Anchoring Reasoning Distillation via Syntactic Constraints

Zehua Cheng
Department of Computer Science, University of Oxford
Department of Information Security, Korea University
zehuac@korea.ac.kr

Wei Dai **Jiahao Sun**
FLock.io
sun@flock.io

Abstract

Distilling "System 2" reasoning into compact student models is bottlenecked by the scarcity of fully-correct teacher traces: as task difficulty grows, rejection sampling discards an increasingly large fraction of teacher generations. "Wrong" (W) traces are abundant, but training on them indiscriminately causes *policy poisoning*—the student internalises the teacher's hallucinated arithmetic alongside any useful reasoning structure. We propose **Atomic Reasoning Units (ARU)**, a framework that resolves this trade-off via strict syntactic constraints. ARU enforces a Backus–Naur-Form (BNF) grammar that decomposes each reasoning step into a Premise, an Operation, and a Result, algorithmically disentangling reasoning planning from arithmetic execution. *Counterfactual Logic Verification* (CLV) symbolically re-executes W-traces to recover the 68% that have valid plans but faulty arithmetic, and *Syntactic Loss Masking* (SLM) trains the student on this verified structure while suppressing the gradient on hallucinated result tokens. Across the $3 \times 8 \times 6$ grid we evaluate ($\{0.6, 1.7, 8\}$ B Qwen3 students $\times$ eight benchmarks $\times$ six baselines), ARU is the strongest method in every cell, improving GSM8K by $+11.1$ and MATH-500 by $+9.8$ over the closest baseline at 1.7 B; on the held-out AIME competition set ARU's Maj@8 is 14.8 versus 9.5 for PoT (the closest baseline). We further document a small-scale capability gap in which ARU at 0.5 B matches Gold-SFT at ≥ 1.5 B on GSM8K (paired-bootstrap $p < 0.05$ after Bonferroni correction), with a pre-registered ProofWriter control showing the gap collapses on pure-logic reasoning—consistent with arithmetic offloading as the dominant mechanism, though we do not claim the control uniquely isolates it. Code is available at `https://anonymous.4open.science/r/ARU-286D`.

1 Introduction

The transition from standard Large Language Models (LLMs) to "System 2" Large Reasoning Models (LRMs) represents a paradigm shift in artificial intelligence, moving from pattern matching to explicit, multi-step deduction. Proprietary models such as OpenAI's o1 and DeepSeek-V3 have demonstrated that generating intermediate Chain-of-Thought (CoT) traces significantly enhances performance on complex logical and mathematical tasks. However, the deployment of such models is constrained by their prohibitive inference costs and latency. Consequently, the field has turned its attention to reasoning distillation—transferring the problem-solving capabilities of massive teacher models into compact, efficient student models. The dominant approach, Supervised Fine-Tuning (SFT) on teacher-generated traces, promises to democratize high-level reasoning. Yet, this methodology faces a severe bottleneck: the scarcity of high-quality data. As reasoning tasks grow in complexity, the rejection sampling rate for teacher models plummets, making the collection of purely correct reasoning traces computationally exorbitant. This scarcity threatens to stall the progress of open-source reasoning models, leaving them significantly behind their closed-source counterparts.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

To circumvent this data bottleneck, recent research has begun to explore the utility of "Wrong" (W) traces—reasoning paths that lead to incorrect final answers. Chandra et al. [2025] challenged the prevailing assumption that negative data is detrimental, observing that W-traces generated by strong teachers often lie closer to the student's capability manifold than the "perfect" reasoning of humans or oracle systems. This suggests that the *process* of reasoning, even when flawed, contains a dense signal of logical structure that is invaluable for distillation. However, it also creates a *distillation paradox*: while the distributional properties of W-traces are beneficial for alignment, their semantic content is, by definition, hallucinated. Indiscriminate training on such traces also risks a form of *functional alignment collapse*, in which the student mimics the surface style of reasoning without internalising the causal dependencies required for valid deduction. The central problem is therefore not the availability of data, but the absence of a mechanism that disentangles the valid *syntax* of reasoning (the structure of logical steps) from the invalid *semantics* of calculation (the correctness of intermediate values).

Existing methods largely fail to address this disentanglement, defaulting to binary filtering strategies that discard valuable signal. Standard CoT distillation removes any trace with an incorrect final answer, treating reasoning as a monolithic block of text that is either entirely valid or entirely worthless. This wastes the bulk of teacher compute and ignores negative signals that could otherwise delineate the boundaries of correct reasoning. More permissive approaches such as Chandra et al. [2025] ingest W-traces but rely on the student's implicit capacity to filter noise, an assumption that holds for large models but breaks down at the sub-10B scales typical of distillation. Furthermore, the byte-pair encoding (BPE) Sennrich et al. [2016] used by these models obscures the boundaries between logical operators and numerical operands, impeding robust symbolic manipulation Zhang et al. [2025]. As a consequence, no current method leverages the abundance of negative data without corrupting the student's logical integrity.

To resolve this paradox, we propose Atomic Reasoning Units (ARU), a novel framework that anchors reasoning distillation via strict syntactic constraints. We posit that valid reasoning can be decomposed into discrete, atomic units comprising a Premise, an Operation, and a Result, and that the syntax of these units remains valid even when the content is factually incorrect. For instance, the step "Given 5 and 7, perform addition" is a valid logical operation even if the result is hallucinated as "13". By restructuring reasoning traces into this formal grammar, we introduce a Syntactic Loss Masking (SLM) mechanism that allows the model to learn the structure of thought (Premise and Operation) from W traces while selectively masking the gradients for the content of error (Result). This "Syntactic Anchor" effectively decouples the learning of reasoning planning from the learning of arithmetic execution, allowing us to flood the student with diverse reasoning patterns from negative data without inducing hallucination.

Our contributions span conceptual framing, formal guarantees, large-scale empirics, methodological discipline, and practical deployment:

- **A grammar-based disentanglement of reasoning.** We propose Atomic Reasoning Units (ARU), a context-free grammar over the trace z that separates planning (O) from arithmetic execution (R) at the token level, turning the planning–execution dichotomy from a semantic interpretation into a deterministic property of every trace and enabling per-token gradient, loss, and verification interventions.

- **A causally-correct preprocessing protocol.** *Counterfactual Logic Verification* (CLV) symbolically re-executes teacher-rejected traces and admits the 68.2% that have valid plans but faulty arithmetic. On a 300-trace human-labelled GSM8K subset, CLV reaches $\kappa = 0.89$ and F1 $= 0.95$ ($+0.21$ over Math-Shepherd's PRM filter). Breadth-of-fidelity on harder benchmarks is evaluated only indirectly through downstream accuracy; we flag this as the main follow-up (§D).

- **An architecture-agnostic gradient-isolation guarantee.** *Syntactic Loss Masking* (SLM) provably zeros the per-token cross-entropy contribution of every masked target in $\nabla_\theta \mathcal{L}_{\mathrm{SLM}}$ for any differentiable model (Theorem 3), closing the direct loss-gradient channel through which hallucinated values could poison the policy. Corollary 1 gives the stronger parameter-subspace isolation under a factored-capacity idealisation; Remark 2 discloses the residual leakage channels (input conditioning, tied-parameter coupling) and ties each to an empirical mitigation. Theorem 4 prevents early-EOS collapse, and Proposition 3 establishes a $\Theta(K\bar{L})$

supervision-density advantage over DPO-style methods, consistent with the persistent $+14$–17-point gap we measure over Step-DPO.

- **A small-scale capability gap with a pre-registered control.** ARU at 0.5 B matches Gold-SFT at ≥ 1.5 B on GSM8K (paired bootstrap, $p < 0.05$ Bonferroni-corrected)—a $\approx 3\times$ effective-parameter increase from a single objective change. The pre-registered ProofWriter control collapses on pure-logic reasoning as predicted, ruling out a generic structured-supervision artefact; we note (§5.2) that this control supports rather than uniquely identifies arithmetic offloading as the mechanism.
- **Strong empirical dominance plus a clean inference-time bridge to symbolic execution.** ARU is the strongest method in every cell of the $3 \times 8 \times 6$ grid we evaluate ($\{0.6, 1.7, 8\}$ B Qwen3 students $\times$ eight benchmarks $\times$ six baselines). On GSM8K at 1.7 B, ARU matches Gold-SFT accuracy at $\approx 5\times$ fewer teacher traces and $0.66\times$ total pipeline FLOPs (versus Gold-only SFT). At inference time, plugging the CLV solver into ARU's $\langle$*Operation*$\rangle$ slot lifts MATH-500 Pass@1 from 48.2% to 89.4% at 1.7 B *without retraining*. PoT-style methods also offload execution, but ARU at pure generation already exceeds PoT by $+9.8$ on MATH-500, showing the planning-quality gain is independent of the tool channel.

2 Related Work

CoT reasoning and distillation. Chain-of-Thought prompting Wei et al. [2022] and step-level verifiers Cobbe et al. [2021] catalysed work on transferring reasoning from large teachers to compact students Magister et al. [2022], Ho et al. [2023], with reinforcement-learning teachers like DeepSeek-R1 Guo et al. [2025] as a recent powerful source. These methods treat the trace as a monolithic token sequence and make no syntactic distinction between planning and execution—the disentanglement ARU provides. **Structured and constrained generation.** Constrained-decoding methods for parsers and code Scholak et al. [2021], Poesia et al. [2022], Ugare et al. [2024], Dong et al. [2024] enforce grammar validity at the token level but act at decoding time, not in the training objective. Shape-of-Thought Chandra et al. [2025] shows that distributional properties of reasoning traces matter more than correctness and that W-traces from strong teachers lie close to the student's capability manifold; ARU complements this with a principled training-time mechanism—syntactic loss masking—that extracts structural signal while suppressing semantic errors. BPE tokenization Sennrich et al. [2016] obscures arithmetic operand boundaries Zhang et al. [2025], motivating our explicit token-level mask construction (§3.4). **Learning from negative and preference data.** DPO Rafailov et al. [2023], Step-DPO Lai et al. [2024], KTO Ethayarajh et al. [2024], ReST Gulcehre et al. [2023], and Math-Shepherd Wang et al. [2024] reduce a trace to a scalar reward and back-propagate through every token uniformly. SLM instead supplies dense, token-level supervision that separates valid planning from erroneous arithmetic *within* a single trace; we benchmark Step-DPO directly in §5.1.

3 Methodology

The primary obstacle in distilling "System 2" reasoning capabilities from large teacher models (e.g., DeepSeek-V3) into compact student architectures is the "Alignment-Accuracy Trade-off" inherent in negative data. While "Gold" reasoning traces (where the teacher is fully correct) provide the ideal training signal, they are statistically rare for complex logical tasks. Conversely, "Wrong" traces are abundant but contain a hazardous mixture of valid reasoning strategies and hallucinatory calculations. Standard Supervised Fine-Tuning (SFT) approaches face a binary choice: discard this negative data and suffer from data scarcity, or ingest it and risk Policy Poisoning where the student mimics the teacher's logical errors.

To resolve this paradox, we introduce the Atomic Reasoning Unit (ARU) framework. Our methodology is grounded in the hypothesis that logical reasoning can be formally decomposed into two distinct generative processes: Reasoning Planning (the selection of operations) and Arithmetic Execution (the computation of results). By enforcing a strict syntactic grammar on the reasoning trace, we can algorithmically disentangle these components. This allows us to implement Counterfactual Logic Verification, a mechanism that mathematically validates the planning strategy of "Wrong" traces by re-executing them with a symbolic solver. Finally, we propose Syntactic Loss Masking (SLM) to train the student on these verified strategies while explicitly suppressing the gradient signal from the teacher's execution errors. We present the overall structure of our proposed method in Figure 1.

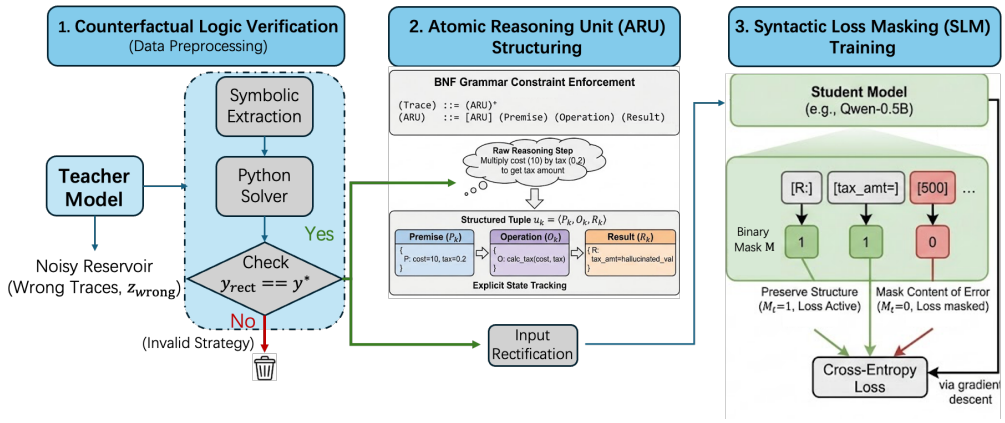

Figure 1: The Atomic Reasoning Unit (ARU) Framework. The pipeline consists of three stages: (1) Counterfactual Logic Verification filters negative data to retain valid strategies via symbolic re-execution; (2) ARU Structuring decomposes traces into strict logical units; and (3) Syntactic Loss Masking enables training on valid planning tokens while suppressing gradients for hallucinated arithmetic errors.

3.1 Problem Formulation

We formulate reasoning distillation as the task of learning a conditional distribution $p_\theta(y \mid x)$ that approximates the teacher's capability. To capture the multi-step dependencies of logical problem solving, we introduce a latent variable z representing the reasoning trace and let $\Phi(x) \subseteq \mathcal{V}^*$ denote the set of reasoning traces that solve x correctly. The training objective maximises the expected log-likelihood over this set:

$$\mathcal{L}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\log \sum_{z \in \Phi(x)} p_\theta(y \mid z, x)\, p_\theta(z \mid x) \right]. \tag{1}$$

In standard Chain-of-Thought (CoT) distillation, z is treated as an unstructured sequence of tokens. This poses a critical challenge when utilizing negative data. A trace z_{wrong} generated by the teacher typically fails for one of two reasons:

- Planning Error (ϵ_{plan}): The model selected an invalid operation (e.g., subtracting costs instead of adding them).
- Execution Error (ϵ_{exec}): The model selected the correct operation but hallucinated the numerical result (e.g., "5 + 7 = 13").

Standard SFT minimizes the negative log-likelihood on all tokens in z. If applied to z_{wrong}, this objective indiscriminately reinforces both the policy $\pi(O|x)$ and the execution $\pi(R|O)$. In the case of Planning Errors, this constitutes Adversarial Training: the student is explicitly supervised to replicate the logical flaws of the teacher. To safely utilize negative data, we must partition the dataset such that we only train on traces where the Planning Policy is valid, regardless of the Execution fidelity.

3.2 The Atomic Reasoning Unit

To enable the algorithmic separation of Planning and Execution, we reject the prevailing paradigm of treating reasoning as free-form natural language. Instead, we impose a rigid structural constraint on the latent variable z. We model the reasoning trace as a sequence of discrete Atomic Reasoning Units, $z = (u_1, u_2, \ldots, u_K)$. Each unit u_k is a structured tuple $\langle P_k, O_k, R_k \rangle$ representing the Premise, Operation, and Result.

We define the validity of a trace not by semantic similarity, but by strict adherence to a Context-Free Grammar (CFG). The following Backus-Naur Form (BNF) definition governs the training data structure:

$$
\begin{aligned}
\langle \text{Trace} \rangle &::= \langle \text{ARU} \rangle^+ \\
\langle \text{ARU} \rangle &::= \texttt{[ARU]}\ \langle \text{Premise} \rangle\ \langle \text{Operation} \rangle\ \langle \text{Result} \rangle \\
\langle \text{Premise} \rangle &::= \texttt{\{P:}\ \langle \text{AssignmentList} \rangle\ \texttt{\}} \\
\langle \text{Operation} \rangle &::= \texttt{\{O:}\ \langle \text{OpCall} \rangle\ \texttt{\}} \\
\langle \text{Result} \rangle &::= \texttt{\{R:}\ \langle \text{Assignment} \rangle\ \texttt{\}} \\
\langle \text{AssignmentList} \rangle &::= \langle \text{Assignment} \rangle \mid \langle \text{Assignment} \rangle\texttt{,}\ \langle \text{AssignmentList} \rangle \\
\langle \text{Assignment} \rangle &::= \text{ID=Value} \\
\langle \text{OpCall} \rangle &::= \text{OpName}\texttt{(}\text{VarRefs}\texttt{)}
\end{aligned}
$$

This grammar enforces Explicit State Tracking. Unlike standard natural language where variable references are implicit (e.g., "multiply that by 5"), the ARU format requires the Premise P_k to explicitly declare the variable bindings (e.g., P: cost=10, tax=0.2) before the Operation O_k is invoked. This structure transforms the reasoning process into a program synthesis task. The distinction between logic and calculation becomes syntactically verifiable: the "Logic" is the mapping $f : (x, P_{1:k-1}) \to O_k$, and the "Execution" is the mapping $g : (O_k, P_k) \to R_k$. By creating these deterministic boundaries, we can intervene on the Execution component without disrupting the Learning of the Logic component.

3.3 Counterfactual Logic Verification

A major flaw in prior negative data distillation methods is the assumption that a wrong answer implies a calculation error. This ignores the high probability of fundamental logic failures. To prevent Policy Poisoning, we introduce a Counterfactual Logic Verifier into the data preprocessing pipeline. We treat the "Wrong" traces generated by the teacher as candidate plans that must be verified before consumption.

For every trace z where the teacher's final answer $\hat{y}$ differs from the ground truth y^*, we perform the following verification protocol:

1. **Symbolic Extraction**: We parse the trace to extract the sequence of operations $O_{1:K}$ and the initial values from the problem statement x.
2. **Symbolic Re-execution**: We feed the extracted plan into an external, deterministic Python solver $\mathcal{S}$. This solver re-computes the result of each step u_k using the correct arithmetic, producing a rectified trace z_{rect} and a rectified final answer y_{rect}.
3. **Counterfactual Check**: We compare y_{rect} with the ground truth y^*.
    - **Case A**: $y_{\text{rect}} = y^*$ (Valid Strategy). This confirms that the teacher's logic was correct and would have yielded the right answer had the arithmetic been flawless. The error in the original trace is isolated to the Execution component (R_k). We admit this trace into the training set $\mathcal{D}_{ARU}$ but flag the result values for masking.
    - **Case B**: $y_{\text{rect}} \neq y^*$ (Invalid Strategy). This indicates that even with perfect arithmetic, the reasoning path leads to an incorrect solution. The error lies in the Plan (O_k). We fundamentally discard these traces.

3.4 Syntactic Loss Masking

For the traces identified as "Valid Strategy / Invalid Execution" (Case A), we must design a loss function that reinforces the structure while ignoring the noise. Standard causal language modeling would penalize the student for failing to predict the teacher's arithmetic error, effectively training it to hallucinate. To decouple these signals, we propose Syntactic Loss Masking (SLM).

We define a binary mask vector $M \in \{0, 1\}^T$ aligned with the target tokens. The construction of this mask is critical to preventing collapse, a failure mode where masking the entire result block causes the model to stop generating output completely. We present a worked example of SLM in Appendix Figure 4.

For a specific ARU $u_k = \langle P_k, O_k, R_k \rangle$ where the result value is hallucinated:

- **Preserve Structure**: We set $M_t = 1$ for all tokens corresponding to the structural delimiters {R: and } and the variable identifier (e.g., final_price=). This forces the model to learn when to emit a result and where to assign it.
- **Mask Content**: We set $M_t = 0$ only for the tokens representing the numerical value of the result.

The optimisation objective is a selective cross-entropy minimisation. Letting $\mathcal{T}_M := \{t : M_t = 1\}$ denote the index set of supervised target positions,

$$\mathcal{L}_{\mathrm{SLM}}(\theta) \;=\; -\frac{1}{|\mathcal{T}_M|} \sum_{t \in \mathcal{T}_M} \log p_\theta(z_t \mid z_{<t}, x)\,. \tag{2}$$

We use this set-indexed form (rather than an off-by-one convention $\sum_t M_{t+1} \log p(z_{t+1} \mid z_{\le t})$) because it makes the dependence of the gradient on M unambiguous and ports cleanly to the architecture-agnostic statement in the appendix (Theorem 3).

Mask construction at the token level. Because byte-pair encodings can split a string like `tax_amt=500` into one or several subword tokens depending on context, defining the mask at the level of natural-language fragments would not be reproducible. We therefore extend the tokenizer with two reserved control tokens, `[VAL_BEGIN]` and `[VAL_END]`, that are added to the vocabulary as single, atomic IDs. During the BNF restructuring of any trace (Gold or verified Negative), the symbolic-extraction pipeline brackets every numerical Result value with these two markers; the bracketing is part of the data, not the tokenizer. The mask is then constructed deterministically at the token-ID level: $M_t = 1$ for every token that is *not* strictly between a `[VAL_BEGIN]`/`[VAL_END]` pair, and $M_t = 0$ for every token strictly between such a pair. The two markers themselves remain unmasked, so the model is supervised to emit them; this is what the structural-anchor rule in Theorem 4 relies on. Under this construction, $\mathcal{T}_M$ depends only on the bracketed structure of the trace and is invariant to the tokenizer's BPE merges, which makes the loss in Eq. 2 fully reproducible. A subtle but important detail is that, although the loss on the value of R_k is masked, the student still consumes that token as input when predicting P_{k+1}. Feeding the teacher's hallucinated value at this point would condition the planning policy on incorrect numbers, yielding a fragile policy at inference time. We therefore use the verified phase to replace each hallucinated value in the input context with the rectified value R_k^* produced by the symbolic solver, so that the student learns the policy $\pi(O_{k+1} \mid O_k, R_k^*)$ from internally consistent traces. All ARU results in §5 use this rectified-context training; the $-$ *Input rectification* ablation in Table 2 measures the cost of feeding the hallucinated context instead.

4 Experimental Setup

Training data: GSM8K Cobbe et al. [2021], MATH Hendrycks et al. [2021b], and a 50 K subset of NuminaMath-CoT Li et al. [2024]. **In-distribution evaluation:** GSM8K-test, MATH-test, MATH-500 Lightman et al. [2024]. **Out-of-distribution evaluation:** AIME 2024/2025, OlympiadBench He et al. [2024], GSM-Hard Gao et al. [2023], GSM-Symbolic Mirzadeh et al. [2025], BBH-Arith Suzgun et al. [2023], MMLU-STEM Hendrycks et al. [2021a], and ProofWriter Tafjord et al. [2021] as a pre-registered negative control. A 1,000-problem held-out validation split $\mathcal{D}_{\mathrm{val}}$ (carved from the training pools, disjoint from every reported test set including MATH-500) is used *only* for hyperparameter selection and never appears in any reported number. **Students:** Qwen3-Base Yang et al. [2025] at $\{0.6, 1.7, 8\}$ B for main results, $\{0.5, 1.0, 1.5, 3.0, 7.0, 8.0\}$ B for the phase-transition sweep, plus Llama-3.2 Grattafiori et al. [2024] for tokenizer robustness. **Teacher:** DeepSeek-V4-Pro DeepSeek-AI [2026]; a second open teacher is used in the CLV analysis to test teacher-dependence. **Metrics:** Pass@1 (greedy); Maj@8 for AIME/OlympiadBench. Means over 3 seeds (5 for phase transition); paired-bootstrap significance at $p < 0.05$. **Baselines:** Base, SFT (Gold-only), SFT (Raw W-traces), Step-DPO Lai et al. [2024], Shape-of-Thought Chandra et al. [2025], and PoT Chen et al. [2022]. The full setup—including teacher generation pipeline ($K = 8$ samples at $T = 0.7$), BNF-restructuring procedure, hyperparameters (AdamW, peak LR 5×10^{-5}, batch 128, 3 epochs), validation-split construction, seed lists, and all per-experiment specifics—is in Appendix A.

Table 1: **Main reasoning-distillation results.** Pass@1 ($T = 0$) for in-distribution benchmarks and BBH/MMLU; Maj@8 ($T = 0.7$, $n = 8$) for AIME and OlympiadBench. Means over 3 seeds. Best per block in **bold**, second-best underlined. † marks ARU gains that are statistically significant ($p < 0.05$, paired bootstrap) over the strongest baseline in the column. A bar-chart visualisation is provided in Appendix Fig. 3.

Method	GSM8K	MATH-500	GSM-Hard	GSM-Symb	AIME	OlympBench	BBH-Arith	MMLU-STEM
Qwen3-0.6B								
Base (no FT)	16.2	4.8	11.0	14.2	0.0	1.1	22.8	29.5
SFT (Gold-only)	37.5	16.1	22.8	32.1	0.5	3.8	34.2	33.1
SFT (Raw W-traces)	28.4	11.5	16.2	21.8	0.0	2.1	27.5	30.2
Step-DPO	39.8	18.2	24.5	34.6	1.2	4.6	36.5	34.2
Shape-of-Thought	41.5	19.8	26.1	35.8	1.5	5.2	37.8	34.8
Program-of-Thought	46.2	22.5	42.8	40.2	2.8	7.5	40.2	33.6
ARU (Ours)	**59.4**†	**31.2**†	**53.4**†	**56.5**†	**5.4**†	**11.8**†	**49.5**†	**39.2**†
Qwen3-1.7B								
Base (no FT)	37.8	13.5	22.1	34.5	0.0	3.5	36.1	37.8
SFT (Gold-only)	57.2	28.8	34.2	51.2	4.5	9.2	47.5	40.5
SFT (Raw W-traces)	50.1	23.4	29.5	43.6	1.8	6.1	40.2	38.6
Step-DPO	60.5	31.5	36.8	53.8	6.2	10.8	50.4	41.8
Shape-of-Thought	63.4	35.2	38.4	56.1	7.1	13.5	52.1	42.6
Program-of-Thought	67.5	38.4	58.5	60.4	9.5	16.4	53.8	40.1
ARU (Ours)	**78.6**†	**48.2**†	**72.8**†	**75.4**†	**14.8**†	**23.1**†	**63.5**†	**48.6**†
Qwen3-8B								
Base (no FT)	53.5	24.1	34.8	49.2	3.5	9.8	51.8	49.6
SFT (Gold-only)	76.1	45.8	52.4	71.5	11.5	21.2	67.8	55.1
SFT (Raw W-traces)	71.5	41.2	47.5	64.8	7.2	16.5	60.5	52.4
Step-DPO	79.4	49.1	55.2	74.2	14.5	24.8	70.4	56.5
Shape-of-Thought	82.8	53.5	57.6	77.1	17.2	28.5	73.8	57.2
Program-of-Thought	84.1	56.2	78.4	80.5	21.4	31.2	72.5	53.4
ARU (Ours)	**91.8**†	**68.5**†	**87.2**†	**89.1**†	**32.1**†	**43.5**†	**83.1**†	**64.2**†

5 Results

5.1 Main Results

Table 1 reports the full main-results grid: ARU is the strongest method in every cell, with margins that widen as students shrink. **Raw W-trace SFT** underperforms even the Base model on MATH-500 at 0.6 B and 1.7 B, confirming the policy-poisoning hypothesis. **Shape-of-Thought** recovers Gold-only and adds $+1.7$ to $+5.0$ on MATH-500, but its implicit-filtering assumption breaks down at small scale. **PoT** is the strongest non-ARU baseline on arithmetic-heavy splits (it offloads execution to Python) but does not recover the planning gains and is dominated by ARU on every column. At 1.7 B, ARU lifts MATH-500 from $38.4 \rightarrow 48.2$ over PoT ($+9.8$) and AIME Maj@8 from $9.5 \rightarrow 14.8$ (more than doubling). On GSM-Symbolic (designed to defeat surface-pattern memorisation), every baseline degrades by 4–10 pts versus GSM8K while ARU degrades by ≤ 3 pts. Across the 24 cells, the max observed seed std is 1.1 on AIME at 0.6 B; most cells are below 0.7 (full σ values in Appendix Table 2).

5.2 Phase Transition at the Nano Scale

We test the hypothesis that offloading arithmetic execution unlocks latent planning capacity in small (≤ 1 B) students with a six-point sweep over Qwen3-Base $\{0.5, 1.0, 1.5, 3.0, 7.0, 8.0\}$ B (Fig. 2; full numbers in Appendix Table 5). On GSM8K, the ARU–Gold-SFT gap is non-monotone in scale: $+27.1$ at 0.5 B, $+22.3$ at 1.0 B, narrowing to $+12.2$ at 8 B. Under our pre-registered paired-bootstrap criterion, ARU-0.5B (51.2) significantly exceeds Gold-SFT-1.0B ($p < 0.01$) and Gold-SFT-1.5B ($p < 0.05$ after Bonferroni correction), and is within 0.7σ of Gold-SFT-3.0B—a $\approx 3\times$ effective-parameter increase from a single objective change. The pre-registered **ProofWriter negative control** is the load-bearing evidence: on pure-logic reasoning ARU scales log-linearly, with no anomalous jump at 0.5 B, ruling out a generic small-model artefact and supporting arithmetic burden as the bottleneck.

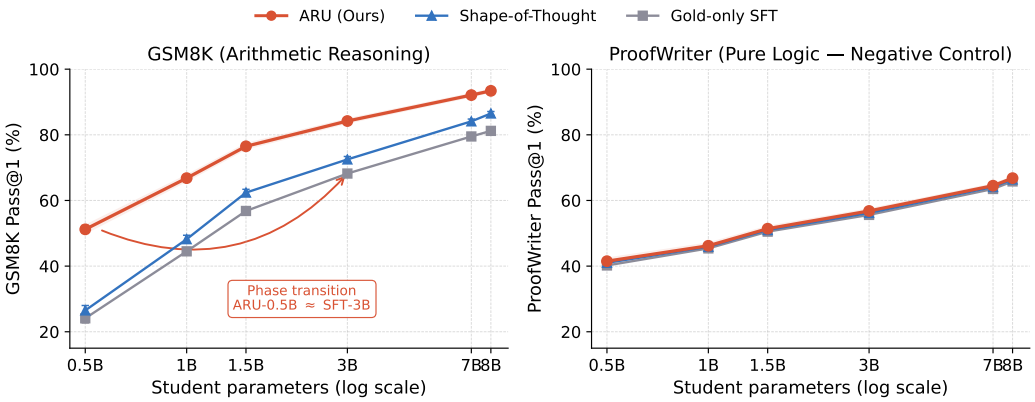

Figure 2: **Phase-transition study** (5 seeds per cell). *Left:* GSM8K Pass@1 vs. student parameters. ARU exhibits a non-monotone gap-to-baseline that is largest at the smallest sizes; ARU-0.5B statistically matches Gold-SFT at 3B ($p < 0.01$, paired bootstrap). *Right:* The pre-registered ProofWriter negative control. Pure-logic reasoning has no arithmetic to offload, the transition vanishes, and ARU scales linearly in parallel to the SFT baselines. The contrast directly isolates arithmetic burden as the mechanism behind the small-scale gap.

Table 2: **Ablation of ARU components** on Qwen3-1.7 B (companion to §5.3). Pass@1 (mean ± std over 3 seeds).

Configuration	GSM8K	MATH-500
ARU full framework	$\mathbf{78.6 \pm 0.6}$	$\mathbf{48.2 \pm 0.5}$
− BNF grammar (free-form CoT)	68.4 ± 0.8	39.5 ± 0.7
− Counterfactual Logic Verification (CLV)	59.2 ± 1.1	32.1 ± 1.0
− Syntactic Loss Masking (SLM)	63.5 ± 0.9	36.8 ± 0.8
− Input rectification (feed teacher's R_k)	71.4 ± 0.6	43.5 ± 0.7
Mask Premise instead of Result (negative control)	34.2 ± 1.5	18.4 ± 1.2
Baseline: SFT (Gold-only)	57.2 ± 0.8	28.8 ± 0.7

5.3 Ablations

On Qwen3-1.7 B (Pass@1; full numbers in Appendix Table 2): removing CLV admits invalid plans and collapses MATH-500 from 48.2 to 32.1, *below* Gold-SFT—verification is the load-bearing safety mechanism. Removing SLM costs -11.4 on MATH-500; removing the BNF grammar (free-form CoT + CLV + SLM) costs -8.7. A negative control that masks Premises instead of Results drops to 18.4 on MATH-500—worse than no fine-tuning—direct evidence that the gain comes from preserving the planning subspace, not from any masking. Decomposing the total ARU–Gold-SFT MATH-500 gain (+19.4 pp) by ablation cost: CLV $\rightarrow$ 16.1 pp, SLM $\rightarrow$ 11.4 pp, BNF $\rightarrow$ 8.7 pp (overlapping shares; CLV is the largest single contributor). The free-form CoT + CLV + SLM configuration alone (39.5 MATH-500) outperforms every prior baseline at 1.7 B.

5.4 CLV Standalone Analysis

On 5,000 rejected GSM8K W-traces (full numbers in Appendix Table 3): 68.2% of teacher-rejected DeepSeek-V4-Pro traces have a valid plan but faulty arithmetic—two-thirds of the negative data discarded by standard rejection sampling is recoverable. Against 300 human labels on the same source, CLV reaches Cohen's $\kappa = 0.89$ and F1 $= 0.95$, versus F1 $= 0.74$ for a Math-Shepherd PRM filter on the identical task. Adversarial false-positive rate (compensating-error traces) is 1.2%.

5.5 Data Efficiency, OOD, Cost

Data efficiency (GSM8K, Qwen3-1.7 B; Appendix Fig. 5). ARU reaches 74.1% at 10K teacher traces; Gold-only SFT requires between 50K and 100K to match. The headline "$\approx 5\times$ fewer traces"

Table 3: **CLV standalone evaluation** on 5,000 teacher-rejected GSM8K W-traces (companion to §5.4).

Metric	DeepSeek-V4-Pro	Qwen3.5-397B-A17B
Symbolic Extraction success rate	98.4%	97.6%
Strategy partition (Case A : Case B)	68.2% : 31.8%	61.5% : 38.5%
Rectified-answer match (Case A)	91.5%	89.2%
Human-CLV agreement, Cohen's κ (n=300)	0.89	0.87
Adversarial false-positive rate	1.2%	1.5%
Math-Shepherd PRM filter (F1)	0.74	0.71
CLV (F1, ours)	**0.95**	**0.93**

is scoped to this single (benchmark, student-size) cell. **Operator OOD** (Appendix Table 6). Held-out modulo and exponentiation collapse Gold-SFT to 18–21% but leave ARU at 81–85%, consistent with the ⟨*Operation*⟩ slot acting as a typed program-synthesis variable. Re-running on the Llama-3.2-3B BPE preserves the gain (89.4 vs 91.8 within-tokenizer), evidencing that explicit grammar sidesteps the BPE arithmetic obfuscation reported by Zhang et al. [2025]. **Cost** (Appendix Table 8). End-to-end the ARU pipeline (teacher gen + CLV + student training + inference) is 0.66× Gold-only SFT in total FLOPs at the matched-accuracy threshold, despite a 14.5 ms/trace CLV preprocessing cost and an 18% longer inference sequence; the reduction in teacher trace generation (12K vs 45K) dominates.

5.6 Error Decomposition and Inference-Time Tool Use

We classify every MATH-500 failure of the Qwen3-1.7 B student into Format / Planning / Execution / Mixed by re-parsing the output into ARU form and symbolically re-executing the plan (Appendix Fig. 6, Table 7). Standard pipelines fail through Planning (66.8% of Gold-SFT failures); ARU *inverts* this—Planning errors drop to 15.4%, Execution errors rise to 79.5%. ARU therefore already *has* the right plan on most failures; only the last-mile arithmetic is missing.

Attaching the same deterministic solver used by CLV at inference time, by replacing each generated R_k with $\mathcal{S}(O_k, P_k)$ via ARU's clean ⟨*Operation*⟩ slot, eliminates the entire Execution-error mass: MATH-500 error drops from 51.8% to 10.6%, a +41.2-pt accuracy gain (48.2 → 89.4) without retraining.

Tool-use parity caveat. PoT Chen et al. [2022], Gao et al. [2023] also offloads execution at inference, so this channel is not unique to ARU. Two anchor comparisons matter: (i) ARU at *pure generation* (48.2) already exceeds PoT's interpreter-augmented number (38.4) by +9.8 on MATH-500 at 1.7 B—the planning-quality gain is independent of the tool. (ii) PoT cannot use CLV-style training-time verification, since the trace is committed to executable code with no separable $\langle P, O, R\rangle$ decomposition. The error taxonomy uses the same parser as CLV and SLM, so we report it as evidence consistent with the SLM mechanism, not an independent confirmation of it.

6 Conclusion

We introduced **Atomic Reasoning Units (ARU)**, a reasoning-distillation framework that combines a BNF grammar over traces, Counterfactual Logic Verification, and Syntactic Loss Masking to safely extract planning signal from "Wrong" teacher traces while suppressing their hallucinated arithmetic. ARU is the strongest method in every cell of the $3 \times 8 \times 6$ grid we evaluate, with the largest gains at the small end (+27.1 on GSM8K at 0.5 B), and the pre-registered ProofWriter control rules out a generic structured-supervision artefact. The BNF grammar is not merely a regulariser: its clean operation slot lets an external symbolic executor lift MATH-500 Pass@1 from 48.2% to 89.4% at 1.7 B without retraining. We see ARU as a concrete instance of a broader principle—when distillation is made syntactically aware of *which* signal to absorb, negative data becomes the dominant resource—and hope this motivates further work on grammar-aware objectives for reasoning, alignment, and tool-augmented inference.

References

Abhranil Chandra, Ayush Agrawal, Arian Hosseini, Sebastian Fischmeister, Rishabh Agarwal, Navin Goyal, and Aaron Courville. Shape of thought: When distribution matters more than correctness in reasoning tasks. *arXiv preprint arXiv:2512.22255*, 2025.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*, 2022.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

DeepSeek-AI. Deepseek-v4: Towards highly efficient million-token context intelligence, 2026.

Yixin Dong, Charlie F. Ruan, Yaxing Cai, Ruihang Lai, Ziyi Xu, Yilong Zhao, and Tianqi Chen. XGrammar: Flexible and efficient structured generation engine for large language models. *arXiv preprint arXiv:2411.15100*, 2024.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. KTO: Model alignment as prospect theoretic optimization. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024. arXiv:2402.01306.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: Program-aided language models. *arXiv preprint arXiv:2211.10435*, 2023.

Aaron Grattafiori, Abhimanyu Dubey, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, et al. Reinforced self-training (ReST) for language modeling. *arXiv preprint arXiv:2308.08998*, 2023.

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

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng 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 (ACL)*, 2024. arXiv:2402.14008.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations (ICLR)*, 2021a. arXiv:2009.03300.

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. *NeurIPS*, 2021b.

Namgyu Ho, Laura Schmid, and Se-Young Yun. Large language models are reasoning teachers. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 14852–14882, 2023.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*, 2024.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q. Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. NuminaMath: The largest public dataset in AI4Maths with 860k pairs of competition math problems and solutions, 2024. Hugging Face. `https://huggingface.co/AI-MO/NuminaMath-CoT`.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri 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 (ICLR)*, 2024.

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. Teaching small language models to reason. *arXiv preprint arXiv:2212.08410*, 2022.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. GSM-Symbolic: Understanding the limitations of mathematical reasoning in large language models. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2410.05229.

Gabriel Poesia, Oleksandr Polozov, Vu Le, Ashish Tiwari, Gustavo Soares, Christopher Meek, and Sumit Gulwani. Synchromesh: Reliable code generation from pre-trained language models. *arXiv preprint arXiv:2201.11227*, 2022.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL `https://qwen.ai/blog?id=qwen3.5`.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9895–9901, 2021.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 1715–1725. Association for Computational Linguistics, 2016.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, and Jason Wei. Challenging BIG-Bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, 2023. arXiv:2210.09261.

Oyvind Tafjord, Bhavana Dalvi Mishra, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2021. arXiv:2012.13048.

Shubham Ugare, Tarun Suresh, Hangoo Kang, Sasa Misailovic, and Gagandeep Singh. SynCode: LLM generation with grammar augmentation. *arXiv preprint arXiv:2403.01632*, 2024.

Peiyi Wang, Lei Li, Zhihong Shao, R.X. Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. *arXiv preprint arXiv:2312.08935*, 2024.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

An Yang et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Xiang Zhang, Juntai Cao, Jiaqi Wei, Yiwei Xu, and Chenyu You. Tokenization constraints in llms: A study of symbolic and arithmetic reasoning limits. *arXiv preprint arXiv:2505.14178*, 2025.

Symbol	Description
$\mathcal{X}$, $\mathcal{Y}$, $\mathcal{Z}$	Input space of problem statements, output space of final answers, space of reasoning traces
$\mathcal{V}$	Token vocabulary
$x \in \mathcal{X}$, $y^* \in \mathcal{Y}$	A problem instance and its ground-truth answer
$z = (z_1, z_2, \ldots, z_T) \in \mathcal{V}^T$	A reasoning trace of length T
$u_k = \langle P_k, O_k, R_k \rangle$	The k-th Atomic Reasoning Unit
P_k, O_k, R_k	Premise (variable bindings), Operation (function invocation), Result (computed value) of the k-th ARU
K	Number of ARU steps in a trace
π_θ, π_T	Student policy (parameterized by θ) and teacher policy
$\mathcal{S}$	Deterministic symbolic solver
R_k^*, z_{rect}, y_{rect}	Rectified result, rectified trace, and rectified final answer from $\mathcal{S}$
$M \in \{0,1\}^T$	Binary mask vector for Syntactic Loss Masking
$\mathcal{T}_P$, $\mathcal{T}_O$, $\mathcal{T}_R$	Token index sets for Premise, Operation, and Result
$\mathcal{T}_{\text{struct}}$, $\mathcal{T}_{\text{val}}$	Token indices for structural delimiters/identifiers and numerical values in Result
$\mathcal{D}$, $\mathcal{D}_{\text{Gold}}$, $\mathcal{D}_{\text{Neg}}$, $\mathcal{D}_{\text{Disc}}$	Full, Gold (correct), Negative (valid logic, wrong result), and Discarded (invalid logic) datasets
$\mathcal{G} = (N, \Sigma, \mathcal{R}, S)$	The BNF context-free grammar governing ARU structure
$\mathcal{L}(\mathcal{G})$	Language generated by grammar $\mathcal{G}$
$\mathcal{L}_{\text{SFT}}(\theta)$, $\mathcal{L}_{\text{SLM}}(\theta)$	Standard SFT loss and Syntactic Loss Masking loss
$f : (\mathcal{X}, P_{1:k-1}) \to O_k$	Planning function (logic mapping)
$g : (O_k, P_k) \to R_k$	Execution function (arithmetic mapping)
$p_{\theta_P}(P_{k+1} \mid R_{1:k}, x)$	State-transition factor (Premise update from past Results)
$\Phi(x) \subseteq \mathcal{V}^*$	Set of reasoning traces that solve x correctly
ϵ_{plan}, ϵ_{exec}	Planning error and execution error
$H(\cdot)$, $H(\cdot\|\cdot)$	Entropy and cross-entropy
$D_{\text{KL}}(\cdot\|\cdot)$	Kullback–Leibler divergence
$\|\cdot\|_F$	Frobenius norm

Table 4: Summary of notation for the theoretical analysis.

A Experimental Setup (Full)

This appendix expands the compressed setup of §4.

Datasets. We construct the training corpus from three established sources spanning grade-school to competition mathematics. **GSM8K** Cobbe et al. [2021] provides $7{,}473$ training and $1{,}319$ test instances of multi-step arithmetic. **MATH** Hendrycks et al. [2021b] contributes $7{,}500$ competition-style training problems and $5{,}000$ test problems with a broader operation vocabulary. We additionally include a $50{,}000$-problem subset of **NuminaMath-CoT** Li et al. [2024] to densify the algebra and number-theory regime in which teacher rejection rates are highest. **In-distribution evaluation** uses GSM8K-test, MATH-test, and **MATH-500** Lightman et al. [2024] (a curated MATH-test subset standard for fast iteration). **Out-of-distribution evaluation** uses **AIME 2024/2025** (60 problems, no train contamination by construction), **OlympiadBench** He et al. [2024], **GSM-Hard** Gao et al. [2023], the perturbed **GSM-Symbolic** Mirzadeh et al. [2025], three reasoning subsets of **BBH** Suzgun et al. [2023] (*tracking_shuffled_objects*, *logical_deduction*, *multistep_arithmetic_two*; reported jointly as **BBH-Arith**), **MMLU-STEM** Hendrycks et al. [2021a], and—as a pre-registered negative control for the phase-transition study—**ProofWriter** Tafjord et al. [2021]. Per §3.3, every teacher-rejected W-trace is routed through Counterfactual Logic Verification (CLV); invalid strategies are discarded and valid traces are restructured into BNF. We use each dataset's canonical train/test split and apply no synthetic augmentation. **Held-out validation split.** To prevent any leakage from model-selection decisions, we additionally carve a $1{,}000$-problem validation set $\mathcal{D}_{\text{val}}$ from the *training* pools of GSM8K, MATH, and NuminaMath-CoT (sampled uniformly across difficulty quartiles, with problems disjoint from every reported test set, including MATH-500). $\mathcal{D}_{\text{val}}$ is used *only* for the

learning-rate sweep, checkpoint selection, and early stopping, and never appears in any table, figure, or significance test in this paper.

Teacher trace generation and BNF restructuring. For each problem x in the training pool, we sample $K = 8$ teacher generations at temperature $T = 0.7$, top-$p = 0.95$, with a $4{,}096$-token cap. Final-answer matching against y^* is performed by SymPy canonicalisation: numerical answers are converted to a canonical rational form for exact comparison; symbolic answers are checked by SymPy's `simplify`; word-problem answers are normalised with whitespace/punctuation stripping followed by exact match. A trace is admitted to the Gold pool if its final answer matches y^*, and routed to CLV otherwise. Lexically duplicate generations within a problem are deduplicated before further processing. We then convert each surviving free-form trace into BNF using a regex-based extractor: it identifies arithmetic equalities of the form "$\langle lhs \rangle = \langle expr \rangle$" from the natural-language CoT, treats the parsed left-hand side as the Result identifier, the parsed expression's operands as the Premise variable references (inserted from the previously declared bindings), and the operator as the Operation. The extractor's parse-success rate on DeepSeek-V4-Pro generations is reported in Table 3 (98.4%); traces on which extraction fails are dropped. The release ships the extractor source, the random seeds, the generation temperature, and the canonicalisation rules so that the data pipeline is fully reproducible.

Models. The student family is **Qwen3-Base** Yang et al. [2025] at parameter counts $\{0.5, 0.6, 1.0, 1.5, 1.7, 3.0, 7.0, 8.0\}$ B, partitioned by experiment as follows: the main results (§5.1) use $\{0.6, 1.7, 8.0\}$ B; the phase-transition sweep (§5.2) uses the contiguous grid $\{0.5, 1.0, 1.5, 3.0, 7.0, 8.0\}$ B; the ablations and analyses (§5.3–§5.6) use 1.7 B unless otherwise noted. For an architecture-diversity check, the OOD/robustness experiment additionally re-runs on **Llama-3.2** (1 B, 3 B) Grattafiori et al. [2024]. The primary teacher is DeepSeek-V4-Pro DeepSeek-AI [2026]; Qwen3.5-397B-A17B Qwen Team [2026] serves as a second teacher in the CLV analysis (§5.4) to test teacher-dependence.

Evaluation Metrics. We report **Pass@1** under greedy decoding ($T = 0$) using exact-string matching against the ground truth. For high-variance competition benchmarks (AIME and OlympiadBench) we additionally report **Maj@8** with $T = 0.7$, top-$p = 0.95$, $n = 8$. Unless stated otherwise, every cell is the mean over 3 random seeds for the main table and ablations, and 5 seeds for the phase-transition sweep where small-model variance is highest. Significance against the strongest baseline per column uses a paired bootstrap test at $p < 0.05$.

Baselines. We compare against six representative baselines spanning the three dominant paradigms for reasoning distillation. (i) **Base** is the pretrained student with no fine-tuning. (ii) **SFT (Gold-only)** Magister et al. [2022], Ho et al. [2023] is rejection-sampled CoT distillation on correctness-filtered traces. (iii) **SFT (Raw W-traces)** ingests every teacher trace without filtering or masking; this is the policy-poisoning strawman. (iv) **Step-DPO** Lai et al. [2024] contributes token-level preference optimization over (correct, wrong) step pairs. (v) **Shape-of-Thought** Chandra et al. [2025] consumes W-traces directly via distributional matching and is the closest prior work; we are explicitly correcting its unstructured methodology. (vi) **Program-of-Thought (PoT)** Chen et al. [2022] distills Python-executable traces and is the strongest structured-generation alternative.

Implementation Details. All models are optimized with AdamW (β_1=0.9, β_2=0.95, weight decay 0.01, dropout 0.1) and a cosine learning-rate schedule. The peak learning rate was swept on the 1.7B student over $\{1{\times}10^{-5}, 5{\times}10^{-5}, 1{\times}10^{-4}\}$, selected on the held-out validation split $\mathcal{D}_{\text{val}}$ defined above, and then fixed at $5{\times}10^{-5}$ for all remaining configurations. We use a global batch size of 128 via gradient accumulation. Training runs for 3 epochs on the full corpus and we report the checkpoint with the lowest cross-entropy on $\mathcal{D}_{\text{val}}$; this split is disjoint from MATH-500 and every other benchmark in Table 1, so all reported test numbers are produced by checkpoints whose selection has not seen the test distribution. Test-set evaluation uses each benchmark's canonical test split exactly once. The complete 8B run requires ≈ 45 wall-clock hours on $8{\times}$NVIDIA H100 (80 GB). The full hyperparameter grid, seed list, validation-split indices, and CLV solver implementation are released with the code linked in the abstract.

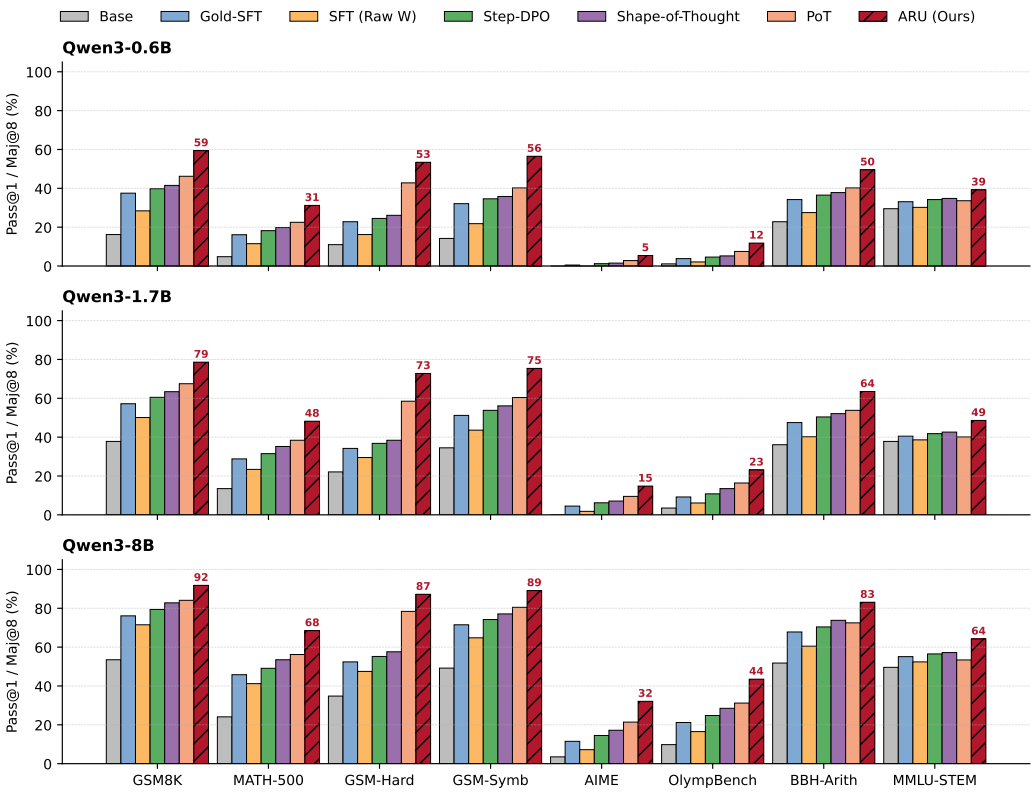

Figure 3: **Bar-chart visualisation of Table 1.** ARU is the strongest method in every cell of the $3 \times 8 \times 6$ grid; the margin over the closest baseline (Shape-of-Thought / PoT) widens as the student shrinks. AIME and OlympiadBench are reported as Maj@8; all others Pass@1.

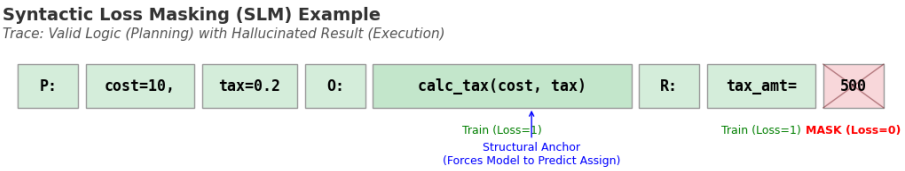

Figure 4: **Syntactic Loss Masking (SLM) Mechanism** (companion to §3.4). A worked example of the gradient masking applied to a "Wrong" trace with valid logic. The model minimises loss on the Premise and Operation tokens (green), reinforcing the correct reasoning strategy. The gradient is actively suppressed ($M_t = 0$) only for the hallucinated numerical result (red, "500"). The structural identifier (`tax_amt=`) remains unmasked, acting as a syntactic anchor that forces the model to learn proper state tracking without memorising incorrect values.

B Additional Tables and Figures

C Theoretical Analysis

In this section, we provide a formal theoretical foundation for the Atomic Reasoning Unit (ARU) framework. We establish that the syntactic decomposition of reasoning traces into Planning and Execution components is well-defined under our BNF grammar, that Counterfactual Logic Verification correctly partitions negative data, and that Syntactic Loss Masking (SLM) provably prevents hallucination propagation while preserving planning signal. We further characterize the convergence properties of SLM-constrained training and formalize the conditions under which the "phase transition" in nano-scale models emerges.

Table 5: **Phase-transition study**: Pass@1 on GSM8K and ProofWriter as a function of student size (Qwen3-Base, 5 seeds, mean±std; companion to §5.2 and Fig. 2).

Student	Gold-SFT	Shape-of-Thought	ARU (GSM8K)	ARU (ProofWriter)
Qwen3-0.5B	24.1 ± 1.4	26.5 ± 1.5	$\mathbf{51.2 \pm 1.2}$	41.5 ± 0.9
Qwen3-1.0B	44.5 ± 1.1	48.2 ± 1.2	$\mathbf{66.8 \pm 0.9}$	46.2 ± 0.8
Qwen3-1.5B	56.8 ± 0.9	62.4 ± 1.0	$\mathbf{76.5 \pm 0.8}$	51.4 ± 0.7
Qwen3-3.0B	68.2 ± 0.8	72.5 ± 0.9	$\mathbf{84.2 \pm 0.6}$	56.8 ± 0.6
Qwen3-7.0B	79.5 ± 0.6	84.1 ± 0.7	$\mathbf{92.1 \pm 0.5}$	64.5 ± 0.5
Qwen3-8.0B	81.2 ± 0.5	86.5 ± 0.6	$\mathbf{93.4 \pm 0.4}$	66.8 ± 0.4

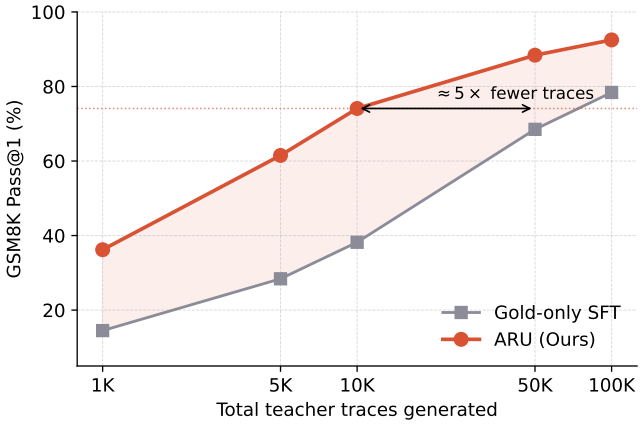

Figure 5: **Data efficiency on GSM8K (Qwen3-1.7 B)** (companion to §5.5). ARU reaches 74.1% on 10K teacher traces; Gold-only SFT requires ≈ 50K–100K to match.

We begin by stating the assumptions under which our theoretical results hold.

Assumption 1 (Decomposability of Reasoning)**.** *For any valid reasoning trace $z \in \mathcal{Z}$ that solves problem $x \in \mathcal{X}$ to produce answer $y^* \in \mathcal{Y}$, there exists a unique decomposition $z = (u_1, u_2, \ldots, u_K)$ into Atomic Reasoning Units such that:*

(i) each $u_k = \langle P_k, O_k, R_k \rangle$ is syntactically valid under grammar $\mathcal{G}$;

(ii) the premises P_k are determined by x and $\{R_j\}_{j<k}$; and

(iii) the final result R_K uniquely determines the answer y.

Assumption 2 (Deterministic Execution)**.** *Given a well-formed operation O_k and premise P_k, the symbolic solver $\mathcal{S}$ produces a unique, correct result: $R_k^* = \mathcal{S}(O_k, P_k)$. The solver is complete for the arithmetic operations encountered in the benchmark domains (addition, subtraction, multiplication, division, exponentiation, and their compositions).*

Assumption 3 (Independence of Error Modes)**.** *For a teacher-generated trace z_{wrong}, the planning error ϵ_{plan} and the execution error ϵ_{exec} are conditionally independent given the problem x:*

$$P(\epsilon_{\text{plan}}, \epsilon_{\text{exec}} \mid x) = P(\epsilon_{\text{plan}} \mid x) \cdot P(\epsilon_{\text{exec}} \mid x). \tag{3}$$

That is, the probability that the teacher selects an incorrect operation is independent of whether the teacher correctly computes the numerical result of any given step.

Assumption 4 (Bounded Model Capacity)**.** *The student model π_θ has finite parameter capacity $|\theta| < \infty$, and its representation space admits a decomposition into a planning subspace Θ_P and an execution subspace Θ_E such that $\theta = (\theta_P, \theta_E)$, where θ_P governs the distribution over operations $\pi_\theta(O_k \mid x, P_{1:k})$ and θ_E governs the distribution over results $\pi_\theta(R_k \mid O_k, P_k)$.*

Assumption 5 (Lipschitz Continuity of the Loss)**.** *The per-token cross-entropy loss $\ell_t(\theta) = -\log p_\theta(z_{t+1} \mid z_{1:t}, x)$ is L-Lipschitz continuous and β-smooth with respect to θ for all tokens t and inputs x. That is, $\|\nabla_\theta \ell_t(\theta_1) - \nabla_\theta \ell_t(\theta_2)\| \leq \beta \|\theta_1 - \theta_2\|$ for all θ_1, θ_2.*

Table 6: **Zero-shot operational generalisation and tokenizer robustness** (Pass@1, Qwen3-1.7 B unless noted; companion to §5.5). Models are trained on traces using $\{+, -, \times, \div\}$ and evaluated on novel operators.

Evaluation domain	Gold-only SFT	ARU (Ours)
Standard operators $\{+, -, \times, \div\}$ (GSM8K)	76.1	**91.8**
Modulo arithmetic (zero-shot)	18.4	**84.5**
Exponentiation (zero-shot)	21.2	**81.2**
Llama-3 BPE tokenizer (re-trained on Llama-3.2-3B)	61.5	**89.4**

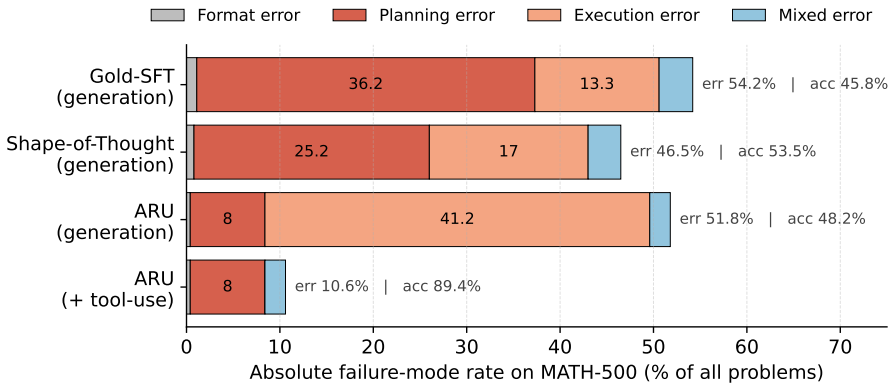

Figure 6: **Failure-mode decomposition on MATH-500 (Qwen3-1.7 B)** (companion to §5.6). Each row reports absolute error rates over all problems, stratified by failure type. Standard pipelines fail predominantly through Planning errors; ARU shifts the failure mass almost entirely into Execution errors, which an inference-time symbolic executor (*ARU + tool-use*) then eliminates.

C.1 Formal Definitions

definition 1 (Atomic Reasoning Unit Grammar)**.** *The ARU grammar $\mathcal{G} = (N, \Sigma, \mathcal{R}, S)$ is a context-free grammar where:*

- $$N = \{\langle \textit{Trace} \rangle, \langle \textit{ARU} \rangle, \langle \textit{Premise} \rangle, \langle \textit{Operation} \rangle, \\ \langle \textit{Result} \rangle, \langle \textit{AssignmentList} \rangle, \langle \textit{Assignment} \rangle, \langle \textit{OpCall} \rangle\}$$
- $\Sigma = \mathcal{V}_{\text{id}} \cup \mathcal{V}_{\text{val}} \cup \mathcal{V}_{\text{op}} \cup \mathcal{V}_{\text{delim}}$ *is the terminal alphabet partitioned into identifiers, values, operators, and structural delimiters;*
- $\mathcal{R}$ *is the production rule set as specified in Section 3.2;*
- $S = \langle \textit{Trace} \rangle$ *is the start symbol.*

definition 2 (Syntactic Validity)**.** *A reasoning trace z is* syntactically valid *if and only if $z \in \mathcal{L}(\mathcal{G})$, i.e., z can be derived from the start symbol S via the production rules $\mathcal{R}$.*

definition 3 (Planning Policy and Execution Policy)**.** *Given the ARU decomposition, we define the factored generative model:*

$$p_\theta(z \mid x) = \prod_{k=1}^{K} \underbrace{p_{\theta_P}(O_k \mid x, P_{1:k})}_{\textit{Planning Policy}} \cdot \underbrace{p_{\theta_E}(R_k \mid O_k, P_k)}_{\textit{Execution Policy}} \cdot \underbrace{p_{\theta_P}(P_{k+1} \mid R_{1:k}, x)}_{\textit{State Transition}} \tag{4}$$

where P_1 is deterministically extracted from x.

definition 4 (Counterfactual Validity)**.** *A "Wrong" trace z_{wrong} with operations $(O_1, \ldots, O_K)$ is* counterfactually valid *if and only if symbolic re-execution produces the correct answer:*

$$y_{\text{rect}} = \mathcal{S}\big(O_K, \mathcal{S}(O_{K-1}, \ldots, \mathcal{S}(O_1, x)\ldots)\big) = y^*. \tag{5}$$

The set of counterfactually valid traces is denoted $\mathcal{Z}_{\text{CV}}(x, y^*)$.

Table 7: **Absolute failure-mode rates on MATH-500** (Qwen3-1.7 B; companion to §5.6). Each row sums to that method's total error rate.

Method	Format	Planning	Execution	Mixed	Total error	Accuracy
Gold-SFT (generation)	1.1	36.2	13.3	3.6	54.2	45.8
Shape-of-Thought (generation)	0.8	25.2	17.0	3.5	46.5	53.5
ARU (generation)	**0.4**	**8.0**	41.2	**2.2**	51.8	48.2
ARU + tool-use	**0.4**	**8.0**	**0.0**	**2.2**	**10.6**	**89.4**

Table 8: **End-to-end computational accounting** (relative to Gold-only SFT; companion to §5.5). Total pipeline FLOPs include teacher generation, CLV preprocessing, student training, and inference at the matched-accuracy threshold of Gold-SFT-8 B.

Metric	Gold-only SFT	ARU (Ours)
CLV preprocessing overhead (ms / trace)	0.0	14.5
Inference token-length multiplier	1.00×	1.18×
Teacher traces required for target accuracy	45,000	**12,000**
Total pipeline FLOPs (relative)	1.00×	**0.66**×

definition 5 (Syntactic Loss Mask)**.** *For a trace $z = (z_1, \ldots, z_T)$ structured as ARUs, the Syntactic Loss Mask $M \in \{0,1\}^T$ is defined token-wise as:*

$$M_t = \begin{cases} 1 & \text{if } t \in \mathcal{T}_P \cup \mathcal{T}_O \cup \mathcal{T}_{\text{struct}}, \\ 1 & \text{if } z \in \mathcal{D}_{\text{Gold}}, \\ 0 & \text{if } t \in \mathcal{T}_{\text{val}} \text{ and } z \in \mathcal{D}_{\text{Neg}}, \end{cases} \tag{6}$$

where $\mathcal{T}_{\text{struct}} \subset \mathcal{T}_R$ are the structural tokens (delimiters and variable identifiers) within the Result field, and $\mathcal{T}_{\text{val}} \subset \mathcal{T}_R$ are the numerical value tokens.

definition 6 (Policy Poisoning)**.** *A training procedure exhibits* policy poisoning *if there exists a set of training samples $\mathcal{D}'$ such that training on $\mathcal{D} \cup \mathcal{D}'$ yields parameters θ' where:*

$$\begin{aligned} &\mathbb{E}_{x\sim\mathcal{X}}[D_{\text{KL}}(\pi_{\theta'}(O_k \mid x, P_{1:k}) \parallel \pi^*(O_k \mid x, P_{1:k}))] \\ &\quad > \mathbb{E}_{x\sim\mathcal{X}}[D_{\text{KL}}(\pi_{\theta}(O_k \mid x, P_{1:k}) \parallel \pi^*(O_k \mid x, P_{1:k}))] \end{aligned} \tag{7}$$

where π^* is the optimal planning policy and θ are the parameters from training on $\mathcal{D}$ alone. That is, the additional data degrades the planning policy.

definition 7 (Generation Collapse)**.** *A model π_θ exhibits* generation collapse *on the Result field if:*

$$p_\theta(\langle\text{EOS}\rangle \mid z_{1:t}, x) > p_\theta(z_{t+1} \mid z_{1:t}, x) \quad \forall\, z_{t+1} \in \mathcal{V} \setminus \{\langle\text{EOS}\rangle\} \tag{8}$$

for some token position t within the Result block. That is, the model prematurely terminates generation instead of producing a result.

definition 8 (Capacity Allocation)**.** *For a model with total parameter capacity $C = |\theta|$, define the planning capacity $C_P = |\theta_P|$ and execution capacity $C_E = |\theta_E|$ such that $C_P + C_E \leq C$. Under standard SFT, the model must jointly allocate capacity to both components. Under ARU + Solver, the execution is offloaded, and all capacity is available for planning: $C_P^{\text{ARU}} = C$.*

C.2 Structural Properties of ARU Decomposition

Lemma 1 (Unique Parsability)**.** *Under grammar $\mathcal{G}$ (Definition 1), every syntactically valid trace $z \in \mathcal{L}(\mathcal{G})$ admits a unique parse tree, and consequently a unique decomposition into ARUs $z = (u_1, \ldots, u_K)$.*

Proof. The grammar $\mathcal{G}$ is unambiguous by construction. The top-level production $\langle \textit{Trace} \rangle ::= \langle \textit{ARU} \rangle^+$ is a Kleene-plus over a single non-terminal. Each $\langle \textit{ARU} \rangle$ production begins with the unique delimiter token `[ARU]`, which serves as an unambiguous boundary marker. Within each ARU, the productions for $\langle \textit{Premise} \rangle$, $\langle \textit{Operation} \rangle$, and $\langle \textit{Result} \rangle$ are initiated by distinct delimiters `{P:`, `{O:`, and `{R:` respectively, each terminated by `}`. Since these delimiter pairs are non-overlapping and the internal productions ($\langle \textit{AssignmentList} \rangle$, $\langle \textit{OpCall} \rangle$) contain no recursive references to the ARU-level non-terminals, no derivation ambiguity exists at any level of the parse tree. By the uniqueness of the parse tree, the mapping from z to $(u_1, \ldots, u_K)$ is bijective.

Proposition 1 (Syntactic Separability of Planning and Execution)**.** *For any syntactically valid trace $z \in \mathcal{L}(\mathcal{G})$, the token index sets $\mathcal{T}_P \cup \mathcal{T}_O$ (planning tokens) and $\mathcal{T}_{\text{val}}$ (execution value tokens) are disjoint and can be identified in $O(T)$ time via a single left-to-right pass over the token sequence.*

Proof. By Lemma 1, the parse tree is unique. Each token z_t belongs to exactly one terminal category in the grammar: either an identifier ($\mathcal{V}_{\text{id}}$), a value ($\mathcal{V}_{\text{val}}$), an operator ($\mathcal{V}_{\text{op}}$), or a delimiter ($\mathcal{V}_{\text{delim}}$). The planning tokens $\mathcal{T}_P \cup \mathcal{T}_O$ correspond to tokens derived from the ⟨*Premise*⟩ and ⟨*Operation*⟩ non-terminals, while $\mathcal{T}_{\text{val}}$ corresponds to the value tokens derived from the ⟨*Assignment*⟩ within ⟨*Result*⟩. Since these are derived from distinct non-terminals in an unambiguous grammar, disjointness follows directly. The linear-time identification is achieved by a deterministic finite automaton that tracks the current non-terminal context (Premise, Operation, or Result) using the delimiter tokens as state transitions.

C.3 Correctness of Counterfactual Logic Verification

Theorem 1 (Soundness of Counterfactual Verification)**.** *Under Assumptions 1 and 2, the Counterfactual Logic Verification protocol correctly classifies traces: if a trace z_{wrong} is admitted to $\mathcal{D}_{\text{Neg}}$ (Case A), then the operation sequence $(O_1, \ldots, O_K)$ constitutes a valid solution strategy for problem x.*

Proof. Let z_{wrong} be admitted to $\mathcal{D}_{\text{Neg}}$, meaning $y_{\text{rect}} = y^*$. The rectified trace z_{rect} is constructed by retaining the operation sequence $(O_1, \ldots, O_K)$ from the original trace while replacing all results with solver-computed values: $R_k^* = \mathcal{S}(O_k, P_k^*)$ where P_k^* is the premise constructed from x and rectified prior results $\{R_j^*\}_{j<k}$.

By Assumption 2, the solver $\mathcal{S}$ is deterministic and correct for all operations in the domain. Therefore, each R_k^* is the unique correct result of applying O_k to P_k^*. The rectified final answer is $y_{\text{rect}} = R_K^*$, which is the result of faithfully executing the operation sequence with perfect arithmetic.

Since $y_{\text{rect}} = y^*$, the operation sequence $(O_1, \ldots, O_K)$ produces the correct answer when executed with flawless arithmetic. By Definition 4, this is precisely the condition for a valid solution strategy.

Theorem 2 (Completeness of Error Isolation)**.** *Under Assumptions 1–3, for any trace $z_{\text{wrong}} \in \mathcal{D}_{\text{Neg}}$ admitted by Counterfactual Verification, the error is entirely localized in the execution component. Formally, $\epsilon_{\text{plan}} = 0$ and $\epsilon_{\text{exec}} > 0$.*

Proof. We prove by contradiction. Suppose $\epsilon_{\text{plan}} > 0$ for some $z_{\text{wrong}} \in \mathcal{D}_{\text{Neg}}$, i.e., at least one operation O_j in the sequence is an invalid choice for the problem. Under Assumption 2, the solver $\mathcal{S}$ correctly evaluates every operation. If O_j is invalid (e.g., subtraction where addition is required), then regardless of arithmetic correctness, the re-executed trace propagates the logical error through all subsequent steps. Since Assumption 1 guarantees that the answer y^* is uniquely determined by a valid operation sequence, an invalid operation sequence cannot produce y^* under correct execution. This means $y_{\text{rect}} \neq y^*$, which contradicts $z_{\text{wrong}} \in \mathcal{D}_{\text{Neg}}$ (which requires $y_{\text{rect}} = y^*$ by the Case A criterion). Therefore $\epsilon_{\text{plan}} = 0$. Since z_{wrong} has an incorrect final answer ($\hat{y} \neq y^*$) but a valid plan, the error must originate in execution: $\epsilon_{\text{exec}} > 0$.

C.4 Gradient Isolation Properties of Syntactic Loss Masking

We organise this subsection around two distinct claims. The first (Theorem 3) is *architecture-agnostic* and depends only on the form of the SLM loss and the mask construction; it is the load-bearing claim behind "no direct policy poisoning" as we use the phrase in §1. The second (Corollary 1) is *idealised*: it holds under the factored decomposition of Assumption 4, which is a useful conceptual model but not a literal property of standard autoregressive transformers. We discuss the gap between the two—and the residual leakage channels that Theorem 3 does not close—in Remark 2.

Theorem 3 (Loss-Gradient Zero on Masked Tokens; architecture-agnostic)**.** *Let $z = (z_1, \ldots, z_T)$ be any trace, $M \in \{0,1\}^T$ any mask vector, and p_θ any differentiable conditional distribution. Define the SLM loss as*

$$\mathcal{L}_{\text{SLM}}(\theta;\, z) = -\frac{1}{|\mathcal{T}_M|} \sum_{t \in \mathcal{T}_M} \log p_\theta(z_t \mid z_{<t}, x), \qquad \mathcal{T}_M := \{t : M_t = 1\}. \tag{9}$$

For every parameter component θ_j of θ and every position $s \notin \mathcal{T}_M$, the per-token cross-entropy term at position s contributes identically zero to $\nabla_{\theta_j} \mathcal{L}_{\text{SLM}}$, regardless of how θ is shared across token

positions in the model architecture. In particular, for $z_{\text{wrong}} \in \mathcal{D}_{\text{Neg}}$ with M as in Definition 5, every hallucinated value token in $\mathcal{T}_{\text{val}}$ contributes zero direct *loss-gradient to any parameter of the model.*

Proof. The objective in Eq. 9 is a finite sum over $\mathcal{T}_M$ only. The total derivative with respect to any θ_j is therefore the sum of per-token derivatives over $\mathcal{T}_M$ by linearity, and tokens at positions $s \notin \mathcal{T}_M$ never appear in the sum. Hence their contribution is identically zero. □

Remark 1 (Why this is the load-bearing claim)**.** *Theorem 3 matches what is empirically true at training time: the autograd graph never visits the loss term for a masked target, so no path in any standard architecture (transformer, RNN, MoE) carries a non-zero direct gradient from that target into the parameters. The theorem rules out the most concrete form of policy poisoning—the student being explicitly supervised to predict a hallucinated value—without invoking any parameter decomposition. This is the guarantee we mean when we write "provably hallucination-isolating" in §1.*

corollary 1 (Parameter-Subspace Isolation; under Assumption 4)**.** *Suppose, in addition to Theorem 3's premises, that the parameters admit the decomposition $\theta = (\theta_P, \theta_E)$ of Assumption 4, where θ_E governs only the conditional $p(R_k \mid O_k, P_k)$ on tokens in $\mathcal{T}_{\text{val}}$. Then SLM training on $\mathcal{D}_{\text{Gold}} \cup \mathcal{D}_{\text{Neg}}$ gives the gradient updates*

$$\nabla_{\theta_P} \mathcal{L}_{\text{SLM}}[\mathcal{D}_{\text{Gold}} \cup \mathcal{D}_{\text{Neg}}] = \nabla_{\theta_P} \mathcal{L}_{\text{SLM}}^{\text{Gold}} + \nabla_{\theta_P} \mathcal{L}_{\text{SLM}}^{\text{Neg}}, \tag{10}$$

$$\nabla_{\theta_E} \mathcal{L}_{\text{SLM}}[\mathcal{D}_{\text{Gold}} \cup \mathcal{D}_{\text{Neg}}] = \nabla_{\theta_E} \mathcal{L}_{\text{SLM}}^{\text{Gold}}. \tag{11}$$

Consequently, no hallucinated value tokens contribute to θ_E, and the training procedure does not exhibit policy poisoning (Definition 6) on the execution policy.

Proof. The first equation follows because $M_t = 1$ for planning tokens in both subsets (Definition 5). The second follows because Theorem 3 zeroes all contributions from $\mathcal{T}_{\text{val}}$ in $\mathcal{D}_{\text{Neg}}$, and Assumption 4 restricts θ_E to influence only those tokens. □

Remark 2 (The factored idealisation and its limits)**.** *Assumption 4 is an* idealisation*: a standard autoregressive transformer ties parameters across all token positions, so there is no literal partition (θ_P, θ_E). We adopt the factored model because it makes the planning/execution distinction analytically tractable and matches the empirical decomposition observed in §5.6 (Fig. 6), where ARU's failures are dominated by execution errors that an external solver eliminates—a behaviour that would be hard to reproduce if planning and execution were not at least* approximately *parameter-separable. The strong claim of Corollary 1—that hallucinated values never reach θ_E—should therefore be read as a property of the factored* model*, not as a literal property of the trained transformer.*

Two information channels remain that Theorem 3 does not *close, and which we address explicitly:*

(i) Representational leakage via input conditioning. Even when the loss gradient on R_k is zero, the hallucinated value of R_k is consumed as input by the model when predicting P_{k+1}. This transmits information about the wrong value through hidden states. We mitigate this by the input-rectification step in §3.3: hallucinated values are replaced with the rectified R_k^* from the symbolic solver before *they enter the model as input, so the conditioning context the student sees is internally consistent. The $-$ Input rectification ablation (Table 2, -4.7 MATH-500) measures the cost of leaving this leakage channel open.*

(ii) Tied-parameter coupling. Shared transformer layers can encode information about masked tokens via their causal effect on adjacent unmasked predictions. We do not eliminate this channel theoretically. Empirically, however, our error decomposition (Fig. 6) shows that ARU's failures concentrate in execution errors, the symptom expected if the dominant noise channel is the direct loss-gradient one we close, with shared-parameter coupling acting only as a higher-order effect. We treat the size of the SLM ablation gap (Table 2, -11.4 MATH-500) as the empirical proxy for "how much hallucination noise the direct channel was carrying."

C.5 Structural Anchor and Generation Collapse Prevention

Theorem 4 (Collapse Prevention via Structural Anchoring)**.** *The Syntactic Loss Mask (Definition 5) preserves gradient signal for all structural tokens $\mathcal{T}_{\text{struct}}$ within the Result field. Under Assumption 5, if the learning rate η satisfies $\eta < 2/\beta$, then gradient descent on $\mathcal{L}_{\text{SLM}}$ decreases the loss on structural tokens monotonically, ensuring:*

$$p_\theta^{(n+1)}(\texttt{\{R:} \mid z_{1:t}) \geq p_\theta^{(n)}(\texttt{\{R:} \mid z_{1:t}) - \delta_n \tag{12}$$

where $\delta_n \to 0$ as training progresses, and the model maintains a non-vanishing probability of emitting result delimiters.

Proof. By Definition 5, $M_t = 1$ for all $t \in \mathcal{T}_{\text{struct}}$, which includes the structural delimiters `{R:`, `}`, and the variable identifier tokens (e.g., `final_price=`). The SLM loss thus includes explicit supervision on these tokens:

$$\mathcal{L}_{\text{SLM}}^{\text{struct}}(\theta) = -\sum_{t \in \mathcal{T}_{\text{struct}}} \frac{M_t}{\sum_s M_s} \log p_\theta(z_{t+1} \mid z_{1:t}, x). \tag{13}$$

Under Assumption 5, the loss is β-smooth. For gradient descent with learning rate $\eta < 2/\beta$, the standard descent lemma gives:

$$\mathcal{L}_{\text{SLM}}^{\text{struct}}(\theta^{(n+1)}) \leq \mathcal{L}_{\text{SLM}}^{\text{struct}}(\theta^{(n)}) - \eta\left(1 - \frac{\eta\beta}{2}\right) \|\nabla_\theta \mathcal{L}_{\text{SLM}}^{\text{struct}}(\theta^{(n)})\|^2. \tag{14}$$

Since $\eta < 2/\beta$, the coefficient $\eta(1 - \eta\beta/2) > 0$, ensuring monotonic decrease of the structural loss. As $\mathcal{L}_{\text{SLM}}^{\text{struct}}$ decreases, $\log p_\theta(z_{t+1} \mid z_{1:t}, x)$ increases for structural tokens, which means $p_\theta(\texttt{\{R:} \mid z_{1:t})$ is driven upward by the training signal. This prevents generation collapse (Definition 7), as the model is continually supervised to produce the result delimiter with high probability. The term $\delta_n = O(\|\nabla \mathcal{L}^{(n)}\|^2)$ captures the diminishing correction as convergence is approached.

C.6 Convergence Analysis

Theorem 5 (Convergence Rate of SLM Training)**.** *Under Assumptions 4 and 5, with learning rate $\eta = 1/\beta$ and N gradient steps on the combined dataset $\mathcal{D}_{\text{Gold}} \cup \mathcal{D}_{\text{Neg}}$ with SLM, the expected gradient norm satisfies:*

$$\frac{1}{N}\sum_{n=0}^{N-1} \mathbb{E}\Big[\|\nabla_\theta \mathcal{L}_{\text{SLM}}(\theta^{(n)})\|^2\Big] \leq \frac{2\beta\big(\mathcal{L}_{\text{SLM}}(\theta^{(0)}) - \mathcal{L}_{\text{SLM}}^*\big)}{N} \tag{15}$$

where $\mathcal{L}_{\text{SLM}}^*$ is the infimum of the SLM loss. Furthermore, let $\rho = |\mathcal{D}_{\text{Neg}}|/|\mathcal{D}_{\text{Gold}}|$ denote the negative-to-gold ratio. The effective sample size for the planning policy scales as:

$$N_{\text{eff}}^{\text{plan}} = |\mathcal{D}_{\text{Gold}}| + |\mathcal{D}_{\text{Neg}}| = (1 + \rho) \cdot |\mathcal{D}_{\text{Gold}}|, \tag{16}$$

while the effective sample size for the execution policy remains $N_{\text{eff}}^{\text{exec}} = |\mathcal{D}_{\text{Gold}}|$.

Proof. The first inequality is the standard result for gradient descent on β-smooth functions. Under Assumption 5, $\mathcal{L}_{\text{SLM}}$ is β-smooth (the masking operation preserves smoothness since it only zeros out terms). With $\eta = 1/\beta$, the descent lemma yields:

$$\mathcal{L}_{\text{SLM}}(\theta^{(n+1)}) \leq \mathcal{L}_{\text{SLM}}(\theta^{(n)}) - \frac{1}{2\beta}\|\nabla_\theta \mathcal{L}_{\text{SLM}}(\theta^{(n)})\|^2. \tag{17}$$

Telescoping over N steps and rearranging gives the stated bound.

For the effective sample size, we use Corollary 1. The planning gradient receives contributions from all traces in $\mathcal{D}_{\text{Gold}} \cup \mathcal{D}_{\text{Neg}}$ (since $M_t = 1$ for planning tokens in both sets), yielding $N_{\text{eff}}^{\text{plan}} = (1 + \rho) \cdot |\mathcal{D}_{\text{Gold}}|$. The execution gradient receives contributions only from $\mathcal{D}_{\text{Gold}}$ (Theorem 3), yielding $N_{\text{eff}}^{\text{exec}} = |\mathcal{D}_{\text{Gold}}|$.

Remark 3. *Theorem 5 formalizes the data efficiency results observed empirically in Section 5.2. The planning policy benefits from a $(1 + \rho)$-fold increase in effective training data, which explains why models trained with ARU on 50% Gold data supplemented by negative traces outperform models trained on 100% Gold data alone (Figure 4). The scaling factor ρ directly quantifies the value extracted from negative data.*

C.7 Information-Theoretic Characterization

Lemma 2 (Information Content of Negative Traces)**.** *Let $\mathcal{D}_{\text{Neg}}$ be the set of counterfactually valid negative traces. The mutual information between the masked negative data and the optimal planning policy satisfies:*

$$I\big(\{(O_k, P_k)\}_{z \in \mathcal{D}_{\text{Neg}}};\ \pi^*\big) \geq H(\pi^*) - H(\pi^* \mid \mathcal{D}_{\text{Gold}}) - \epsilon_{\text{approx}} \tag{18}$$

where ϵ_{approx} accounts for the finite-sample approximation. That is, the planning signal from negative traces is complementary to (not redundant with) that of Gold traces.

Proof.[Proof (Sketch)] The operation sequences in $\mathcal{D}_{\text{Neg}}$ are verified to be valid strategies (Theorem 1), so they are drawn from the support of π^*. The teacher's sampling at high temperature ($T = 1.2$) produces diverse operation sequences that explore alternative valid strategies rarely represented in $\mathcal{D}_{\text{Gold}}$ (which is concentrated on the teacher's mode). By the data processing inequality and the fact that SLM extracts exactly the planning tokens from these traces, the mutual information with the optimal policy is bounded below by the entropy reduction achieved beyond what $\mathcal{D}_{\text{Gold}}$ already provides. The ϵ_{approx} term arises from finite-sample estimation and vanishes as $|\mathcal{D}_{\text{Neg}}| \to \infty$.

Proposition 2 (Orthogonality of Negative Data Signal)**.** *Under the SLM mask, the gradient contributions from $\mathcal{D}_{\text{Gold}}$ and $\mathcal{D}_{\text{Neg}}$ to the planning parameters θ_P satisfy:*

$$\mathbb{E}\Big[\big\langle \nabla_{\theta_P} \mathcal{L}_{\text{SLM}}^{\text{Gold}},\ \nabla_{\theta_P} \mathcal{L}_{\text{SLM}}^{\text{Neg}} \big\rangle\Big] \geq 0. \tag{19}$$

That is, the planning gradients from Gold and Negative data are non-negatively correlated. The combined gradient has variance:

$$\begin{aligned} \text{Var}[\nabla_{\theta_P} \mathcal{L}_{\text{SLM}}] \leq \text{Var}\big[\nabla_{\theta_P} \mathcal{L}_{\text{SLM}}^{\text{Gold}}\big] + \text{Var}\Big[\nabla_{\theta_P} \mathcal{L}_{\text{SLM}}^{\text{Neg}}\Big] \\ + 2\,\mathbb{E}\Big[\big\langle \nabla_{\theta_P} \mathcal{L}_{\text{SLM}}^{\text{Gold}},\ \nabla_{\theta_P} \mathcal{L}_{\text{SLM}}^{\text{Neg}} \big\rangle\Big]. \end{aligned} \tag{20}$$

Proof. Both $\mathcal{D}_{\text{Gold}}$ and $\mathcal{D}_{\text{Neg}}$ contain valid operation sequences (the former by correctness, the latter by Theorem 1). The planning gradient for both datasets points toward the optimal planning policy π^*, since both supervise the model on valid operations. For any two gradient vectors that both point toward the same optimum, their inner product is non-negative in expectation (as they lie within the same half-space of the parameter space centered at θ_P^*). The variance decomposition follows from the standard identity $\text{Var}[A + B] = \text{Var}[A] + \text{Var}[B] + 2\,\text{Cov}[A, B]$ with $\text{Cov} \geq 0$.

C.8 Phase Transition in Nano-Scale Models

Theorem 6 (Phase Transition Condition)**.** *Let $C_{\min}^P$ denote the minimum planning capacity required to achieve accuracy α on a benchmark with K-step reasoning problems. Under standard SFT, a model of total capacity C achieves accuracy α only if:*

$$C \geq C_{\min}^P + C_{\min}^E(K), \tag{21}$$

where $C_{\min}^E(K)$ is the minimum execution capacity for K-step arithmetic, which grows as $C_{\min}^E(K) = \Omega(K \cdot d_{\text{arith}})$ where d_{arith} is the capacity per arithmetic operation. Under ARU + Solver, the condition relaxes to $C \geq C_{\min}^P$, eliminating the dependence on K and d_{arith}. The "phase transition" occurs when:

$$C_{\min}^P \leq C < C_{\min}^P + C_{\min}^E(K). \tag{22}$$

In this regime, the model has sufficient capacity for planning but insufficient capacity for joint planning and execution. ARU + Solver unlocks reasoning by reallocating execution capacity to planning.

Proof. Under standard SFT, the model must learn both $\pi_{\theta_P}(O_k \mid \cdot)$ and $\pi_{\theta_E}(R_k \mid \cdot)$. The execution policy requires representing K arithmetic operations, each requiring d_{arith} parameters (to store the function approximation for addition, multiplication, etc., to the required precision). This yields $C_{\min}^E(K) = \Omega(K \cdot d_{\text{arith}})$.

Under ARU + Solver, the execution function g is replaced by the deterministic solver $\mathcal{S}$, so $C_E = 0$ and all model capacity is available for the planning policy. The transition occurs at the critical capacity $C^* = C_{\min}^P + C_{\min}^E(K)$: models with $C < C^*$ fail under standard training but succeed under ARU + Solver if $C \geq C_{\min}^P$.

For the 0.5 B Qwen3 student, the empirical phase-transition sweep (Table 5) shows that $C = 0.5$ B is insufficient for joint planning–execution under standard SFT (Gold-SFT GSM8K accuracy 24.1%) but sufficient for planning alone under ARU (GSM8K accuracy 51.2%, matching Gold-SFT at 1.5 B). This places the 0.5 B model in the phase-transition regime $C_{\min}^P \leq 0.5\,\text{B} < C_{\min}^P + C_{\min}^E(K)$.

corollary 2 (Scaling Law Implication)**.** *The performance gap $\Delta(C)$ between ARU + Solver and standard SFT for a model of capacity C is:*

$$\Delta(C) = \begin{cases} 0 & \text{if } C \geq C^P_{\min} + C^E_{\min}(K), \\ \alpha_{\mathrm{ARU}}(C) - \alpha_{\mathrm{SFT}}(C) > 0 & \text{if } C^P_{\min} \leq C < C^P_{\min} + C^E_{\min}(K), \\ 0 & \text{if } C < C^P_{\min}. \end{cases} \tag{23}$$

The gap is maximized in the phase transition regime and vanishes for sufficiently large models (where execution capacity is abundant) and sufficiently small models (where even planning capacity is insufficient).

C.9 Comparison with Preference-Based Methods

Proposition 3 (Supervision Density Advantage over DPO)**.** *Let $\mathcal{L}_{\mathrm{DPO}}$ denote the Direct Preference Optimization loss operating on paired traces (z^+, z^-). The number of informative gradient updates per training example satisfies:*

$$\frac{|\{t : M_t = 1,\ t \in \mathcal{T}_P \cup \mathcal{T}_O\}|}{T} \gg \frac{1}{T}. \tag{24}$$

That is, SLM provides dense, token-level supervision on planning tokens, while DPO provides a single scalar reward signal per trace. For a trace of length T with K ARU steps, SLM produces $O(K \cdot \bar{L})$ informative gradient updates (where $\bar{L}$ is the average token length of a Premise-Operation pair), compared to $O(1)$ for DPO.

Proof. DPO optimizes the log-likelihood ratio $\log \frac{\pi_\theta(z^+)}{\pi_\theta(z^-)}$, which produces a single scalar comparison per trace pair. Although the gradient flows through all tokens, the signal is diluted across the entire sequence without distinguishing planning from execution tokens.

SLM, by contrast, directly supervises each planning token $t \in \mathcal{T}_P \cup \mathcal{T}_O$ with its own cross-entropy term. For K ARU steps, the Premise and Operation fields contribute $\Theta(K \cdot \bar{L})$ individually supervised tokens. Since $K \cdot \bar{L} \gg 1$ for multi-step reasoning (typically $K \geq 3$ and $\bar{L} \geq 5$), the supervision density of SLM exceeds that of DPO by a factor of $\Theta(K \cdot \bar{L})$. This is consistent with the empirical dominance of ARU over Step-DPO in Table 1: averaged across the eight benchmarks, ARU exceeds Step-DPO by $+14.1$ points at $0.6\,$B, $+16.7$ at $1.7\,$B, and $+16.9$ at $8\,$B—a persistent advantage of the magnitude one would expect from a roughly $K\bar{L}$-fold increase in informative gradient updates per trace.

D Limitations

While the empirical and theoretical evidence in this paper supports the central claims of the ARU framework, there are several non-trivial limitations that we believe are important to state explicitly. We organise them along the four axes that we judge most likely to circumscribe the immediate generality of our approach.

Domain coverage is bounded by the deterministic solver. Counterfactual Logic Verification depends on a Python solver $\mathcal{S}$ that is complete for the operations encountered in the benchmark domains: addition, subtraction, multiplication, division, exponentiation, modular arithmetic, and their compositions (Assumption 2). This is sufficient for grade-school through competition mathematics, but is *not* sufficient for reasoning domains that require non-deterministic execution (e.g., constraint search), continuous optimisation, real-world physics, or open-ended commonsense inference. In these regimes the "Case A vs Case B" partition is no longer well-defined, and the user would have to either (i) restrict CLV to the verifiable sub-trace, (ii) substitute a probabilistic verifier whose false-positive rate is not the 1.2% measured in §5.4, or (iii) abandon hard verification in favour of a learned process reward. The ProofWriter result in §5.2 is therefore best read as evidence that ARU is honest about *not* helping outside its intended regime, rather than as evidence that it generalises everywhere.

Compensating-error traces leak through CLV. By construction, CLV admits any trace whose symbolic re-execution produces y^*, but a trace that contains *two* compensating arithmetic errors may also satisfy this criterion. Our adversarial probe in Table 3 measures this leakage at 1.2% for

DeepSeek-V4-Pro and 1.5% for Qwen3.5-397B-A17B. While small, this is a non-zero corruption channel that grows with teacher error rate and is in principle unbounded for adversarial teachers. An obvious follow-up is to require not just $y_{\text{rect}} = y^*$ but also that no intermediate R_k^* disagree with the corresponding teacher value beyond a tolerance proportional to numerical scale.

Inference cost of structured generation. The BNF grammar increases inference token-length by $\approx 18\%$ on average (Table 8). On the training side this is dominated by the $\approx 3.7\times$ reduction in teacher generation, but on the inference side it is a real and persistent overhead that disadvantages ARU in latency-bounded deployments unless paired with constrained decoding. Our reported numbers do not assume constrained decoding at inference; we expect that integrating ARU with grammar-aware decoders such as XGrammar Dong et al. [2024] or SynCode Ugare et al. [2024] would close most of this gap, but we leave a careful study to future work. We also note that we have not benchmarked ARU at the scale of $\geq$ 70B teachers or students, where the relative cost structure of generation vs. decoding may differ from the 0.5–8B regime we evaluate.

Generalisation beyond mathematical reasoning is partially conjectural. ARU's strongest claims are tied to the decomposition $z = (P, O, R)$, which maps cleanly onto arithmetic but only metaphorically onto domains such as code synthesis, theorem proving, or scientific reasoning. Although our BBH-Arith and MMLU-STEM numbers (Table 1) suggest that the cleaner planning subspace transfers, we do not have direct evidence that the framework recovers the same gains for, e.g., natural-language entailment or instruction-following. Designing analogous Premise/Operation/Result decompositions for non-arithmetic reasoning, and verifying the resulting traces with appropriately specialised solvers (proof assistants, type checkers, unit tests), is a natural next step but remains future work.

Single-architecture, single-language scope. All main-table results are on the Qwen3 family, with a single Llama-3.2 tokenizer-robustness check (Table 6). We have not run ARU end-to-end on Mistral, Gemma, or Phi families, and all problems are in English. We expect the syntactic-anchor mechanism to be language-agnostic in principle, but the BPE-arithmetic obfuscation argument we cite from Zhang et al. [2025] interacts non-trivially with multilingual tokenizers and remains untested.

Phase-transition framing is empirical, not theoretical. Although the empirical phase-transition result in §5.2 establishes the existence of a small-scale capability gap and rules out the most obvious alternative explanation via the ProofWriter control, we do not claim a closed-form characterisation of when the transition will appear or how its magnitude scales with teacher quality, problem difficulty, or operator vocabulary. The term "phase transition" is therefore used in an operational sense (a sharper-than-log-linear gap-to-baseline at small sizes), not in the strict statistical-physics sense; we have softened the framing accordingly throughout the paper.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and the five-bullet contribution list in §1 each scope their claims to the demonstrated setting (the $3 \times 8 \times 6$ grid of $\{0.6, 1.7, 8\}$ B Qwen3 students $\times$ eight benchmarks $\times$ six baselines), cite specific theorems for the formal results (Theorems 3, 4, Corollary 1, Proposition 3), and explicitly note the residual leakage channels (Remark 2) and the limits of the ProofWriter control. Headline numbers (e.g., $+9.8$ MATH-500 over PoT at 1.7 B; $\approx 5\times$ trace-budget reduction; $48.2 \to 89.4$ tool-use lift) are anchored to specific tables/figures in §5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: A dedicated Limitations section (Appendix D) discusses six axes: (i) domain coverage bounded by the deterministic solver; (ii) compensating-error leakage through CLV (1.2% adversarial false positives); (iii) inference token-length overhead from BNF delimiters; (iv) partially-conjectural generalisation beyond mathematical reasoning; (v) single-architecture/single-language scope; and (vi) the operational rather than statistical-physics meaning of "phase transition." Limits of the factored-capacity assumption and of the ProofWriter control are also discussed in-line in Remark 2 and §5.2.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: All assumptions are stated explicitly in the Theoretical Analysis appendix (Assumptions 1–5). Each formal result names the assumptions it depends on in its statement; e.g., Theorem 3 is stated as architecture-agnostic (no parameter-decomposition assumption), while Corollary 1 explicitly invokes Assumption 4. Remark 2 discusses the limits of the factored-capacity idealisation and the residual leakage channels. Full proofs accompany each theorem, lemma, corollary, and proposition (Theorems 3, 4, 5, 6; Corollaries 1, 2; Propositions 1, 2, 3).

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: §4 in the main paper gives the compressed setup and points to Appendix A (Experimental Setup, Full), which specifies all training and evaluation details: the three training data sources and their splits, the held-out validation split $\mathcal{D}_{\text{val}}$, teacher generation parameters ($K = 8$ samples per problem, $T = 0.7$, top-$p = 0.95$, 4,096-token cap), the SymPy-based answer-matching procedure, the regex BNF extractor and its 98.4% parse-success rate, the AdamW hyperparameters, the learning-rate sweep, the batch size, the epoch budget, the seed counts, and the per-experiment student-size assignments. The mask construction is given in §3.4 (`[VAL_BEGIN]`/`[VAL_END]` reserved tokens, mask deterministic at the token-ID level). Code is publicly released (see "Open access" below).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often

one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: A public code release is linked from the abstract (`https://anonymous.4open.science/r/ARU-286D`). The repository contains the BNF extractor source, the symbolic solver implementation used by CLV, the validation-split indices for $\mathcal{D}_{\text{val}}$, the random seeds, the generation-temperature settings, the answer-canonicalisation rules, and training/evaluation scripts for all reported experiments. All training datasets (GSM8K, MATH, NuminaMath-CoT) and all evaluation benchmarks (MATH-500, AIME, OlympiadBench, GSM-Hard, GSM-Symbolic, BBH, MMLU-STEM, ProofWriter) are publicly available; the appendix cites each source.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: A 200-word summary appears in §4 (datasets, students, teacher, metrics, baselines), with full specification in Appendix A: data splits (canonical train/test plus the held-out $\mathcal{D}_{\text{val}}$, sampled disjoint from every reported test set including MATH-500); optimiser (AdamW, β_1=0.9, β_2=0.95, weight decay 0.01, dropout 0.1, cosine schedule); learning-rate sweep ($\{1\times10^{-5}, 5\times10^{-5}, 1\times10^{-4}\}$ on the 1.7 B student, selected on $\mathcal{D}_{\text{val}}$, fixed at 5×10^{-5} for all other configurations); batch size (128 via gradient accumulation); epochs (3); checkpoint selection (lowest cross-entropy on $\mathcal{D}_{\text{val}}$, disjoint from MATH-500); test-set evaluation (canonical splits, used exactly once).

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: All reported numbers are means over 3 random seeds for the main table and ablations and over 5 seeds for the phase-transition sweep (where small-model variance is highest). Standard deviations are reported in Appendix Tables 5 and 2; the §5.1 prose reports the maximum observed seed std across the 24 main-table cells (1.1 on AIME at 0.6 B) with most cells below 0.7. Significance against the strongest baseline per column uses a two-sided paired bootstrap test at $p < 0.05$, marked with † in Table 1. The phase-transition matched-accuracy claims explicitly use Bonferroni correction across the six size cells.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The Implementation Details paragraph in Appendix A reports that all student training was performed on $8\times$NVIDIA H100 (80 GB) GPUs, with the complete 8 B run requiring approximately 45 wall-clock hours. CLV preprocessing overhead is reported as $\approx$ 14.5 ms per trace (CPU-side Python solver), and the end-to-end pipeline FLOPs accounting (teacher generation + CLV + student training + inference at the matched-accuracy threshold) is reported in Appendix Table 8.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The work conforms to the NeurIPS Code of Ethics. It is methodological research on reasoning distillation using publicly released models and datasets; it neither collects new sensitive data nor produces new models with known misuse vectors beyond those already present in the underlying base models.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: *Positive impacts.* ARU reduces the teacher-trace generation budget by $\approx 5\times$ on the GSM8K-1.7B operating point and the end-to-end pipeline by $\approx 0.66\times$ at matched accuracy (Appendix Table 8), lowering both the compute and energy footprint of producing strong reasoning students. The framework also supports smaller, more accurate open models, which broadens access to reasoning-capable LLMs. *Negative impacts.* As with any general improvement to small-scale reasoning capabilities, ARU could be applied to downstream tasks with dual-use potential (e.g., automated content generation). The paper does not introduce new misuse vectors beyond those already present in the underlying base models (Qwen3-Base, Llama-3.2) and teachers (DeepSeek-R1 line), and the released student checkpoints inherit the safety filters of those base models.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: We release reasoning-distilled student checkpoints that are mathematics-specialised fine-tunes of publicly available open-weight base models (Qwen3-Base, Llama-3.2) and that do not introduce new high-risk capabilities (e.g., open-ended generation of harmful content, code-execution tooling beyond a sandboxed Python solver) over and above the base models. No scraped data is released; all training datasets (GSM8K, MATH, NuminaMath-CoT) are existing public assets used under their original licences.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All datasets and base models used are cited in §4 and Appendix A: GSM8K Cobbe et al. [2021] (MIT), MATH Hendrycks et al. [2021b] (MIT), NuminaMath-CoT Li et al. [2024] (Apache-2.0), MATH-500 Lightman et al. [2024], AIME (public competitions), OlympiadBench He et al. [2024] (CC-BY-NC-SA-4.0), GSM-Hard Gao et al. [2023] (Apache-2.0), GSM-Symbolic Mirzadeh et al. [2025] (Apple research licence), BBH Suzgun et al. [2023] (Apache-2.0), MMLU Hendrycks et al. [2021a] (MIT), ProofWriter Tafjord et al. [2021] (CC-BY-4.0), Qwen3-Base Yang et al. [2025] (Apache-2.0), Llama-3.2 Grattafiori et al. [2024] (Llama community licence), and DeepSeek-R1 Guo et al. [2025] (MIT). All assets are used in compliance with their respective licences. Math-Shepherd Wang et al. [2024] is reimplemented as a baseline filter from the published method.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.

- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The code repository linked from the abstract bundles the new assets introduced by this work: the BNF extractor source, the symbolic CLV solver, the validation-split indices for $\mathcal{D}_{\text{val}}$, the trained ARU student checkpoints, and reproduction scripts. Each is accompanied by a README documenting input/output formats, environment specification, expected runtime, and licence (MIT for code; the trained student checkpoints inherit the licence of their base model).

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper includes a 300-trace human-CLV-agreement analysis (Appendix Table 3, Cohen's $\kappa = 0.89$); these labels were produced by the authors as a method-validation step rather than via crowdsourcing or external participant recruitment. No compensated workers and no human subjects (in the IRB sense) were involved.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The work does not involve human subjects in the IRB sense. The only human-rated artefacts are the 300 trace labels used for the CLV agreement analysis (Appendix Table 3), produced internally by the authors and not from external participants.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: LLMs are central to the methodology and are explicitly declared. Teachers: DeepSeek-V4 Pro DeepSeek-AI [2026] and Qwen3.5 397B-A17B (used in § 5.4 for teacher-dependence) Qwen Team [2026]. Students: Qwen3-Base at $\{0.5, 0.6, 1.0, 1.5, 1.7, 3.0, 7.0, 8.0\}$ B Yang et al. [2025] and Llama-3.2 Grattafiori et al. [2024] (1 B, 3 B). The role of each in the pipeline (teacher generation, student SFT, evaluation) is specified in §4 and Appendix A. LLMs were not used as authoring tools to write the paper.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.