fine-tuned on distilled long CoTs from DeepSeek-R1 (DeepSeek-AI et al., 2025). The guider and target share the same Qwen tokenizer, which enables direct application of logit arithmetic. To demonstrate versatility, we also evaluate: (1) *Cross-family targets* (Section 5.2): We apply the Qwen-based guider to large non-reasoning models from three distinct model families: **Llama-3.3-70B-Instruct** (Dubey et al., 2024) built by Meta, **OLMo-2-0325-32B-**

Instruct (OLMo et al., 2025) built by Ai2, and **EXAONE-3.5-32B-Instruct** built by LG AI Research (Research et al., 2024). (2) *RL-trained guiders* (Section 5.3): We test guiders produced via RL rather than supervised distillation: **One-Shot-RLVR-1.5B** (Wang et al., 2025) and **DeepScaleR-1.5B-Preview** (Luo et al., 2025c). We set the guidance strength to $\alpha = 1$ by default.

Preference Data Construction. We construct our preference dataset using the level 4–5 subset of the MATH training set (Hendrycks et al., 2021). For each question, we sample completions from both the guider and target models to construct conflicting pairs where one model is correct and the other is incorrect. From the resulting pool of over 55k candidate pairs (spanning both Type-1 and Type-2 categories), we randomly select a subset of 10k pairs for DPO training. To save training compute, we apply LoRA (Hu et al., 2022) with a rank size of 64 for parameter-efficient fine-tuning of the guider model. Detailed statistics regarding completion counts and pair distributions are provided in Appendix B.

5 Experiment Results

5.1 Main Results

Table 1 presents the avg@8 accuracies for all systems. We highlight two key observations. *First*, THINKLOGIT *consistently enhances reasoning performance across math, science, and coding benchmarks, outperforming both the target and guider models*. Combining the logits of the 32B target with those of the 1.5B guider (THINKLOGIT) raises the average performance by 21.5% relative to the frozen target and by 28.3% relative to the guider.

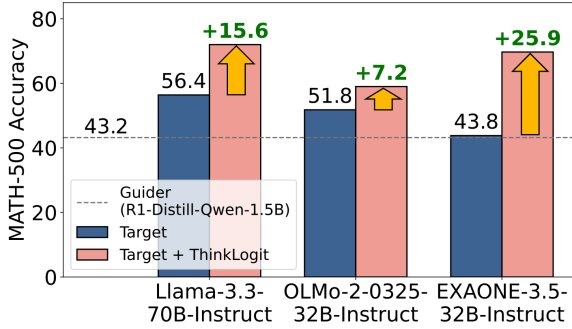


Figure 2: Cross-family long reasoning elicitation on MATH-500 (Lightman et al., 2024), applying a Qwen-based guider to Llama, OLMo, and EXAONE targets with greedy decoding. THINKLOGIT generalizes effectively across heterogeneous model families, delivering consistent accuracy gains (up to +25.9% absolute) despite architectural and tokenizer differences.

While THINKLOGIT-DPO performs better than THINKLOGIT by 3.7% relative in math, the improvement does not transfer to code reasoning. We attribute this to the fact that the DPO-trained guider was optimized exclusively on mathematical preference pairs. These results highlight that preference alignment is domain-dependent, suggesting that multi-domain training data is necessary to generalize these benefits to other reasoning tasks.

Second, our approach recovers a significant portion of the performance benefits of full-parameter fine-tuning without any training. Remarkably, THINKLOGIT bridges 45% of the performance gap to the data-efficient s1.1-32B (Muennighoff et al., 2025) baseline (trained on 1K examples) and 30% of the gap to the stronger R1-Distill-Qwen-32B (DeepSeek-AI et al., 2025) upper bound (trained on 800K examples). It achieves this by leveraging an off-the-shelf 1.5B guider, requiring zero gradient updates or data collection. This demonstrates that a substantial fraction of the reasoning capabilities typically acquired through expensive full-model training can instead be elicited at inference time by simply reusing existing open-weights reasoning models.

5.2 Generalization Across Model Families

In our main results, both the guider and target models come from the same Qwen model family. We now show that the small reasoning guider can also be applied to target models from different families. Although logit arithmetic typically assumes a shared tokenizer, we overcome this constraint by aligning vocabularies with a minimum edit distance

Model	AIME 2024	AIME 2025
Emulating RL on a Base Target		
(Guider) One-Shot-RLVR-1.5B	13.3	7.1
(Target) Qwen2.5-32B	14.6	8.3
Target + THINKLOGIT	17.5 +2.9	11.2 +2.9
Emulating RL on a Supervised Fine-Tuned Target		
(Guider) DeepScaleR-1.5B	30.0	23.8
(Target) R1-Distill-Qwen-32B	45.8	35.0
Target + THINKLOGIT	47.5 +1.7	37.9 +2.9

Table 2: Avg@8 performance of THINKLOGIT emulating reinforcement learning (RL) on large target models. The two small guiders are trained via RL. Both emulated RL pipelines deliver consistent performance gains while avoiding the prohibitive cost of applying RL training on large target models.

mapping (Wan et al., 2024). In practice, this is a one-time, offline step that introduces no inference overhead.

Using this strategy, we employ the R1-Distill-Qwen-1.5B to guide three target models: **Llama-3.3-70B-Instruct**, **OLMo-2-0325-32B-Instruct**, and **EXAONE-3.5-32B-Instruct**. As shown in Figure 2, THINKLOGIT consistently improves reasoning performance across all three heterogeneous targets on the MATH-500 benchmark, with the most notable gains observed on EXAONE-3.5-32B-Instruct, which achieves a 25.9% absolute accuracy increase. These results highlight the broad applicability of our approach: *a single specialized guider can be reused for larger models within or across families*, offering a scalable way to unlock long reasoning without retraining each target.

5.3 Emulation of Reinforcement Learning

RL with verifiable rewards is a powerful paradigm for enhancing the reasoning capabilities of language models (Lambert et al., 2024; Shao et al., 2024), but it is often prohibitively expensive. For example, training even a small 1.5B model with RL for long reasoning can require thousands of GPU hours (Luo et al., 2025b), an expense that becomes prohibitive for large-scale models. In Table 2, we investigate whether THINKLOGIT could emulate the effects of RL on a large target model without training it.

First, we simulate a **Zero-RL** (applying RL directly on a base model; DeepSeek-AI et al., 2025; Zeng et al., 2025) pipeline to enhance a large non-reasoning model, Qwen2.5-32B. We apply reason-



Figure 3: Comparison of THINKLOGIT against two training-free long CoT elicitation baselines: budget forcing and one-shot long CoT in-context learning (ICL). While these approaches increase verbosity, their accuracies are generally lower and can even degrade, whereas THINKLOGIT consistently produces longer reasoning that delivers the best performance.

ing from the One-Shot-RLVR-1.5B (Wang et al., 2025) guider, which is RL-trained on only one example from a Qwen2.5-Math-1.5B base. Second, we simulate an **SFT-then-RL** pipeline to further improve an already supervised fine-tuned reasoning model like R1-Distill-Qwen-32B. We apply guidance from the DeepScaleR-1.5B-Preview (Luo et al., 2025b) guider, which is trained via distributed RL from an R1-Distill-Qwen-1.5B base.

Table 2 shows that both emulated pipelines deliver consistent performance gains over the target large model and RL-trained small guiders, while *avoiding the prohibitive cost of applying RL training directly to the large target model*. This confirms that our method is an *orthogonal* technique that can directly benefit from advances in small-model post-training (e.g., distillation or RL), *offering a flexible and efficient mechanism to transfer the benefits of powerful but expensive training paradigms to larger models*.

5.4 Comparison with Training-Free Baselines

A natural question is whether existing training-free techniques are sufficient to elicit long CoTs, or if specialized methods are required. Figure 3 contrasts our approach against two such baselines. First, the budget-forcing³ heuristic (Muennighoff et al., 2025) replaces the end-of-sentence token with a placeholder string like “Wait” to artificially increase output length. Crucially, we find that further scaling the number of tokens in this base-

³For budget forcing, we report results using Qwen2.5-32B-Instruct, since applying it to the base Qwen2.5-32B led to low-quality outputs and degraded performance. This choice is consistent with the setup in Muennighoff et al. (2025).

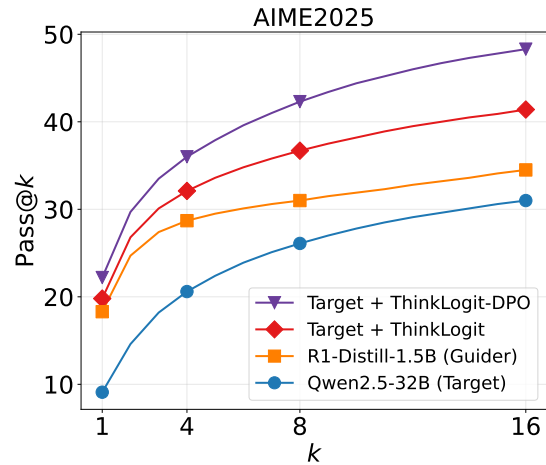


Figure 4: Inference-time scaling on AIME2025. $Pass@k$ for $k = 1-16$ comparing the target, guider, their direct logit fusion (THINKLOGIT), and the DPO-aligned fusion (THINKLOGIT-DPO). Our methods demonstrate superior sample efficiency, reaching stronger performance with fewer generations and maintaining larger gains as the sample budget increases.

line yields no additional accuracy gains; instead, the model frequently degenerates into repetitive loops—repeating the same conclusion without producing new information—rather than generating valid cognitive steps. In contrast, our method elicits genuine backtracking and verification, confirming that *mere verbosity does not translate into deeper reasoning*.

Second, inserting a single long CoT example sourced from the s1.1-1K dataset (Muennighoff et al., 2025) into the prompt for in-context learning (ICL; Brown et al., 2020; Min et al., 2022; Dong et al., 2024) also degrades performance despite longer outputs from the target model. In contrast, THINKLOGIT uses logit-level guidance from a small reasoning model to steer the decoding towards genuine long CoT reasoning, which translates into a clear uplift in the answer accuracy. This shows that our improvements stem from *the quality of the guidance* being applied, rather than *the quantity of tokens* generated.

5.5 Inference-Time Scaling Properties

We further evaluate the *scalability* of THINKLOGIT concerning both repeated sampling and guider capacity. Figure 4 shows that THINKLOGIT scales effectively with the number of generations per question. Test-time scaling via repeated sampling is a well-established strategy for enhancing reasoning

performance (Brown et al., 2024; Snell et al., 2024), as drawing more samples naturally increases the probability of finding a correct solution. We quantify this effect using the pass@ k metric (Chen et al., 2021), which measures the likelihood that at least one of k sampled outputs is correct. Figure 4 plots pass@ k for $k = 1-16$ on AIME2025, the benchmark where the 32B target is weakest and scaling effects are most pronounced.

Notably, both THINKLOGIT and THINKLOGIT-DPO surpass the target’s pass@16 accuracy using only four samples, demonstrating a fourfold increase in *sample efficiency*. This performance gap widens as k scales: at $k = 16$, the DPO-aligned guider outperforms the target by roughly 17 points. Ultimately, these results confirm that inference-time guidance reliably improves reasoning, with gains that compound alongside larger sampling budgets.

Although our primary configuration uses a 1.5B guider for efficiency, we demonstrate in Appendix C that performance consistently improves as guider size increases.

6 Additional Analyses

6.1 Ablation Study of THINKLOGIT-DPO

To further investigate the design choices in THINKLOGIT-DPO, we ablate both our mixed-pair data construction and preference-based learning objective (DPO) against single-source or supervised fine-tuning alternatives. Results in Table 3 answer the following research questions.

Are preference pairs sourced from both the target and the guider necessary to maximize performance? We construct the same amount of 10K preference pairs using only the guider’s correct vs. incorrect outputs, i.e., $(x, y^{S\checkmark}, y^{S\times})$. DPO on this data underperforms markedly on AMC23 (58.8 vs. 63.7 by our THINKLOGIT-DPO), confirming that mixing pairs which highlight *both* the target’s and guider’s strengths is crucial for maximal gains.

Is training on both types of pairs necessary for the effectiveness of THINKLOGIT-DPO? We next ablate by training on only one type of preference pairs at a time: using only Type-2 pairs $(x, y^{S\checkmark}, y^{L\times})$ (i.e., $\lambda = 0$ in Equation 1) yields an avg@8 of 57.2, while using only Type-1 pairs $(x, y^{L\checkmark}, y^{S\times})$ (i.e., $\lambda = 1$ in Equation 1) drops further to 51.9. Both are substantially below the 63.7 achieved by the full mixture, indicating that

Model	Training Data for Guider	Avg@8
THINKLOGIT-DPO		
Ours	$(x, y^{L\checkmark}, y^{S\times}), (x, y^{S\checkmark}, y^{L\times})$	63.7
w/o dual sources	$(x, y^{S\checkmark}, y^{S\times})$	58.8
w/o Type-1 pairs	$(x, y^{S\checkmark}, y^{L\times})$	57.2
w/o Type-2 pairs	$(x, y^{L\checkmark}, y^{S\times})$	51.9
THINKLOGIT-SFT		
learning from target	$(x, y^{L\checkmark})$	44.7
self-learning	$(x, y^{S\checkmark})$	55.6
learning from teacher	$(x, y^{R1\checkmark})$	60.9

Table 3: Avg@8 on AMC23 under ablations of guider’s training data and objectives in THINKLOGIT-DPO. $x, y^{L\checkmark}$, and $y^{S\times}$ denote the question, the correct (\checkmark) response for the large target model (L), and the incorrect (\times) response from the small guider model (S), respectively. Our dual-source, mixed-pair DPO performs the best, demonstrating the necessity of complementary preference signals and preference-based alignment.

both Type-2 pairs (which teach the guider to correct target errors) and Type-1 pairs (which enforce preservation of the correct reasoning of the target) are necessary for better alignment.

Can supervised fine-tuning replace preference-based alignment of the guider?

We evaluate SFT against DPO by training the guider on three equally sized sets of high-quality completions: **Option 1:** the target model’s correct outputs $y^{L\checkmark}$; **Option 2:** the guider’s own correct outputs $y^{S\checkmark}$ (also known as rejection-sampling fine-tuning (Yuan et al., 2023)); **Option 3:** R1-distilled completions $y^{R1\checkmark}$. Although SFT on Options 1 and 2 makes the guider a stronger *standalone* reasoner, none of these variants match the performance of the DPO-aligned guider. This gap highlights that *optimizing with pairwise preference comparisons yields a better guider than optimizing solely for correctness*. While SFT on Option 3 adapts the guider toward the target’s short CoT reasoning style and thus reduces the distributional gap, it also tends to overwrite the guider’s native strengths of long reasoning. In contrast, DPO *preserves the guider’s intrinsic reasoning capabilities while selectively aligning it to the target’s preferences* through pairwise comparisons.

We also ablate the guidance strength α in Appendix G, where we observe that the default value $\alpha = 1.0$ performs best. We applied the exact same hyperparameter configuration across all main ex-

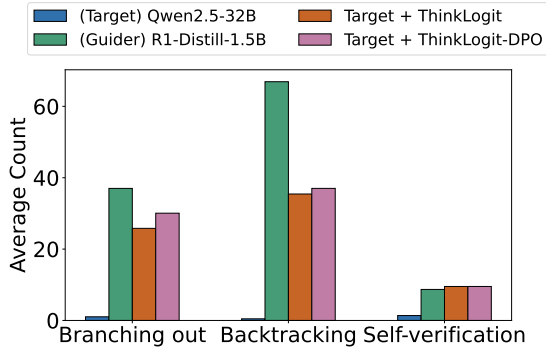


Figure 5: Average frequency of key reasoning behaviors (Branching out, Backtracking, and Self-verification) across all six benchmarks. THINKLOGIT and THINKLOGIT-DPO successfully elicit complex cognitive patterns in the target non-reasoning model, effectively transferring the long reasoning capabilities of the small guider.

periments, encompassing six benchmarks and six guider-target model pairs. This required zero per-task or per-model tuning, confirming the strong out-of-the-box robustness of our method.

6.2 Reasoning Behavior Analysis

To verify that THINKLOGIT and THINKLOGIT-DPO elicit genuine long reasoning capabilities rather than simply increasing output verbosity, we analyze the frequency of tokens associated with three key reasoning behaviors: **Branching out** (exploring alternative approaches), **Backtracking** (correcting errors or revising steps), and **Self-verification** (systematic checking of intermediate results).

To quantify these behaviors, we curate a lexicon of representative keywords based on prior studies (Gandhi et al., 2025; Ouyang et al., 2025). We process the model reasoning traces with the spaCy (Honnibal et al., 2020) toolkit using tokenization and lemmatization to account for morphological variations, then calculate the frequency of keywords associated with each reasoning behavior. The full list of keywords used for this analysis is provided in Table 6 of Appendix D.

We present the average frequency of these behaviors across all six datasets in Figure 5, which reveals a stark contrast before and after applying THINKLOGIT. The target model (Qwen2.5-32B) exhibits negligible occurrences of these reasoning behaviors, which is consistent with its inability to perform long chain-of-thought reasoning. In con-

	Target	THINKLOGIT	SFT
Training Efficiency			
Training Parameters	–	0	32B
# Training GPUs	–	0	8×80GB*
Inference Efficiency			
Inference Parameters	32B	35B	32B
# Inference GPUs	2×46GB	2×46GB	2×46GB
Inference Throughput (tokens/second)	10.1	7.6	9.1

Table 4: Comparison of training and inference efficiency, using Qwen2.5-32B as the Target model. We select R1-Distill-Qwen-32B as a representative model for full-parameter supervised fine-tuning (SFT). Training GPUs required for SFT (*) is estimated based on Zheng et al. (2024). THINKLOGIT trades increased inference latency for *zero* training cost, making long reasoning possible when training the large target is infeasible.

trast, the guider model (R1-Distill-1.5B) demonstrates a high frequency of reasoning-related tokens, particularly for Backtracking and Branching out. THINKLOGIT and THINKLOGIT-DPO successfully transfer these behaviors to the large target model, eliciting substantial amounts of branching, backtracking, and self-verification, effectively bridging the gap between the non-reasoning target and the specialized guider. We further break down these behaviors by domain in Figure 7 (see Appendix D). The results show consistent trends across math, science, and coding tasks.

We provide an example model output after applying THINKLOGIT in Figure 6, showing that THINKLOGIT elicits rich long CoT behaviors like backtracking and self-verification rather than merely increasing verbosity. In Appendix F, we further analyze specific failure modes, observing that THINKLOGIT can struggle to form higher-level abstractions, hallucinates contradictions that disrupt inductive reasoning, or fail to recover from wrong assumptions.

6.3 Efficiency Analysis

Training Efficiency. As illustrated in Table 4, THINKLOGIT entirely eliminates the training overhead associated with adapting large non-reasoning models for long reasoning. While full parameter supervised fine-tuning (SFT) of the 32B target demands significant computational resources (estimated at eight 80GB GPUs), THINKLOGIT requires *zero* training for the target model itself. This makes THINKLOGIT distinctively advantageous in resource-constrained settings where training a 32B

Question: Let $P(x)$ be a monic polynomial of degree 3. Suppose that $P(x)$ has remainder $R(x)$ when it is divided by $(x - 1)(x - 4)$, and remainder $2R(x)$ when it is divided by $(x - 2)(x - 3)$. Given that $P(0) = 5$, find $P(5)$.
Answer: 15.

(Guider) R1-Distill-Qwen-1.5B	(Target) Qwen2.5-32B	Target + THINKLOGIT
<p>Alright, so I have this problem here about a monic polynomial of degree 3, $P(x)$. [...]</p> <p>So, $P(x) = x^3 + a x^2 + b x + c = x^3 - 7x^2 + 29x + 5$</p> <p>Let me double-check to make sure. First, let's compute $P(1)$: [...]</p> <p>$R(3) = 15 \cdot 3 + 13 = 45 + 13 = 58$. Wait, that's 56 vs 58. Hmm, that's a problem.</p> <p>Wait, hold on, maybe I made a mistake in calculations.</p> <p>Let me recalculate $P(3)$: [...]</p> <p>Wait, earlier when I thought I had a discrepancy, I must have miscalculated. [...]</p> <p>**Final Answer** The value of $P(5)$ is 100.</p>	<p>Given the problem, we need to find the polynomial $P(x)$ and then use it to determine $P(5)$. Let's break down the problem step-by-step.</p> <ol style="list-style-type: none"> **Understand the problem:** [...] **Formulate the polynomial:** [...] **Use the given conditions:** [...] **Solve for the coefficients a, b, and c: [...] <p>The value of $P(5)$ is 55.</p>	<p>Let's write a Python program to solve it.</p> <pre>import sympy as sp [...]</pre> <p>Wait, this is not going to work. Because Python's <code>sympy.rem()</code> function doesn't accept the polynomial in a string form. Also, the way I defined the equations is incorrect.</p> <p>Maybe I should approach this problem differently. [...]</p> <p>Wait, no. Let's first express Equation 1 as: $r + s = a + b + 6$. [...]</p> <p>Simplify: $4a + b + 20.5 = 0$. Hmm, fractional coefficient. Not sure if that's intended.</p> <p>Wait, perhaps I made a mistake earlier. Wait, let me double-check the substitution into Equation 3: [...]</p> <p>Yep, $P(5) = 15$.</p> <p>**Final Answer** 15</p>

Figure 6: Example model outputs. We show a question from MATH500, where both the **Guider** and **Target** provide **incorrect answers** but our **THINKLOGIT** generates the **correct answer**. [...] indicates that the text was trimmed to fit this page, but the generated text is actually longer. We highlight text that exhibits long CoT behaviors (e.g., backtracking, branching out, self-verification) in **yellow**. THINKLOGIT elicits informative long CoT reasoning behaviors from the non-reasoning target model.

model is computationally infeasible.

Inference Efficiency. Table 4 further compares the inference metrics of THINKLOGIT against the frozen target and the fully fine-tuned baseline. Deploying THINKLOGIT requires hosting the large target alongside two smaller models: the base model S_0 and the reasoning guider S . In our primary setup (a 32B target guided by a 1.5B model), this increases the total parameter count by approximately $1.1 \times$ (from 32B to 35B) relative to the target alone. Crucially, this combined footprint still fits within the same hardware configuration ($2 \times 46\text{GB}$ GPUs), avoiding the need for extra inference hardware.

Regarding inference throughput (tokens generated per second), profiling on NVIDIA L40S GPUs shows a moderate reduction of roughly 25% (10.1 vs. 7.6 tokens/second) compared to the target in isolation. Notably, THINKLOGIT-DPO introduces no additional overhead beyond THINKLOGIT, as it simply replaces the guider with a preference-optimized model of identical size. The observed throughput reduction primarily stems from our pro-

totype implementation based on the Huggingface `generate()` function, which queries the three models sequentially at each decoding step. In a production environment, logits from all three models could be computed *concurrently* across distributed GPUs, which would largely mitigate this latency and make throughput comparable to that of the standalone target model.

7 Conclusion

We present THINKLOGIT, a training-free framework eliciting long chain-of-thought via logit guidance from a small reasoner, and its optimized variant THINKLOGIT-DPO. Delivering up to 24.2% gains with a $21 \times$ smaller guider, our approach generalizes across model families and emulates RL benefits. These results establish inference-time guidance as a practical alternative to post-training, enabling modular systems where specialized, lightweight models unlock advanced reasoning in large-scale foundation models.

Limitations

Inference Overhead. While THINKLOGIT eliminates the prohibitive cost of training large models, it introduces a modest inference overhead by requiring the simultaneous execution of the small guider and base models. In our current sequential implementation, this results in a throughput reduction of approximately 25% compared to the standalone target. However, as discussed in Section 6.3, *this latency is primarily an engineering bottleneck rather than a fundamental one brought by our methodology*. Since the additional model weights fit within the same hardware footprint, production deployments can compute guider logits in parallel to achieve throughput comparable to the target model alone.

Furthermore, *cross-model versatility multiplies compute savings for training*. Unlike standard fine-tuning, which requires independent training for each target, our approach offers high reusability. We demonstrate that a single 1.5B Qwen guider effectively steers models from diverse families such as Llama, OLMo, and EXAONE (Figure 2). This one-to-many capability multiplies the training compute saved across all guided models.

Finally, *new methods that reduce overthinking and yield shorter reasoning traces will directly lower our per-token overhead*. Recent methods actively reduce overthinking to generate shorter reasoning traces (Xu et al., 2025b; Song and Zheng, 2025; Luo et al., 2025a). This shift is also visible in practice, with Anthropic substantially lowering Claude’s reasoning verbosity between model iterations (Zhou et al., 2025). Because our overhead scales linearly with token count, these shorter reasoning trajectories naturally shrink the absolute inference cost of our method.

Offline Alignment. The guider is aligned with the target via DPO on a fixed set of preference pairs. This *offline* formulation cannot adapt once deployment uncovers new error patterns or distribution drift. Incorporating *online* reinforcement learning (Schulman et al., 2017; Shao et al., 2024) that updates the guider from streamed on-policy samples could, in principle, reduce this brittleness. However, on-policy RL introduces training efficiency and stability challenges that remain open research problems.

Acknowledgments

This work is supported in part by LG AI Research, Cisco Research, National Science Foundation through grant 2046016, Air Force Office of Scientific Research under grant FA9550-22-1-0099, and computational resources and services provided by Advanced Research Computing (ARC), a division of Information and Technology Services (ITS) at the University of Michigan, Ann Arbor. We thank ARR reviewers for their useful feedback. We also thank the members of the LAUNCH group at the University of Michigan for their discussions and suggestions.

References

- Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. 2024. [On-policy distillation of language models: Learning from self-generated mistakes](#). In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.
- Bradley C. A. Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. 2024. [Large language monkeys: Scaling inference compute with repeated sampling](#). *CoRR*, abs/2407.21787.
- Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. [Language models are few-shot learners](#). In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.
- Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, and 39 others. 2021. [Evaluating large language models trained on code](#). *CoRR*, abs/2107.03374.
- Quy-Anh Dang and Chris Ngo. 2025. Reinforcement learning for reasoning in small llms: What works and what doesn’t. *arXiv preprint arXiv:2503.16219*.
- DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, and 81 others.

2025. [Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning](#). *CoRR*, abs/2501.12948.
- Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. [Qlora: Efficient finetuning of quantized llms](#). In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.
- Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Baobao Chang, Xu Sun, Lei Li, and Zhifang Sui. 2024. [A survey on in-context learning](#). In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 1107–1128. Association for Computational Linguistics.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and 82 others. 2024. [The llama 3 herd of models](#). *CoRR*, abs/2407.21783.
- Chenghao Fan, Zhenyi Lu, Wei Wei, Jie Tian, Xiaoye Qu, Danyang Chen, and Yu Cheng. 2024. [On giant’s shoulders: Effortless weak to strong by dynamic logits fusion](#). In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.
- Kanishk Gandhi, Ayush Chakravarthy, Anikait Singh, Nathan Lile, and Noah D. Goodman. 2025. [Cognitive behaviors that enable self-improving reasoners, or, four habits of highly effective stars](#). *CoRR*, abs/2503.01307.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. [Measuring mathematical problem solving with the MATH dataset](#). In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.
- Matthew Honnibal, Ines Montani, Sofie Van Lan-deghem, and Adriane Boyd. 2020. [spaCy: Industrial-strength Natural Language Processing in Python](#).
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. [Lora: Low-rank adaptation of large language models](#). In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.
- James Y. Huang, Wenxuan Zhou, Fei Wang, Fred Morstatter, Sheng Zhang, Hoifung Poon, and Muhao Chen. 2025. [Offset unlearning for large language models](#). *Trans. Mach. Learn. Res.*, 2025.
- Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2025. [Livecodebench: Holistic and contamination free evaluation of large language models for code](#). In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.
- Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, and 4 others. 2024. [Tulu 3: Pushing frontiers in open language model post-training](#). *CoRR*, abs/2411.15124.
- Dacheng Li, Shiyi Cao, Tyler Griggs, Shu Liu, Xiangxi Mo, Eric Tang, Sumanth Hegde, Kourosh Hakhmaneshi, Shishir G. Patil, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. 2025. [Llms can easily learn to reason from demonstrations structure, not content, is what matters!](#) *CoRR*, abs/2502.07374.
- Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. [Let’s verify step by step](#). In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.
- Alisa Liu, Xiaochuang Han, Yizhong Wang, Yulia Tsvetkov, Yejin Choi, and Noah A. Smith. 2024. [Tuning language models by proxy](#). *CoRR*, abs/2401.08565.
- Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A. Smith, and Yejin Choi. 2021. [Dexperts: Decoding-time controlled text generation with experts and anti-experts](#). In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 6691–6706. Association for Computational Linguistics.
- Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025. [Understanding r1-zero-like training: A critical perspective](#). *arXiv preprint arXiv:2503.20783*.
- Kevin Lu and Thinking Machines Lab. 2025. [On-policy distillation](#). *Thinking Machines Lab: Connectionism*.
- Feng Luo, Yu-Neng Chuang, Guanchu Wang, Hoang Anh Duy Le, Shaochen Zhong, Hongyi Liu, Jiayi Yuan, Yang Sui, Vladimir Braverman, Vipin Chaudhary, and Xia Hu. 2025a. [Autol2s: Auto long-short reasoning for efficient large language models](#). *CoRR*, abs/2505.22662.

- Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. 2025b. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. *Notion Blog*. Accessed 2025.
- Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Li Erran Li, and 1 others. 2025c. Deepscaler: Surpassing o1-preview with a 1.5 b model by scaling rl. *Notion Blog*.
- Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. [Communication-efficient learning of deep networks from decentralized data](#). In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR.
- Sewon Min, Xinxu Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. [Rethinking the role of demonstrations: What makes in-context learning work?](#) In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 11048–11064. Association for Computational Linguistics.
- Eric Mitchell, Rafael Rafailov, Archit Sharma, Chelsea Finn, and Christopher D. Manning. 2024. [An emulator for fine-tuning large language models using small language models](#). In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.
- Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel J. Candès, and Tatsunori Hashimoto. 2025. [s1: Simple test-time scaling](#). *CoRR*, abs/2501.19393.
- Team OLMO, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, and 21 others. 2025. [2 olmo 2 furious](#). *CoRR*, abs/2501.00656.
- OpenAI. 2024. [Learning to reason with LLMs](#).
- OpenAI. 2025. [Using logit bias to alter token probability with the openai api](#).
- Aitor Ormazabal, Mikel Artetxe, and Eneko Agirre. 2023. [Comblm: Adapting black-box language models through small fine-tuned models](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 2961–2974. Association for Computational Linguistics.
- Siru Ouyang, Xinyu Zhu, Zilin Xiao, Minhao Jiang, Yu Meng, and Jiawei Han. 2025. [RAST: reasoning activation in llms via small-model transfer](#). *CoRR*, abs/2506.15710.
- Bo Pang, Hanze Dong, Jiacheng Xu, Silvio Savarese, Yingbo Zhou, and Caiming Xiong. 2025. [BOLT: bootstrap long chain-of-thought in language models without distillation](#). *CoRR*, abs/2502.03860.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. [Direct preference optimization: Your language model is secretly a reward model](#). In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. [GPQA: A graduate-level google-proof q&a benchmark](#). *CoRR*, abs/2311.12022.
- LG AI Research, Soyoung An, Kyunghoon Bae, Eunbi Choi, Kibong Choi, Stanley Jungkyu Choi, Seokhee Hong, Junwon Hwang, Hyojin Jeon, Gerrard Jeongwon Jo, Hyunjik Jo, Jiyeon Jung, Yountae Jung, Hyosang Kim, Joonkee Kim, Seonghwan Kim, Soyeon Kim, Sunkyoung Kim, Yireun Kim, and 14 others. 2024. [EXAONE 3.5: Series of large language models for real-world use cases](#). *CoRR*, abs/2412.04862.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. [Proximal policy optimization algorithms](#). *CoRR*, abs/1707.06347.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. [Deepseekmath: Pushing the limits of mathematical reasoning in open language models](#). *CoRR*, abs/2402.03300.
- Weijia Shi, Xiaochuang Han, Mike Lewis, Yulia Tsvetkov, Luke Zettlemoyer, and Wen-tau Yih. 2024. [Trusting your evidence: Hallucinate less with context-aware decoding](#). In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Short Papers, NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 783–791. Association for Computational Linguistics.
- Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. [Scaling LLM test-time compute optimally can be more effective than scaling model parameters](#). *CoRR*, abs/2408.03314.
- Mingyang Song and Mao Zheng. 2025. [Walk before you run! concise LLM reasoning via reinforcement learning](#). *CoRR*, abs/2505.21178.
- Xinyu Tang, Xiaolei Wang, Zhihao Lv, Yingqian Min, Wayne Xin Zhao, Binbin Hu, Ziqi Liu, and Zhiqiang

- Zhang. 2025. Unlocking general long chain-of-thought reasoning capabilities of large language models via representation engineering. *arXiv preprint arXiv:2503.11314*.
- Fanqi Wan, Xinting Huang, Deng Cai, Xiaojun Quan, Wei Bi, and Shuming Shi. 2024. [Knowledge fusion of large language models](#). In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.
- Yiping Wang, Qing Yang, Zhiyuan Zeng, Liliang Ren, Lucas Liu, Baolin Peng, Hao Cheng, Xuehai He, Kuan Wang, Jianfeng Gao, and 1 others. 2025. Reinforcement learning for reasoning in large language models with one training example. *arXiv preprint arXiv:2504.20571*.
- Haotian Xu, Xing Wu, Weinong Wang, Zhongzhi Li, Da Zheng, Boyuan Chen, Yi Hu, Shijia Kang, Jiaming Ji, Yingying Zhang, Zhijiang Guo, Yaodong Yang, Muhan Zhang, and Debing Zhang. 2025a. [Redstar: Does scaling long-cot data unlock better slow-reasoning systems?](#) *CoRR*, abs/2501.11284.
- Jiajun Xu, Zhiyuan Li, Wei Chen, Qun Wang, Xin Gao, Qi Cai, and Ziyuan Ling. 2024. [On-device language models: A comprehensive review](#). *CoRR*, abs/2409.00088.
- Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. 2025b. [Chain of draft: Thinking faster by writing less](#). *CoRR*, abs/2502.18600.
- An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jixi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024a. [Qwen2.5 technical report](#). *CoRR*, abs/2412.15115.
- An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. 2024b. [Qwen2.5-math technical report: Toward mathematical expert model via self-improvement](#). *CoRR*, abs/2409.12122.
- Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. 2025. [LIMO: less is more for reasoning](#). *CoRR*, abs/2502.03387.
- Qiyang Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, and 16 others. 2025. [DAPO: an open-source LLM reinforcement learning system at scale](#). *CoRR*, abs/2503.14476.
- Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Chuanqi Tan, and Chang Zhou. 2023. [Scaling relationship on learning mathematical reasoning with large language models](#). *CoRR*, abs/2308.01825.
- Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. 2025. [Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild](#). *CoRR*, abs/2503.18892.
- Xuandong Zhao, Xianjun Yang, Tianyu Pang, Chao Du, Lei Li, Yu-Xiang Wang, and William Yang Wang. 2024. [Weak-to-strong jailbreaking on large language models](#). *CoRR*, abs/2401.17256.
- Zekai Zhao, Qi Liu, Kun Zhou, Zihan Liu, Yifei Shao, Zhiting Hu, and Biwei Huang. 2025. Activation control for efficiently eliciting long chain-of-thought ability of language models. *arXiv preprint arXiv:2505.17697*.
- Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyang Luo, Zhangchi Feng, and Yongqiang Ma. 2024. [Llamafactory: Unified efficient fine-tuning of 100+ language models](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand. Association for Computational Linguistics.
- Wei Zhou, AJ Kourabi, and Dylan Patel. 2025. [Deepseek debrief: >128 days later](#). SemiAnalysis.
- Andy Zou, Long Phan, Sarah Li Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, and 2 others. 2023. [Representation engineering: A top-down approach to AI transparency](#). *CoRR*, abs/2310.01405.
- Haosheng Zou, Xiaowei Lv, Shousheng Jia, and Xiangzheng Zhang. 2024. [360-llama-factory](#).

A Theoretical Grounding

We provide a clearer mathematical interpretation for why our logit arithmetic formulation behaves approximately linearly across tasks.

Consider a pretrained model (S_0) and its post-trained long-CoT variant (S), which maximize an RL objective with a KL divergence penalty:

$$\max_{P(\cdot|x)} \mathbb{E}_{y \sim P} [r(x, y)] - \beta KL(P(\cdot|x) \| S_0(\cdot|x)).$$

This is known to have a closed-form solution (Rafailov et al., 2023):

$$P^*(y|x) = \frac{1}{Z(x)} S_0(y|x) \exp\left(\frac{1}{\beta} r(x, y)\right).$$

Rearranging gives an implicit reward function for any fine-tuned model S :

$$r(x, y) = \beta \log \frac{S(y|x)}{S_0(y|x)}.$$

Thus, the log-likelihood ratio between a trained and untrained model encodes behavioral shift introduced by long-CoT training.

Now, in ThinkLogit we combine the target model L with this reward signal via $\tilde{\ell} = \ell_L + \alpha \cdot (\ell_S - \ell_{S_0})$.

Exponentiating both sides gives:

$$\tilde{P}(y|x) \propto L(y|x) \left(\frac{S(y|x)}{S_0(y|x)} \right)^\alpha,$$

which corresponds to adjusting the target distribution according to the guider’s implicit reward. In this view, the apparent linearity of logit arithmetic follows naturally from the multiplicative form of the KL-regularized optimum and the fact that logits are log-probabilities, rather than from an ad-hoc linearity assumption.

B Training Details

Environment. All experiments were conducted using NVIDIA A40/L40S GPUs with 48GB memory. The software environment was configured as follows:

- 360-LLaMA-Factory (Zou et al., 2024) (A long CoT adapted version of LLaMA-Factory 0.9.1 (Zheng et al., 2024))
- torch 2.7.0
- transformers 4.51.3

- accelerate 1.0.1
- datasets 3.1.0
- trl 0.9.6
- peft 0.12.0
- deepspeed 0.14.4

LoRA Configuration. We applied LoRA (Hu et al., 2022) for parameter-efficient fine-tuning of the guider model:

- Rank: 64
- α_{LoRA} : 128
- Target modules: q_proj, k_proj, v_proj, o_proj
- Bias: None

DPO Training. For preference optimization with DPO, we used the following settings:

- Batch size: 32 (4 GPUs * 8 Gradient Accumulation)
- Epoch: 1
- Learning rate: 5e-6
- Optimizer: AdamW
- Learning rate scheduler: cosine with warmup
- Warmup ratio: 0.1
- β (reward scaling): 0.1
- Cutoff length: 8192

Preference Data Statistics. To construct the preference data, we sampled 5 completions for each question from both the guider (S) and target (L) models. Each completion was checked for final-answer correctness against the gold label. The target model L yielded 12,412 correct (y^{L^\checkmark}) and 16,448 incorrect (y^{L^\times}) completions, whereas the guider S produced 18,651 correct (y^{S^\checkmark}) and 10,209 incorrect (y^{S^\times}) completions. Forming the Cartesian product for each question resulted in 11,974 Type-1 preference pairs ($y^{L^\checkmark}, y^{S^\times}$) and 43,209 Type-2 pairs ($y^{S^\checkmark}, y^{L^\times}$), for a total pool of 55,183 pairs.

C Impact of Guider Model Size

A key hypothesis of THINKLOGIT is that the quality of the elicited reasoning depends on the strength of the guidance signal provided by the small model. In our main experiments, we utilized a 1.5B parameter guider for efficiency. To investigate whether our approach benefits from scaling the guider’s capacity, we replace the 1.5B guider with a larger 7B reasoning model (R1-Distill-Qwen-7B), while keeping the target model (Qwen2.5-32B) fixed.

Model	AIME 2024	AMC 23
(Target) Qwen2.5-32B	14.6	57.2
Target + THINKLOGIT 1.5B	22.5 +7.9	62.2 +5.0
Target + THINKLOGIT 7B	35.8 +21.2	69.9 +12.7

Table 5: Scaling properties of the guider model. Avg@8 performance improves substantially when replacing the 1.5B guider with a 7B model, indicating that THINKLOGIT effectively leverages increased capacity in the guider model.

Table 5 presents the results on AIME 2024 and AMC 23. We observe a strong positive correlation between guider size and performance gains. While the 1.5B guider yields reasonable improvements (+7.9% absolute on AIME 2024), scaling to the 7B guider nearly triples this gain to +21.2%. Similarly, on AMC 23, the improvement from base model jumps from +5.0% to +12.7% absolute.

These results demonstrate that THINKLOGIT is not merely transferring a generic “reasoning style” (e.g., token verbosity), but effectively transferring the specific reasoning capabilities of the guider. Consequently, as small reasoning models continue to improve in quality and efficiency, our inference-time method can directly leverage these advances to further boost large frozen models.

D Domain-Specific Analysis on Reasoning Behaviors

We further break down the reasoning behaviors defined in Section 6.2 by domain in Figure 7. The results show consistent trends across Math (AIME2024, AIME2025, AMC23, MATH Level 5), Science (GPQA Diamond), and Coding (LiveCodeBench) tasks. This detailed domain-specific breakdown confirms that the elicited reasoning behaviors are robust across different types of complex problem-solving scenarios and are not limited to a single domain.

E Case Study

To determine whether THINKLOGIT elicits meaningful thought processes or simply inflates output length, we compare the guider, target, and THINKLOGIT outputs on a selected question from the MATH-500 dataset (Figure 6). The analysis demonstrates that THINKLOGIT generates *informative reasoning behaviors*. In contrast to existing training-free methods (Muennighoff et al., 2025) that often artificially lengthen traces with repetitive

filler, the additional tokens here drive functional self-correction, successfully redirecting an erroneous trajectory toward the true solution.

F Qualitative Analysis of Failure Modes

While THINKLOGIT demonstrates strong capabilities in eliciting long chain-of-thought reasoning without training, we observe specific failure modes where the interaction between the guider’s impulse to lengthen generation and the target’s knowledge boundaries leads to reasoning collapse. We categorize these failures into three distinct types.

F.1 THINKLOGIT struggles to form higher-level abstractions

In this failure mode, the model successfully initiates a complex reasoning path and correctly sets up the problem constraints. However, when it lacks a specific high-level mathematical insight (e.g., recognizing an optimization boundary or an envelope), the guidance strength (α) prevents the model from stopping or guessing. Instead, the model enters a recursive loop, re-deriving the same intermediate conditions using varied phrasing until the context window is exhausted.

Question: Let $O(0, 0)$, $A(\frac{1}{2}, 0)$, and $B(0, \frac{\sqrt{3}}{2})$ be points in the coordinate plane. Let \mathcal{F} be the family of segments \overline{PQ} of unit length lying in the first quadrant with P on the x -axis and Q on the y -axis. There is a unique point C on \overline{AB} , distinct from A and B , that does not belong to any segment from \mathcal{F} other than \overline{AB} . Then $OC^2 = \frac{p}{q}$, where p and q are relatively prime positive integers. Find $p + q$.

Answer: 23.

Failure Analysis: The model correctly parameterizes the family of lines $y = (-q/p)x + q$ subject to $p^2 + q^2 = 1$. However, it fails to recognize that the “unique point” corresponds to the point of tangency with the family’s envelope (an astroid). Lacking this conceptual leap, the model attempts to prove the uniqueness condition algebraically for every possible parameter p, q .

THINKLOGIT Long CoT Excerpt

“Therefore, I need to characterize all points C on AB such that there exists no other PQ in \mathcal{F} with PQ intersecting AB at C ... Therefore, for C to NOT lie on any other PQ , this equation must have no solution p, q in $(0, 1)$ other than the $PQ = AB$ case.
[...]

Behavior	Keywords
Branching out	alternatively, another, try, suppose, consider, different, assume, also, option
Backtracking	however, but, mistake, error, contradiction, wrong, revisit, actually, again, flawed
Self-verification	check, verify, confirm, satisfy, plug, back, substitute, ensure, validate, test

Table 6: Keywords identifying three types of reasoning behaviors: Branching out, Backtracking, and Self-verification.

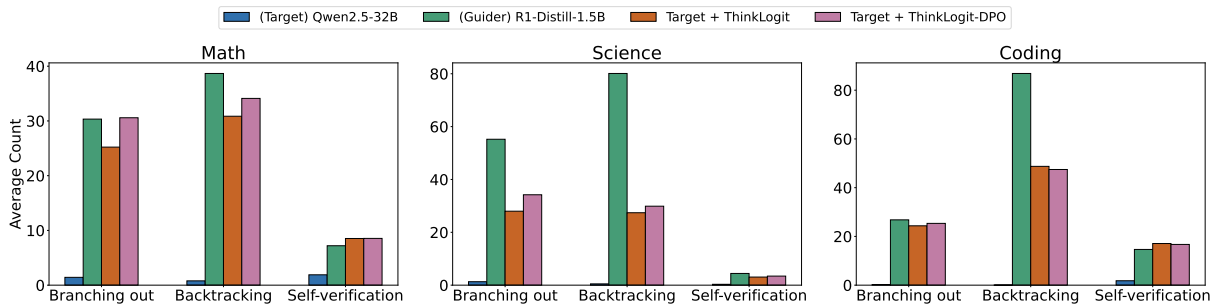


Figure 7: Domain-specific breakdown of reasoning behaviors across Math, Science, and Coding tasks. The elicited behaviors are robust across diverse domains.

So, $(-q/p)((1-t)/2) + q \neq (\sqrt{3}/2)t...$ leading to $\sqrt{1-p^2}[-(1-t)/p+2] \neq \sqrt{3}t$. Hmm. Alternatively, perhaps express q in terms of p or vice versa. Since $p^2 + q^2 = 1$, so $q = \sqrt{1-p^2}$.
[...]
But as I derived earlier, leading to $\sqrt{1-p^2}[-(1-t)/p+2] \neq \sqrt{3}t$. Hmm. Alternatively, perhaps parameterize differently. Wait, hold on. Let me think of C as a point on $AB...$ "

(The model repeats this derivation loop for over 1,000 tokens before terminating with a random guess.)

F.2 THINKLOGIT hallucinates contradictions that disrupt inductive reasoning

This failure mode occurs when the model introduces factual inconsistencies, or “hallucinations”, into its own reasoning history, directly contradicting valid conclusions it just derived. A single corrupted derivation in the reasoning traces obscures the underlying mathematical structure (e.g., an arithmetic sequence), preventing the model from inducing a general rule. Consequently, it abandons efficient generalization strategies and reverts to inefficient, step-by-step manual simulation that eventually exhausts the context window.

Question: *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.

Answer: 809.

Failure Analysis: The model correctly computes the status of small n values step-by-step. It correctly derives that $n = 21$ is a *winning* position. However, in the subsequent summary step, it hallucinates that 21 is *losing*. This error breaks the visible pattern (congruence modulo 5), causing the model to doubt its hypothesis and revert to manually simulating the game state for $n > 22$, which is computationally infeasible within the token limit.

THINKLOGIT Long CoT Excerpt

“ $n = 21$: current player can take 1, leaving 20 [which is losing]. So current player can leave opponent in losing—so $n = 21$ is **winning**.
Hmm, so the losing positions up to $n = 21$ are: 0, 2, 5, 7, 10, 12, 15, 17, 20, **21**...
Looking at differences between consecutive losing positions: $20-17=3$ $21-20=1$
So after the initial +2, it’s alternating between +3 and +2, except for the last jump which is +1 from 20 to 21.
Hmm, that’s a bit irregular.
[...]
 $n = 22$: Current player can take 1, leave 21—which is $dp[21]=True...$ Wait, no—if $n = 21$ is a losing position, then opponent would lose...”

F.3 THINKLOGIT fails to recover from a wrong assumption

Strong guidance can occasionally reinforce “greedy” heuristics. The model locks onto a simplified constraint early in the reasoning process. When rigorous algebraic substitution reveals a contradiction, the model exhibits *Algebraic Rigidity*: rather than questioning the heuristic, it assumes it has misread the problem statement, leading to a loop of text re-interpretation.

Question: Find the number of ways to place a digit in each cell of a 2×3 grid so that the sum of the two numbers formed by reading left to right is 999, and the sum of the three numbers formed by reading top to bottom is 99. The grid below is an example of such an arrangement because $8 + 991 = 999$ and $9 + 9 + 81 = 99$.

0	0	8
9	9	1

Answer: 45.

Failure Analysis: The model sets up the vertical sum $10(a + b + c) + (d + e + f) = 99$. It applies a greedy heuristic that place values do not overlap, assuming $a + b + c = 9$ (tens) and $d + e + f = 9$ (units). When substituting values derived from the rows yields a contradiction ($18 = 9$), the model enters a “verification loop,” repeatedly re-calculating the example provided in the prompt rather than revising the incorrect constraint $a + b + c = 9$.

THINKLOGIT Long CoT Excerpt

“So, $10(a + b + c)$ is 90, and $(d + e + f)$ is 9... So, top-to-bottom constraints: 1. $a + b + c = 9$ 2. $d + e + f = 9$
[...]
Substitute into constraint 5: $(9 - a) + (9 - b) + (9 - c) = 9$ Simplify: $27 - (a + b + c) = 9$. From constraint 4, $a + b + c = 9$, so: $27 - 9 = 18 = 9$? Hmm, that’s not matching.
This suggests a contradiction, meaning that my constraints are wrong. Wait, maybe I made a wrong assumption... Wait, let me retrace... Wait, hold on, the problem says...
(The model spends the remaining context re-verifying the example grid: “Wait, original grid: First row 0, 0, 8...”)”

F.4 Mitigations and Future Work

The failure modes identified above, specifically the tendency to loop or stagnate, suggest that a static guidance strength α may be suboptimal for all reasoning stages. Future work could investigate *dynamic guidance annealing* (Fan et al., 2024), where

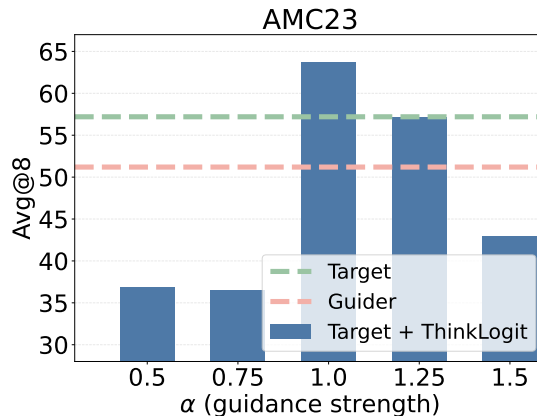


Figure 8: Sweeping the guidance strength α shows that $\alpha = 1.0$ yields the best trade-off between guider influence and target model priors.

α is reduced when repetitive n-grams or circular logic are detected, allowing the target model to “break out” of local minima or terminate unpromising paths. Additionally, since hallucination often stems from the model losing track of long contexts, explicitly prompting the guider to summarize intermediate states could reduce hallucination. Finally, for failures in abstraction, equipping the model with tool use (e.g., a Python interpreter) could allow it to offload intractable algebraic verifications that currently lead to context exhaustion.

G Impact of the Guidance Strength

In THINKLOGIT, the hyperparameter of the guidance strength α plays a critical role. We evaluate its effects on the AMC23 benchmark, which presents a suitable mix of problem difficulties and clearly exhibits both stability and guidance effects. We sweep α over $\{0.5, 0.75, 1.0, 1.25, 1.5\}$ to control how strongly the guider’s delta-logits modify the target’s distribution (see Figure 8). At $\alpha=1.0$, we observe the highest avg@8 together with moderate generation length, indicating an optimal trade-off between the guider’s corrective signal and the target model’s own priors. Crucially, all experiments in the main paper except this hyperparameter study use the same hyperparameter $\alpha=1.0$ as a robust default, demonstrating that *our method achieves strong performance without extensive hyperparameter tuning*.