AID: ASYMMETRIC INTERMEDIATE-ANSWER DISTILLATION FOR BUDGET-CONSTRAINED LONG-COT INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Supervised fine-tuning (SFT) on long chain-of-thought (long-CoT) traces is a common way to transfer reasoning ability from a large teacher to a smaller student, but this transfer can be brittle when the inference-time reasoning budget is limited. The distilled students often fail to finish reasoning within the available budget, and forcing an answer from an unfinished state does not reliably recover the correct solution. We introduce AID (Asymmetric Intermediate-Answer Distillation), which retains supervision on the full teacher trace while additionally training the student to predict the final answer from randomly sampled intermediate states. AID uses an asymmetric attention mask to keep the auxiliary answer branch out of the reasoning stream's forward context, allowing full-CoT imitation and intermediate-state answerability to be learned jointly. AID requires neither additional data generation nor extra parameters, and leaves inference unchanged. Experiments with Qwen3-0.6B-Base trained on OpenR1-Math and KodCode show that plain SFT degrades sharply on splits that combine long traces with harder problems, whereas AID substantially mitigates this degradation. Under long-trace distillation, accuracy improves from 22.9% to 37.0% on MATH-500 and from 25.6% to 35.6% on MBPP+. In a truncation-based evaluation on held-out validation problems, which reveals progressively longer prefixes of a reference reasoning trace, the normalized area under the accuracy curve rises from 0.191 to 0.419 on math and from 0.321 to 0.450 on code. Our results indicate that intermediate-state answerability provides a complementary training signal that makes long-CoT distillation more robust under constrained inference budgets.

1 INTRODUCTION

Long chain-of-thought (Long-CoT) distillation is an effective way to transfer the reasoning ability of a large teacher model to a small student model. Standard supervised fine-tuning (SFT) trains the student to imitate the complete reasoning trace generated by the teacher and to produce the final answer at its end. However, long and complex reasoning traces produced by strong teachers can exceed the limited learning capacity of small students, making them difficult to imitate reliably (Li et al., 2025; Yin et al., 2025; Jiang et al., 2025). In this work, we observe that this difficulty becomes particularly problematic when the inference-time reasoning budget is limited. Small students frequently fail to complete their reasoning within the given budget and tend to fall into repetitive loops that exhaust it. Robust Long-CoT distillation therefore requires not only the ability to imitate complete reasoning traces, but also the ability to derive the correct answer from intermediate, unfinished reasoning states.

To examine how students complete their reasoning under Long-CoT distillation, we use OpenR1-Math-220k (Lozhkov et al., 2025), a dataset of math problems paired with teacher reasoning traces. We partition the training data by naturally occurring trace length into a short band (512–2K tokens) and a long band (4K–8K tokens), distill a student on each, and evaluate both on the MATH-500 benchmark (Lightman et al., 2024) under the same 16K-token reasoning budget (Figure 1). Under Long-CoT, the budget exhaustion rate of the 0.6B student is 89.8%, far higher than 39.1% under Short-CoT; this gap is also observed at 1.7B but largely disappears at 4B (Figure 1(a)). Moreover, as shown in Figure 1(b), 86.7% of all generations from the 0.6B Long-CoT student exhaust the budget

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

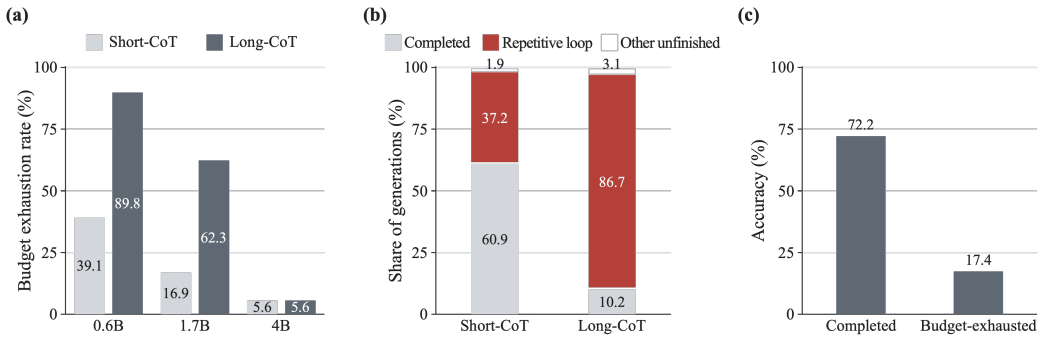

Figure 1: Reasoning-budget exhaustion of SFT students under Long-CoT. (a) Budget exhaustion rates of SFT students trained on Short-CoT and Long-CoT data, evaluated under the same 16K-token budget. (b) Breakdown of the 0.6B SFT student's generations into completed reasoning, repetitive loops, and other unfinished cases. (c) MATH-500 accuracy of the 0.6B Long-CoT SFT student, split by whether reasoning was completed within the budget.

in a repetitive loop, repeating identical or similar reasoning without making progress. Although the Long-CoT condition contains harder problems on average, these observations show that small students often fail to complete their reasoning within a limited budget even after being trained on long reasoning traces. Finally, as shown in Figure 1(c), the 0.6B SFT student achieves 72.2% accuracy when it completes its reasoning, but only 17.4% when it exhausts the budget. While this gap may partly reflect differences in problem difficulty, it also points to a limitation of standard Long-CoT SFT: it never explicitly trains the model to predict the answer from intermediate reasoning states. Improving Long-CoT distillation under limited reasoning budgets therefore requires learning not only to imitate complete reasoning traces, but also *intermediate-state answerability*, i.e., the ability to derive the correct answer from an unfinished reasoning state.

To this end, we propose Asymmetric Intermediate-Answer Distillation (AID). AID retains supervision on the teacher's complete reasoning trace while additionally training the student to predict the final answer from randomly selected intermediate reasoning states. An asymmetric attention mask allows the auxiliary answer branch to attend to the reasoning prefix, while preventing the auxiliary answer tokens from entering the forward context of the reasoning stream. As a result, AID jointly learns full-CoT imitation and intermediate-state answerability without additional model parameters or extra data generation. Evaluating Qwen3 students of several sizes (Yang et al., 2025) on math and code reasoning tasks, we find that AID improves final accuracy over standard SFT, especially in conditions that require long reasoning and for smaller students, and substantially improves intermediate-state answerability. Notably, in Long-CoT math experiments, AID barely changes how often the model voluntarily terminates its reasoning, yet improves answer recovery from generations whose reasoning remains unfinished. Taken together, our results show that jointly learning to imitate complete reasoning traces and to recover answers from intermediate states makes Long-CoT distillation of small language models more robust under limited reasoning budgets.

2 Asymmetric Intermediate-Answer Distillation (AID)

2.1 Standard Reasoning Distillation

Standard Long-CoT distillation uses as the training sequence $x^{\mathrm{SFT}} = [x^{\mathrm{pre}}, x^{\mathrm{rea}}, x^{\mathrm{ans}}]$, the concatenation of the input prefix x^{pre}, the teacher's reasoning trace x^{rea}, and the final answer x^{ans}. We denote their token lengths by L_p, L_r, and L_a, respectively, where L_r and L_a include the corresponding boundary tokens. The SFT objective over the reasoning and answer tokens is

$$\mathcal{L}_{\mathrm{SFT}} = -\frac{1}{|\mathcal{T}_{\mathrm{rea}} \cup \mathcal{T}_{\mathrm{ans}}|} \sum_{t \in \mathcal{T}_{\mathrm{rea}} \cup \mathcal{T}_{\mathrm{ans}}} \log p_\theta(x_t \mid x_{<t}), \tag{1}$$

where x_t is the t-th token of x^{SFT}, and $\mathcal{T}_{\mathrm{rea}}$ and $\mathcal{T}_{\mathrm{ans}}$ denote the sets of supervised positions in the reasoning trace and the final answer, respectively. Under standard SFT, the final answer is there-

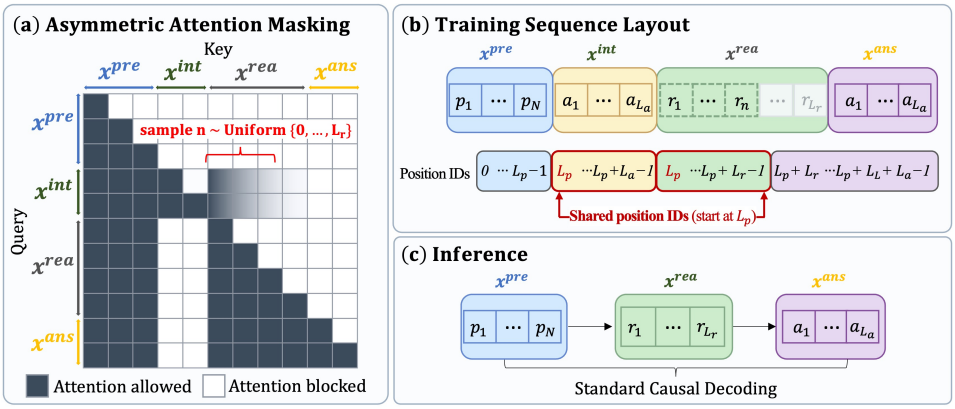

Figure 2: Overview of AID. (a) The asymmetric attention mask conditions the intermediate answer on a sampled reasoning prefix while blocking attention in the reverse direction. (b) A training-only answer segment is inserted, and the position IDs of the reasoning and final-answer segments are shifted by $-L_a$. (c) At inference, standard causal decoding is used without the auxiliary answer.

fore supervised only after the teacher's complete reasoning trace. AID retains this objective while additionally training the model to predict the final answer from intermediate reasoning states.

2.2 Intermediate-Answer Supervision

To train the model to predict the final answer from intermediate reasoning states, AID inserts a training-only auxiliary answer sequence x^{int} between the input prefix and the reasoning segment. The AID training sequence is

$$x^{\text{AID}} = [x^{\text{pre}}, x^{\text{int}}, x^{\text{rea}}, x^{\text{ans}}], \tag{2}$$

where x^{int} is a copy of the final answer x^{ans} with length L_a. For each training example, we sample the number of revealed reasoning tokens $n \sim \text{Uniform}\{0, 1, \ldots, L_r\}$ and supervise x^{int} to predict the final answer conditioned on the input prefix and the first n reasoning tokens. When $n = 0$, no reasoning tokens are visible; when $n = L_r$, the entire reasoning trace is visible. This auxiliary supervision provides an additional training signal while leaving the original supervision on the reasoning trace and the final answer unchanged.

2.3 Asymmetric Attention Masking

In the serialized sequence, x^{int} precedes x^{rea}. Under standard causal attention, subsequent reasoning tokens would therefore attend to the auxiliary answer, which could interfere with learning the original reasoning trace. To prevent this, AID applies an asymmetric attention mask. We denote the four token segments by I_{pre}, I_{int}, I_{rea}, and I_{ans}, and let $M_{ij}^{(n)} = 1$ indicate that query position i may attend to key position j, and $M_{ij}^{(n)} = 0$ otherwise. Starting from the standard causal mask, AID allows the auxiliary answer tokens to attend to the first n reasoning tokens, while blocking the reasoning and final-answer tokens from attending to the auxiliary answer:

$$M_{ij}^{(n)} = \begin{cases} 1, & i \in I_{\text{int}},\ j \in I_{\text{rea}}^{1:n}, \\ 0, & i \in I_{\text{rea}} \cup I_{\text{ans}},\ j \in I_{\text{int}}, \end{cases} \tag{3}$$

where $I_{\text{rea}}^{1:n}$ denotes the positions of the first n tokens of the reasoning segment; all other entries follow the standard causal mask. The auxiliary answer can thus predict the final answer from the selected intermediate reasoning state, while the original reasoning and final answer remain unaffected by the content of the auxiliary answer. This attention blocking does not stop backpropagation of the auxiliary answer loss: its gradients still flow into the shared model parameters.

2.4 Position Alignment

Inserting x^{int} shifts the positions of the reasoning and final-answer tokens in the serialized sequence by L_a. AID reassigns positions so that the reasoning and answer tokens have the same position IDs as in the standard SFT sequence:

$$\text{pos}'(i) = \begin{cases} i, & i \in I_{\text{pre}} \cup I_{\text{int}}, \\ i - L_a, & i \in I_{\text{rea}} \cup I_{\text{ans}}. \end{cases} \tag{4}$$

Consequently, x^{rea} and x^{ans} use the same position IDs as in the corresponding SFT sequence, and both x^{int} and x^{rea} start at position ID L_p, as illustrated in Figure 2(b). For the Rotary Position Embedding (RoPE) (Su et al., 2024)-based models used in this work, this requires only reassigning position IDs, without modifying RoPE itself.

2.5 Training Objective and Inference

The tokens SOT and SOA, which mark the start of the reasoning and answer segments, are provided as context and excluded from the loss, whereas EOT and EOS, which mark the end of these segments, are included as prediction targets. We denote the set of supervised positions in the auxiliary answer by $\mathcal{T}_{\text{int}}$ and define the full set of supervised positions as $\mathcal{T}_{\text{AID}} = \mathcal{T}_{\text{int}} \cup \mathcal{T}_{\text{rea}} \cup \mathcal{T}_{\text{ans}}$. For a supervised position t, let $\mathcal{A}_n(t)$ denote the set of input positions visible when predicting x_t under the asymmetric mask $M^{(n)}$. The AID training objective is

$$\mathcal{L}_{\text{AID}} = -\frac{1}{|\mathcal{T}_{\text{AID}}|} \sum_{t \in \mathcal{T}_{\text{AID}}} \log p_\theta \left(x_t \mid x^{\text{AID}}_{\mathcal{A}_n(t)},\ \text{pos}' \right). \tag{5}$$

All supervised tokens are weighted equally. AID only appends a copy of the final answer to the existing training sequence as the auxiliary answer, and requires neither additional teacher generation nor additional model parameters. At inference, the training-only segment x^{int} is omitted, and decoding uses the same position IDs and causal attention as a standard SFT model. The procedure for extracting the final answer under a fixed reasoning budget is described in Section 3.1, and an analysis of the training-time computational cost is provided in Appendix B.

3 Experiments and Analysis

We evaluate AID from two perspectives. First, we examine whether it improves downstream accuracy under a limited reasoning budget, and analyze how its effect varies with student size and teacher trace length. Second, we compare performance by whether reasoning is completed and directly evaluate answerability at intermediate reasoning states, to determine whether AID improves answer recovery from unfinished reasoning states independently of any increase in reasoning completion. To this end, we compare Short-CoT and Long-CoT distillation conditions across students of several sizes on math and code reasoning tasks.

3.1 Experimental Setup

Dataset. We construct math and code reasoning data from OpenR1-Math-220k and KodCode-V1-SFT-R1 (Xu et al., 2025), respectively. We partition teacher reasoning traces by their naturally occurring length into Short-CoT (512–2,048 tokens) and Long-CoT (4,096–8,192 tokens) subsets (Table 1). The two conditions consist mostly of different problems, and Long-CoT problems are harder. We measure difficulty on OpenR1-Math by the fraction of problems, among those with exactly two DeepSeek-R1 (R1) (Guo et al., 2025) generations, whose two generations disagree in final-answer correctness (*R1 answer disagreement rate*), and on KodCode by the fraction of problems carrying the dataset's *hard* label. Both indicators are higher under Long-CoT (math: 4.4%→7.9%; code: 8.2%→25.3%). We therefore interpret the two conditions as naturally occurring, distinct distillation regimes rather than a controlled experiment that isolates the causal effect of reasoning length alone. Additional statistics, including source and difficulty composition, are provided in Appendix A.

Models and Training. We use Qwen3-Base models with 0.6B, 1.7B, and 4B parameters as students and fine-tune all parameters for 5 epochs. All results are averaged over three training seeds,

Table 1: Data statistics of the Short-CoT and Long-CoT subsets. Token lengths are measured with the student model's tokenizer. As difficulty indicators, we report for OpenR1-Math the fraction of problems whose two R1 reasoning traces disagree in final-answer correctness, and for KodCode the fraction of problems labeled *hard*.

	Mathematics		**Code**	
Characteristic	**Short-CoT**	**Long-CoT**	**Short-CoT**	**Long-CoT**
Training / validation samples	4,000 / 1,000	4,000 / 1,000	4,900 / 100	4,900 / 100
Sequence length (tokens)				
Reasoning, range	512–2,048	4,096–8,192	512–2,048	4,096–8,192
Reasoning, mean	1,452	5,842	1,254	5,760
Question, mean	71	89	210	317
Answer, mean	4.7	6.5	88	193
Difficulty indicators				
R1 answer disagreement rate	4.4%	7.9%	–	–
Hard-problem proportion	–	–	8.2%	25.3%

Table 2: Downstream accuracy of the 0.6B students, averaged over three training seeds. Math benchmarks are MATH-500, Minerva, and OlympiadBench; code benchmarks are HumanEval+ and MBPP+. The best result in each column within each condition is shown in bold.

	Short-CoT					**Long-CoT**				
Method	**MATH**	**Minerva**	**Olymp.**	**HE+**	**MBPP+**	**MATH**	**Minerva**	**Olymp.**	**HE+**	**MBPP+**
SFT	0.351	**0.112**	0.104	**0.346**	**0.394**	0.229	0.054	0.089	0.240	0.256
UPFT-8	0.276	0.088	0.081	0.207	0.267	0.178	0.049	0.056	0.063	0.058
DFT	0.233	0.069	0.076	0.183	0.293	0.084	0.033	0.015	0.102	0.102
VCORE	0.357	0.087	0.111	0.268	0.350	0.239	0.047	0.081	0.203	0.213
AID	**0.370**	0.094	**0.124**	0.344	0.392	**0.370**	**0.107**	**0.139**	**0.276**	**0.356**

and the larger students are used to analyze how performance changes with student size. Training details are provided in Appendix B.

Baselines. We compare AID with SFT on full teacher reasoning traces; UPFT-8, the prefix-length $N = 8$ setting of Unsupervised Prefix Fine-Tuning (UPFT) (Ji et al., 2025); Dynamic Fine-Tuning (DFT) (Wu et al., 2026); and Variance-Controlled Optimization-based Reweighting (VCORE) (Gong et al., 2026). These baselines test whether reducing reasoning supervision or reweighting it is sufficient, without explicitly training the model to predict answers from intermediate states. Implementation details and model-specific adjustments are provided in Appendix C.

Evaluation. We evaluate math reasoning on MATH-500 (Hendrycks et al., 2021; Lightman et al., 2024), Minerva Math (Lewkowycz et al., 2022), and OlympiadBench (He et al., 2024), and code reasoning on HumanEval+ and MBPP+ (Chen et al., 2021; Austin et al., 2021; Liu et al., 2023). All Short-CoT and Long-CoT models receive reasoning budgets of 4,096 and 16,384 tokens, respectively. If reasoning does not terminate within the budget, generation is stopped and the model is prompted for the final answer, with an additional answer budget of 64 tokens for math and 2,048 tokens for code. Along with downstream accuracy, we analyze budget exhaustion and answer recovery from unfinished reasoning states. Details of the evaluation protocol and the computation of RS-AUC (Section 3.4) are provided in Appendix D.

3.2 AID Is Particularly Effective under Long-CoT Conditions

Table 2 summarizes the downstream accuracy of the 0.6B students. The gains from AID are much more pronounced under Long-CoT than under Short-CoT.

Long-CoT. AID achieves the highest seed-averaged accuracy on all five benchmarks. Compared with SFT, it improves MATH-500 accuracy from 22.9% to 37.0% and MBPP+ accuracy from 25.6%

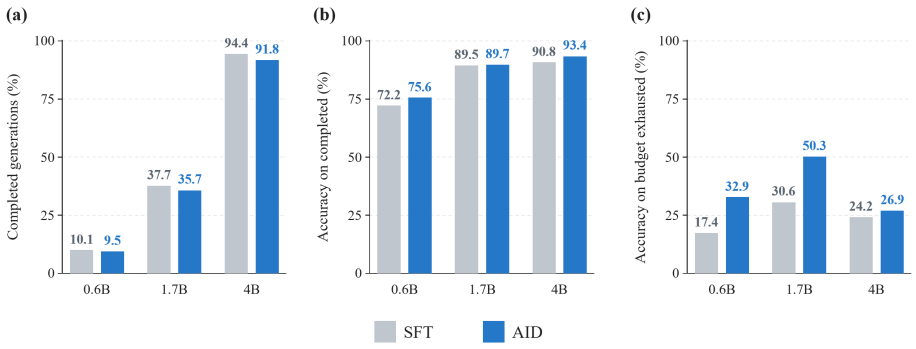

Figure 3: Performance on Long-CoT MATH-500 by reasoning completion, under a 16K-token reasoning budget. (a) Fraction of generations that complete reasoning within the budget. (b) Accuracy on completed generations. (c) Accuracy on budget-exhausted generations. The benefit of AID is most pronounced at 0.6B and 1.7B, where budget exhaustion is frequent, and becomes small at 4B, where most generations complete their reasoning.

to 35.6%, with consistent gains on the remaining benchmarks. In contrast, UPFT−8 and DFT degrade performance, and VCORE remains comparable to or below SFT. These results indicate that adjusting the scope of reasoning supervision or per-token loss weights alone is not enough to reproduce the gains of AID.

Short-CoT. The gains from AID are relatively small: it performs comparably to SFT on most benchmarks, improving on MATH-500 and OlympiadBench but falling slightly behind on Minerva. The benefit of AID is therefore not uniform across distillation conditions, but is most pronounced under Long-CoT, where reasoning is naturally longer and problems are harder. In the next section, we analyze this performance difference by whether reasoning is completed.

3.3 AID Yields Larger Gains in Budget-Exhausted Generations

To understand the gains under Long-CoT, we compare MATH-500 accuracy by whether reasoning is completed within the 16K-token budget (Figure 3). The completion rate increases with model size, but at each size AID and SFT have similar completion rates, with AID slightly lower (Figure 3(a)). In contrast, the accuracy gains differ sharply by completion status. On completed generations (Figure 3(b)), AID improves accuracy by 3.4, 0.2, and 2.6 percentage points at 0.6B, 1.7B, and 4B, respectively, whereas on budget-exhausted generations (Figure 3(c)) the gains reach 15.5 and 19.7 points at 0.6B and 1.7B; at 4B, where few generations exhaust the budget, the gain is 2.7 points, comparable to that on completed generations. This gain persists when the comparison is restricted to problem–seed pairs on which both methods exhaust the budget (16.5% $\rightarrow$ 31.1% at 0.6B; Appendix E). The benefit of AID thus comes mainly not from a higher completion rate, but from an improved ability to derive the correct answer from generations that fail to finish reasoning within the budget.

3.4 AID Improves Intermediate-State Answerability

The analysis in the previous section is based on each model's own generated reasoning, so differences in the traces themselves may contribute to the accuracy gap. To control for this, we give every method the first $k\%$ ($k \in \{0, 10, 25, 50, 75, 100\}$) of the same reference teacher trace, terminate reasoning at that point, and have the model generate the final answer, thereby evaluating intermediate-state answerability. Forcing answers from truncated reasoning has been used to analyze chain-of-thought faithfulness (Lanham et al., 2023) and to evaluate anytime reasoning across token budgets (Qi et al., 2025; Zhang et al., 2026); unlike these settings, which truncate each model's own reasoning, we condition all methods on an identical teacher prefix so that differences in generated traces are controlled for. Following the area-based summaries used in this line of work, we report the Reasoning-State Area Under the Curve (RS-AUC), the normalized area under the accu-

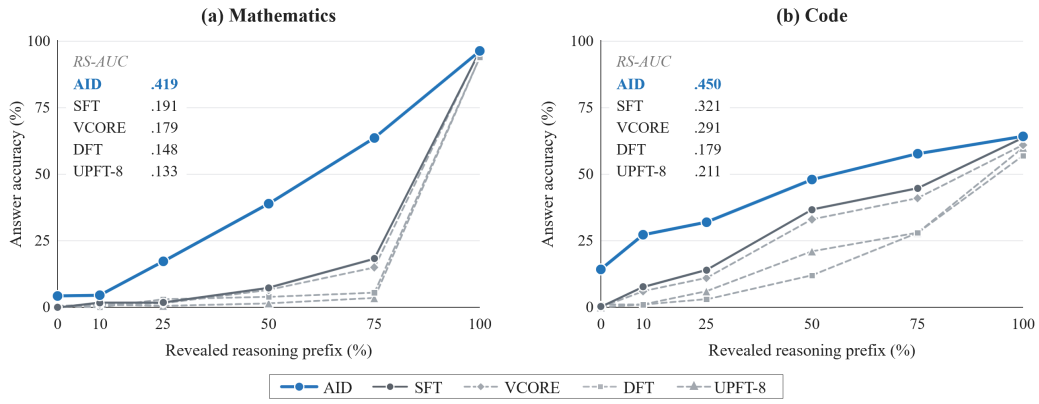

Figure 4: Intermediate-state answerability under controlled reasoning prefixes. All models receive the same portion of a reference teacher reasoning trace and then generate the final answer. Results are shown for Long-CoT (a) math and (b) code. RS-AUC is the normalized area under the accuracy curve over the revealed fraction of reasoning.

Table 3: **Ablation study on Long-CoT math.** Final denotes MATH-500 accuracy, and RS-AUC is the normalized area under the accuracy curve obtained by revealing progressively longer prefixes of the reference reasoning trace.

Condition	Modification	Final	RS-AUC
SFT	Full-trace supervision only	.229	.191
AID	Full method	**.370**	**.419**
w/o interior-state sampling	Sample only $n \in \{0, L_r\}$	.366	.383
w/o asymmetric attention	Allow reasoning to attend to auxiliary answers	.299	.339
w/o final-answer loss	Remove answer loss after full reasoning	.286	.324
w/o intermediate-answer loss	Remove auxiliary answer loss	.251	.178

racy curve computed over the actual k intervals. Evaluation uses only standard causal decoding; neither the training-only auxiliary answer nor the asymmetric attention mask of AID is used. We analyze the Long-CoT results here and report Short-CoT results in Appendix F.

In both domains, AID achieves the highest accuracy across intermediate reasoning states (Figure 4), increasing RS-AUC over SFT from .191 to .419 in math and from .321 to .450 in code. In contrast, UPFT−8, DFT, and VCORE all obtain lower RS-AUC than SFT in both domains. In math, the gap is especially large at later prefixes: at $k = 75\%$, AID reaches .637, more than three times the accuracy of SFT (.183). When the full trace is provided ($k = 100\%$), however, both methods converge to about .97, indicating that the difference arises at unfinished reasoning states. In code, AID reaches .143 even at $k = 0\%$, where no reasoning is provided (all other methods: .010 or below), while the gap at $k = 100\%$ is small. These results show that AID improves answerability even at identical intermediate reasoning states, where differences in each model's generated traces are controlled for.

3.5 Ablation Study

In the Long-CoT math setting, we analyze how the two answer losses, asymmetric attention, and interior-state sampling affect final accuracy and intermediate-state answerability. *w/o interior-state sampling* replaces uniform sampling over the entire reasoning trace with endpoint-only sampling, so that the intermediate answer is supervised only at $n \in \{0, L_r\}$ rather than at uniformly sampled intermediate states. Results are reported in Table 3.

Intermediate- and Final-Answer Losses. Removing the intermediate-answer loss reduces RS-AUC from .419 to .178, even below the .191 of SFT, and lowers final accuracy from .370 to .251. Removing the final-answer loss also reduces final accuracy and RS-AUC, to .286 and .324, respectively. These results show that intermediate-answer supervision provides a key training signal for

Problem

Given

$$f(x) = \begin{cases} x, & x \text{ is irrational}, \\ \dfrac{q+1}{p}, & x = \dfrac{q}{p},\ p, q \in \mathbb{Z}_+,\ (p,q) = 1,\ p > q, \end{cases}$$

find the maximum value of $f(x)$ on $\left(\frac{7}{8}, \frac{8}{9}\right)$. **Gold:** $16/17$

Final Answers at Different Reasoning Prefixes

Revealed	Teacher reasoning at the cut	SFT	AID
0%	(none)	✗ *"Okay, so I need to find…"*	✗ $1/8$
10%	Converts interval to decimals…	✗ *"Okay, let's start by…"*	✗ $1/2$
25%	$f = x + 1/p$; proposes $15/17$…	✗ *"f(15/17) = 16/17 … So 16/…"*	✓ **$16/17$**
50%	Confirms $15/17$; enumerating p…	✗ 16	✓ **$16/17$**
75%	Checked $p \leq 32$; $16/17$ largest…	✗ *"Suppose we have a…"*	✓ **$16/17$**
100%	Bounds $p \leq 19$; concludes $16/17$	✓ **$16/17$**	✓ **$16/17$**

Figure 5: Final answers after revealing the first $k\%$ of the same reference teacher trace (8,101 tokens) and requiring a final answer within 64 tokens. Gray quotes show the beginning of SFT outputs that exhausted the budget without committing to an answer: instead of answering, SFT continues reasoning or restarts from scratch. At k=25%, SFT reaches $16/17$ but fails to commit to it, whereas AID immediately produces the correct answer.

intermediate-state answerability, and that final-answer supervision also contributes to downstream performance. **Asymmetric Attention Isolation.** Removing the asymmetric mask allows reasoning tokens to attend to the auxiliary answer, reducing final accuracy and RS-AUC to .299 and .339, respectively. This shows that, beyond adding intermediate-answer supervision, it is important to isolate the auxiliary answer so that it does not enter the context of the original reasoning. **Interior-State Sampling.** Replacing uniform sampling with endpoint-only sampling keeps final accuracy close to that of full AID (.366 vs. .370), but reduces RS-AUC from .419 to .383. This suggests that supervising answers at diverse intermediate states, rather than only at the endpoints, contributes to intermediate-state answerability. Additional ablations on position alignment and under the Short-CoT condition are provided in Appendix G.

3.6 Qualitative Analysis of Intermediate-State Answerability

To examine answer recovery from intermediate reasoning states, we analyze individual cases under the same controlled-prefix setting as in Section 3.4. We give SFT and AID the first $k\%$ of the same reference teacher trace, terminate reasoning at that point, and have each model generate the final answer within a budget of at most 64 tokens. We select cases from problems satisfying the following conditions: neither model answers correctly at k=0; AID reaches the correct answer at an earlier revealed fraction than SFT and remains correct at all subsequent fractions; and both models answer correctly at k=100. The last condition equalizes final correctness when the full reasoning is provided, focusing the comparison on when each model becomes able to answer from intermediate states. Of the 200 validation problems, 36 satisfy these conditions, and Figure 5 shows one of them.

In this example, AID produces the correct answer $\frac{16}{17}$ from the 25% point of the reference reasoning onward and keeps it at every subsequent fraction. In contrast, SFT produces the correct answer only at 100%, when the full reasoning is provided. At 0%, 10%, 25%, and 75%, SFT fails to terminate within the 64-token answer budget. Notably, at 25%, the continuation generated by SFT itself arrives at $f(15/17) = \frac{16}{17}$, but exhausts the budget without committing to it as the final answer. At 50%, SFT does terminate, but outputs only the numerator 16 instead of the fraction. This example shows that even when both models answer correctly given the full reference reasoning, they can differ substantially in their ability to commit to the correct answer from a partial reasoning state: AID answers correctly with only 25% of the reference reasoning, whereas SFT exhausts the answer

budget or outputs an incomplete answer even when given more reasoning. This concretely illustrates the quantitative finding that AID improves answerability at intermediate reasoning states.

4 Related Work

Reasoning Distillation. Reasoning distillation transfers the teacher's rationales and thereby provides a richer training signal than answer-only supervision (Hsieh et al., 2023), and DeepSeek-R1 showed that supervised fine-tuning on long reasoning traces can transfer reasoning ability to small models (Guo et al., 2025). However, long traces can burden small models, for example through error accumulation (Luo et al., 2025), motivating methods that either change the training targets or reweight individual tokens. The former include selecting and compressing traces (Wang et al., 2025), supervising only a short prefix as in `UPFT` (Ji et al., 2025), and restructuring the training targets via prefix alignment as in `P-ALIGN` (Liu et al., 2026); the latter include `DFT` (Wu et al., 2026) and `VCORE` (Gong et al., 2026), which adjust loss weights according to predicted probabilities or per-token contributions. In contrast, AID modifies neither the teacher traces nor the loss weights, and instead adds intermediate-answer supervision to the same training examples.

Intermediate Prediction under Inference Budgets. Truncating the reasoning process to force an answer has been used to analyze CoT faithfulness (Lanham et al., 2023). For inference efficiency, budget forcing (Muennighoff et al., 2025) inserts an end-of-thinking token once the budget is exceeded, and `DEER` (Yang et al., 2026) exits early based on the confidence of tentative answers. These methods control when reasoning stops but do not train the ability to answer from an interrupted state. `AnytimeReasoner` (Qi et al., 2025) optimizes this ability with reinforcement learning, which requires verifiable rewards and student rollouts, and Budget-Aware Anytime Reasoning (Zhang et al., 2026) improves the quality of intermediate solutions across reasoning budgets using LLM-generated preference data and inference-time self-improvement. In contrast, AID learns intermediate-state answerability through supervised fine-tuning on teacher traces alone, and since it leaves the inference procedure unchanged, it is in principle compatible with inference-time termination methods.

5 Limitations and Scope

Our Short-CoT and Long-CoT conditions consist of naturally occurring teacher traces for different problems, and Long-CoT problems are also harder on average. The observed performance differences therefore cannot be interpreted as a causal effect of reasoning length itself. In addition, the benefit of AID is most pronounced under Long-CoT for small students, which frequently exhaust the reasoning budget; under Short-CoT and for larger students with high completion rates, the gains are small or inconsistent. Finally, this work considers fixed reasoning budgets, and combining AID with adaptive budget allocation or early-exit methods is left for future work.

6 Conclusion

To address the failure of small students to complete Long-CoT reasoning within a limited budget, we proposed AID, which retains supervision on the full teacher reasoning trace while training the student to predict the final answer from intermediate reasoning states. AID isolates the auxiliary answer from the original reasoning stream with an asymmetric attention mask, and requires neither additional model parameters nor changes to the inference procedure. Experiments show that AID improves downstream accuracy under Long-CoT, with gains driven mainly by better answer recovery in budget-exhausted generations rather than by higher completion rates. Intermediate-state answerability also improves under evaluation with identical teacher reasoning prefixes. These findings suggest that imitating complete reasoning traces and intermediate-state answerability are distinct and complementary objectives in reasoning distillation. A promising direction for future work is a custom attention implementation that exploits the block structure of the asymmetric mask to further reduce training-time overhead.

AI USE STATEMENT

The core research idea, proposed methodology, primary experimental design, and base implementation of this work were developed by the authors without the use of generative AI. We used generative AI tools to assist with implementing code for additional experiments and analyses, translating and editing the manuscript, drafting and revising parts of the paper, identifying and summarizing relevant literature, and assisting with the presentation and interpretation of experimental results. All AI-assisted code was reviewed and tested by the authors, relevant literature was verified against the original sources, and all AI-assisted text, claims, and interpretations were checked by the authors. We take responsibility for the final content of this work, including text, claims, code, and other artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work does not involve human subjects or the collection of personal or sensitive data. Our experiments use existing publicly available datasets and benchmarks, which are used for research purposes in accordance with their respective licenses and terms of use. We are not aware of additional ethical concerns specific to the proposed method beyond those generally associated with large language model research.

REPRODUCIBILITY STATEMENT

We provide detailed information necessary to reproduce our experiments in the appendix. Appendix A describes dataset construction and statistics; Appendix B reports training hyperparameters, implementation details, and computational setup; Appendix C describes the implementation of all comparison methods; and Appendix D specifies the evaluation protocol and RS-AUC computation. We additionally provide source code and instructions for reproducing the experiments in an anonymous repository at `https://anonymous.4open.science/r/AID-E21D`.

REFERENCES

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021. doi: 10.48550/arXiv.2108.07732.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. doi: 10.48550/arXiv.2107.03374.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Juechu Dong, Boyuan Feng, Driss Guessous, Yanbo Liang, and Horace He. Flex attention: A programming model for generating optimized attention kernels. In *Proceedings of Machine Learning and Systems*, volume 7, 2025.

Xuan Gong, Senmiao Wang, Hanbo Huang, Ruoyu Sun, and Shiyu Liang. VCORE: Variance-controlled optimization-based reweighting for chain-of-thought supervision. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 28161–28180, San Diego, California, United States, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.1298.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, 2025. doi: 10.1038/s41586-025-09422-z.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3828–3850, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.211.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 8003–8017, Toronto, Canada, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.507.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Ke Ji, Jiahao Xu, Tian Liang, Qiuzhi Liu, Zhiwei He, Xiaoyuan Liu, Xingyu Chen, Junying Chen, Benyou Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. The first few tokens are all you need: An efficient and effective unsupervised prefix fine-tuning method for reasoning models. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-0314.

Wangyi Jiang, Yaojie Lu, Hongyu Lin, Xianpei Han, and Le Sun. Teach small models to reason by curriculum distillation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 7412–7422, Suzhou, China, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.376.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626. Association for Computing Machinery, 2023. doi: 10.1145/3600006.3613165.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamilė Lukošiūtė, Karina Nguyen, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Robin Larson, Sam McCandlish, Sandipan Kundu, Saurav Kadavath, Shannon Yang, Thomas Henighan, Timothy Maxwell, Timothy Telleen-Lawton, Tristan Hume, Zac Hatfield-Dodds, Jared Kaplan, Jan Brauner, Samuel R. Bowman, and Ethan Perez. Measuring faithfulness in chain-of-thought reasoning. *arXiv preprint arXiv:2307.13702*, 2023. doi: 10.48550/arXiv.2307.13702.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. In *Advances in Neural Information Processing Systems*, volume 35, 2022. doi: 10.52202/068431-0278.

Yuetai Li, Xiang Yue, Zhangchen Xu, Fengqing Jiang, Luyao Niu, Bill Yuchen Lin, Bhaskar Ramasubramanian, and Radha Poovendran. Small models struggle to learn from strong reasoners. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 25366–25394, Vienna, Austria, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-acl.1301.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation. In *Advances in Neural Information Processing Systems*, volume 36, pp. 21558–21572, 2023. doi: 10.52202/075280-0943.

Zhenghao Liu, Zhuoyang Wu, Xinze Li, Yukun Yan, Shuo Wang, Zulong Chen, Yu Gu, Ge Yu, and Maosong Sun. Long-chain reasoning distillation via adaptive prefix alignment. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 18041–18059, San Diego, California, United States, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.822.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Anton Lozhkov, Hynek Kydlíček, Loubna Ben Allal, Guilherme Penedo, Edward Beeching, Quentin Gallouédec, Nathan Habib, Lewis Tunstall, and Leandro von Werra. OpenR1-Math-220k. Hugging Face dataset, 2025.

Renjie Luo, Jiaxi Li, Chen Huang, and Wei Lu. Through the valley: Path to effective long CoT training for small language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 4972–4992, Suzhou, China, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.251.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20275–20321, Suzhou, China, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.1025.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, et al. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Penghui Qi, Zichen Liu, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Optimizing anytime reasoning via budget relative policy optimization. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-0793.

Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. doi: 10.1016/j.neucom.2023.127063.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. TRL: Transformer reinforcement learning. `https://github.com/huggingface/trl`, 2020.

Zhaoyang Wang, Jinqi Jiang, Tian Qiu, Hui Liu, Xianfeng Tang, and Huaxiu Yao. Efficient long CoT reasoning in small language models. In *NeurIPS 2025 Workshop on Efficient Reasoning*, 2025.

Yongliang Wu, Yizhou Zhou, Ziheng Zhou, Yingzhe Peng, Xinyu Ye, Xinting Hu, Wenbo Zhu, Lu Qi, Ming-Hsuan Yang, and Xu Yang. On the generalization of SFT: A reinforcement learning perspective with reward rectification. In *International Conference on Learning Representations*, 2026.

Zhangchen Xu, Yang Liu, Yueqin Yin, Mingyuan Zhou, and Radha Poovendran. KodCode: A diverse, challenging, and verifiable synthetic dataset for coding. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 6980–7008, Vienna, Austria, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-acl.365.

An Yang et al. Qwen3 Technical Report, 2025.

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weiping Wang. Dynamic early exit in reasoning models. In *International Conference on Learning Representations*, 2026.

Huifeng Yin, Yu Zhao, Minghao Wu, Xuanfan Ni, Bo Zeng, Hao Wang, Tianqi Shi, Liangying Shao, Chenyang Lyu, Longyue Wang, Weihua Luo, and Kaifu Zhang. Marco-o1 v2: Towards widening the distillation bottleneck for reasoning models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23506–23516, Vienna, Austria, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.1145.

Xuanming Zhang, Shwan Ashrafi, Aziza Mirsaidova, Amir H. Rezaeian, Miguel Ballesteros, Lydia Chilton, Zhou Yu, and Dan Roth. Budget-aware anytime reasoning with LLM-synthesized preference data. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 8587–8599. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.417.

A Additional Dataset Details

Dataset construction. We construct the Short-CoT and Long-CoT subsets by partitioning naturally occurring teacher traces according to their reasoning length, rather than by shortening or extending reasoning traces for the same problems. For OpenR1-Math-220k, we start from 93,733 problems in the default configuration, remove multiple-choice questions and examples with malformed answers, and retain only problems with at least one teacher trace that is both correct and reasoning-complete. This filtering leaves 59,152 problems, which yield 111,856 traces after trace-level expansion, where each retained teacher generation of a problem is treated as a separate example. Among these, 21,875 traces from 13,817 problems fall in the Short-CoT range of 512–2,048 reasoning tokens, and 35,794 traces from 24,752 problems fall in the Long-CoT range of 4,096–8,192 tokens. For KodCode-V1-SFT-R1, each of the 268,211 rows corresponds to a single problem; 86,239 and 67,507 rows fall in the Short-CoT and Long-CoT ranges, respectively. From each band we sample 5,000 examples, which are split into 4,000 training and 1,000 validation examples for math, and 4,900 training and 100 validation examples for code. For mathematics, the training and validation sets are disjoint at the problem level, so no underlying problem appears in both splits even when multiple teacher traces are available for the same problem. We intentionally leave the 2,048–4,096-token interval unused so that the two regimes are clearly separated along the length axis.

The two bands are nearly disjoint at the problem level. In the math data, 17 problems (0.34% of one band) appear in both selected subsets; these are associated with distinct teacher traces whose lengths fall in their respective bands. The code subsets share no problem. Short-CoT and Long-CoT should therefore be interpreted as two naturally occurring problem regimes rather than paired interventions that manipulate trace length for the same problems.

Length statistics. All token statistics are measured with the Qwen3-0.6B-Base tokenizer over the training split, i.e., the distribution actually used for optimization. Table 4 complements the means reported in Table 1 with medians.

Table 4: Token-length statistics of the training splits (mean / median).

	Math		**Code**	
	Short-CoT	**Long-CoT**	**Short-CoT**	**Long-CoT**
Reasoning	1,452 / 1,476	5,842 / 5,746	1,254 / 1,241	5,760 / 5,544
Question	71 / 62	89 / 78	210 / 175	317 / 288
Answer	4.7 / 3	6.5 / 4	88 / 62	193 / 164

Difficulty characteristics. The two bands also differ in problem difficulty. For OpenR1-Math, we consider problems with exactly two available R1 generations and report the fraction whose two generations disagree in final-answer correctness. We restrict to this subset because the Long-CoT band contains more problems with four or more generations (11.0% vs. 4.7%), which mechanically inflates the probability that at least one generation is incorrect. The disagreement rate is 4.4% in Short-CoT and 7.9% in Long-CoT. For KodCode, we use the difficulty labels provided with the dataset; the proportion of examples labeled *hard* increases from 8.2% to 25.3%, while the proportion labeled *easy* decreases from 76.4% to 44.2%. Difficulty statistics are computed over the full 5,000-example subsets. These statistics confirm that reasoning length and problem difficulty covary in our naturally partitioned data; comparisons between Short-CoT and Long-CoT therefore characterize the two regimes and do not identify a causal effect of trace length alone.

Source composition. Tables 5 and 6 report the source composition of the selected 5,000-example subsets. These labels denote dataset provenance rather than a manually annotated subject taxonomy. The shift in provenance independently supports the difficulty difference: in math, the share of `olympiads` drops from 83.7% to 65.3% and is replaced mainly by `cn_contest` and `aops_forum`; in code, the synthetic `Prefill` and `Filter` subsets shrink substantially, whereas competitive-programming subsets such as `Leetcode`, `Data_Structure`, and `Codeforces` grow.

Table 5: Source composition (%) of the selected OpenR1-Math subsets.

Source	Short-CoT	Long-CoT
olympiads	83.7	65.3
cn_contest	8.4	16.7
aops_forum	6.6	13.8
amc_aime	0.6	1.7
inequalities	0.4	1.4
olympiads_ref	0.0	0.8
number_theory	0.3	0.3

Table 6: Source composition (%) of the selected KodCode subsets.

Source	Short	Long	Source	Short	Long
Prefill	21.2	5.9	Leetcode	5.6	11.5
Filter	18.0	4.4	Data_Structure	5.1	12.2
Taco	15.3	21.2	Evol	4.9	3.1
Apps	6.6	6.7	Docs	4.9	7.9
Algorithm	6.5	8.9	Codeforces	4.3	8.4
Code_Contests	5.7	8.0	Package	1.9	1.8

B TRAINING DETAILS

Optimization. All students are Qwen3-Base models (0.6B, 1.7B, 4B) trained with full-parameter fine-tuning. Table 7 lists the shared hyperparameters. All methods, including baselines, use the same data, number of epochs, effective batch size, and learning-rate schedule.

Table 7: Training hyperparameters shared across all methods.

Hyperparameter	Value
Optimizer	AdamW (Loshchilov & Hutter, 2019) (β_1=0.9, β_2=0.999, ϵ=10^{-8})
Weight decay	0.0
Learning rate	2×10^{-5}
LR schedule	Cosine decay to zero with 3% linear warmup
Effective batch size	32 sequences
Epochs	5
Optimizer steps	625 (math), 770 (code)
Precision	bf16 mixed precision
Maximum sequence length	3,072 / 9,216 (math Short/Long); 5,632 / 11,776 (code Short/Long)
Training seeds	42, 43, 44

The sequence-length limits include additional room for chat-template tokens. An audit of 50 training runs found no over-length examples or truncated answer targets.

AID implementation. For each training example, the number of revealed reasoning tokens $n \sim \mathrm{Uniform}\{0, 1, \ldots, L_r\}$ is resampled every time the example is drawn, so that different epochs supervise different intermediate states of the same trace. The asymmetric mask of Eq. 3 is constructed per example as a boolean 4D mask of shape $[B, 1, L, L]$ and passed to the model together with the reassigned position IDs of Eq. 4. No architectural component or parameter is added, and the auxiliary answer segment is discarded at inference.

Computational overhead. AID adds L_a tokens per sequence, which is negligible relative to L_r (Table 1). Its main cost comes from the custom attention mask. To isolate this cost, we benchmark a single training step (forward, backward, and optimizer step) of Qwen3-0.6B-Base with batch size 2 and gradient checkpointing enabled, changing only the attention backend and keeping the model, sequence, and trainer identical (Table 8). With the `eager` backend, which materializes the attention score matrix regardless of the mask, AID and `SFT` have nearly identical step time (1.00–1.01$\times$ for

math), showing that the asymmetric mask itself adds negligible computation; the slightly larger ratio for code (1.08×) reflects the longer auxiliary answer. The overhead arises because an arbitrary 4D mask prevents PyTorch (Paszke et al., 2019) SDPA from dispatching to its FlashAttention (Dao et al., 2022) backend. At the actual training lengths this results in roughly 2× step time, and the ratio grows with sequence length, while peak memory increases only marginally (less than 1% for math and about 3% for code, mainly due to the additional auxiliary-answer tokens). Since the AID mask has a simple block structure, a custom attention implementation that exploits this structure could further reduce the overhead, e.g., FlexAttention (Dong et al., 2025) with a `mask_mod`, or two FlashAttention calls in which the main sequence uses a standard causal mask and the auxiliary answer attends to the shared prefix. We leave such an implementation to future work.

Table 8: Controlled per-step cost of AID relative to SFT (Qwen3-0.6B-Base on a single H200 GPU, batch size 2, gradient checkpointing on). R is the reasoning length; rows in bold correspond to the average training lengths of the Long-CoT subsets. Code sequences are slightly longer under AID because of the longer auxiliary answer. Measurements use the smallest student, for which attention accounts for the largest share of compute.

		SDPA		Eager	
Domain	R	**Time**	**Memory**	**Time**	**Memory**
Math	2,048	1.46×	1.00×	1.01×	1.00×
Math	4,096	1.87×	1.00×	1.00×	1.00×
Math	**5,842**	**2.06**×	**1.00**×	**1.00**×	**1.00**×
Math	8,192	2.40×	1.00×	1.00×	1.00×
Code	**5,760**	**2.23**×	**1.03**×	**1.08**×	**1.03**×

Compute. All experiments were conducted on NVIDIA H200 GPUs with 141 GB of memory. The 0.6B and 1.7B students are trained on a single GPU, and the 4B Long-CoT students on two GPUs. With two GPUs, a single 4B training run takes about 5.5 hours on the Long-CoT math subset and about 7.5 hours on the Long-CoT code subset.

C Baseline Implementation

All baselines use the same data, optimizer, schedule, number of epochs, and evaluation protocol as SFT and AID. The original UPFT setting assumes an instruction-tuned or reasoning student, whereas our students are base models; where an adaptation was required, we chose the option that is favorable to the baseline.

UPFT. The original method trains on the first N tokens of self-generated responses. Because a base model cannot yet produce well-formed reasoning, we instead use the first N tokens of the teacher trace, keeping the structure ratio $p = 0.1$ of the original paper. The original work prescribes $N \in \{8, 32, 128\}$ depending on the model, so we searched over these values on the Long-CoT math subset (MATH-500 accuracy 0.178, 0.162, and 0.168, respectively) and use $N = 8$ throughout. UPFT is primarily a cost-saving method: in our setting it uses about one tenth of the supervised tokens and one fourth of the training time of SFT, which should be taken into account when reading its accuracy.

DFT. We use the implementation in TRL (von Werra et al., 2020) (`loss_type="dft"`), which scales the per-token cross-entropy by the detached token probability, $\ell_t = -\mathrm{sg}(p_\theta(x_t \mid x_{<t})) \log p_\theta(x_t \mid x_{<t})$, and normalizes by the number of supervised tokens as in the original paper.

VCORE. We reimplemented VCORE following the released code. At every optimizer step, per-token contributions are estimated by a finite-difference probe along the gradient of an independent probe batch, converted into weights by a within-sequence softmax with temperature $\tau = 5 \times 10^3$ ($\epsilon = 10^{-4}$), and the resulting update is rescaled by the variance-control coefficient. We made three choices that differ from the release. (i) We enable variance control, which is part of the method described in the paper but disabled in the released code. (ii) We apply the method to full-parameter

fine-tuning and perturb all parameters, rather than LoRA (Hu et al., 2022) parameters only; we verified that the relative perturbation $\epsilon\|g\|/\|\theta\|$ stays around 10^{-7}, well within the linear regime. (iii) The probe batch has the same number of sequences as the training batch. We did not tune τ or ϵ and use the smallest-model math configuration reported in the original paper. Because of the additional probe backward pass, `VCORE` takes about $2\times$ the training time of `SFT`.

D Evaluation Details

Benchmarks. We evaluate on MATH-500 (Hendrycks et al., 2021; Lightman et al., 2024) (500 problems), Minerva Math (Lewkowycz et al., 2022) (272), OlympiadBench (He et al., 2024) (672), HumanEval+ (Chen et al., 2021; Liu et al., 2023) (164), and MBPP+ (Austin et al., 2021; Liu et al., 2023) (378). For all three math benchmarks, answers are extracted from the final-answer segment and scored with the same three-stage equivalence checker: normalized string matching, numerical comparison with relative tolerance 10^{-6}, and symbolic equivalence using `math_verify`[1] when needed. Code is scored by executing the EvalPlus (Liu et al., 2023) test suites. All generations use greedy decoding with vLLM (Kwon et al., 2023).

Fixed-budget inference. Short-CoT and Long-CoT students receive reasoning budgets of 4,096 and 16,384 tokens, respectively. If the model does not emit the end-of-thinking token within the budget, generation is stopped, the end-of-thinking token is appended, and the model generates the final answer with an answer budget of 64 tokens for math and 2,048 tokens for code. The same budgets are applied to all methods within each condition. A generation is counted as *completed* if the model emits the end-of-thinking token within the reasoning budget, and as *budget-exhausted* otherwise.

Controlled-prefix evaluation. For the intermediate-state analysis (Section 3.4), we use problems from the validation split: 200 for Long-CoT math, 300 for Short-CoT math, and 100 for each code condition. For each problem and each $k \in \{0, 10, 25, 50, 75, 100\}$, we provide the question followed by the first $\lfloor k\% \cdot L_r \rfloor$ tokens of the reference teacher trace, append the end-of-thinking token, and let the model generate the answer with the same answer budget as above. All methods, including AID, use standard causal decoding; the auxiliary answer segment and asymmetric mask are not used. The same problems, cut points, and answer budgets are shared across methods.

RS-AUC. Let a_k be the accuracy at revealed fraction k, evaluated on the grid $(k_0, \ldots, k_5) = (0, 10, 25, 50, 75, 100)$. Because the grid is non-uniform, we compute the normalized area under the accuracy curve with the trapezoidal rule over $[0, 1]$:

$$\begin{aligned} \text{RS-AUC} &= \sum_{j=1}^{5} \frac{k_j - k_{j-1}}{100} \cdot \frac{a_{k_{j-1}} + a_{k_j}}{2} \\ &= 0.05\, a_0 + 0.125\, a_{10} + 0.2\, a_{25} + 0.25\, a_{50} + 0.25\, a_{75} + 0.125\, a_{100}. \end{aligned} \tag{6}$$

Contamination check. We compared 13-gram sets of each benchmark against the Long-CoT math training set. No benchmark problem is an exact duplicate of a training problem, and the fraction of problems sharing any 13-gram with the training set is 0.6% for MATH-500, 0.0% for Minerva Math, and 1.5% for OlympiadBench; we removed two OlympiadBench problems flagged by this check, leaving 672. HumanEval+ and MBPP+ contain no exact duplicates of the code training data, and at most 1.9% of their problems share a 13-gram with it. We initially considered AMC23 and AIME24 but excluded them because of their small size (30–40 problems) and because about a quarter of their problems share 13-grams with the training data.

E Matched Comparison on Budget-Exhausted Generations

Because the budget-exhausted sets may contain different problems for each method, we additionally compare AID and `SFT` on the 1,240 MATH-500 problem–seed pairs at 0.6B for which both methods

[1] `https://github.com/huggingface/Math-Verify`

exhaust the 16K-token reasoning budget. On this matched set, accuracy improves from 16.5% with SFT to 31.1% with AID, showing that the gain on budget-exhausted generations is not explained by differences in which problems each method fails to complete.

F Intermediate-State Answerability under Short-CoT

Table 9 reports the controlled-prefix results for the Short-CoT condition, complementing the Long-CoT results in Figure 4. In math, AID again shows a large advantage at intermediate prefixes (e.g., 0.518 vs. 0.136 at $k = 50\%$) while matching SFT at $k = 100\%$. In code, the advantage is smaller because SFT already answers reasonably well from short prefixes in this condition (0.353 at $k = 0\%$), leaving less room for improvement.

Table 9: Controlled-prefix accuracy under Short-CoT. Each column gives the accuracy after revealing $k\%$ of the reference teacher trace. Best per column in bold.

Method	0	10	25	50	75	100	RS-AUC
Math (n=300)							
SFT	.009	.012	.047	.136	.364	**.968**	.257
UPFT-8	.007	.007	.040	.080	.237	.967	.209
DFT	.010	.020	.047	.113	.243	.967	.222
VCORE	.007	.007	.043	.133	.333	.963	.247
AID	**.076**	**.090**	**.139**	**.518**	**.853**	.964	**.506**
Code (n=100)							
SFT	.353	.410	.480	.523	.577	.613	.517
UPFT-8	.160	.240	.350	.340	.490	.550	.384
DFT	.250	.000	.090	.210	.390	.610	.257
VCORE	.310	.300	.400	.490	.580	.610	.477
AID	**.437**	**.480**	**.510**	**.550**	**.633**	**.640**	**.560**

Answer termination. Table 10 reports, for math, the fraction of controlled-prefix generations in which the model finishes its answer within the answer budget. At low k, the baselines rarely produce a complete answer, whereas AID terminates at least 94% of cases regardless of k. At $k = 100\%$ all methods terminate in at least 99% of cases, indicating that the low termination at small k reflects a failure to switch from an unfinished reasoning state to answering rather than a failure to follow the answer format.

Table 10: Answer termination rate in the controlled-prefix evaluation (math).

	Short-CoT				Long-CoT			
Method	0	25	50	75	0	25	50	75
SFT	.000	.217	.317	.497	.000	.060	.195	.305
UPFT-8	.000	.023	.033	.120	.000	.000	.000	.015
DFT	.000	.000	.020	.130	.000	.005	.005	.015
VCORE	.000	.187	.320	.473	.000	.060	.130	.275
AID	**.967**	**.993**	**.990**	**.990**	**.955**	**.940**	**.960**	**.950**

G Additional Ablations

Table 11 reports the ablations of Section 3.5 under Short-CoT math, together with the position-alignment ablation. *w/o position alignment* keeps the serialized position IDs, so that reasoning and final-answer tokens are shifted by L_a relative to standard SFT.

Loss terms and sampling. The Short-CoT results follow the same pattern as Long-CoT. Removing the intermediate-answer loss leaves the endpoint accuracy unchanged (.964 vs. .964) but reduces RS-AUC from .506 to .298, confirming that the auxiliary objective specifically improves answerability at intermediate states. Removing the final-answer loss severely degrades final accuracy (.115),

Table 11: Ablations under Short-CoT math. Results except Trunc. are means over three training seeds. Final is MATH-500 accuracy with self-generated reasoning; Trunc. reports the seed-42 fraction of MATH-500 generations that exhaust the reasoning budget; Endpoint is controlled-prefix accuracy at $k = 100\%$.

Condition	Final	Minerva	Olymp.	Trunc.	RS-AUC	Endpoint
AID	**.370**	**.094**	.124	.456	**.506**	.964
w/o interior-state sampling	.366	**.094**	**.128**	.346	.458	.966
w/o asymmetric attention	.265	.071	.076	.376	.396	.940
w/o final-answer loss	.115	.025	.054	–	.366	.337
w/o intermediate-answer loss	.339	.083	.123	–	.298	.964
w/o position alignment	.019	.031	.006	.990	.497	**.971**

and its endpoint accuracy also drops to .337; its lower RS-AUC therefore partly reflects damaged answer extraction from completed traces, not only from intermediate ones. Replacing uniform sampling with endpoint sampling mainly lowers RS-AUC (.506 to .458) while final accuracy remains close.

Position alignment. Removing position alignment produces a striking dissociation. Final accuracy collapses to .019 because 97–99% of generations fail to terminate reasoning within the budget, yet the RS-AUC (.497) and endpoint accuracy (.971) remain at the level of full AID. This pattern was consistent across three seeds (final accuracy .014–.026, RS-AUC .494–.500, truncation rate .972–.990). Thus, without position alignment the model can still extract the answer from a given reasoning state, but fails to end its own reasoning during autoregressive generation. This also shows that final accuracy and intermediate-state answerability capture distinct capabilities. The effect is much weaker under Long-CoT, where the same ablation obtains a final accuracy of .337 and an RS-AUC of .309 (mean of three seeds), compared with .370 and .419 for full AID.