# HINTMR: ELICITING STRONGER MATHEMATICAL REASONING IN SMALL LANGUAGE MODELS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Small language models (SLMs) often struggle with complex mathematical reasoning, where errors made early in a solution can compound over long reasoning trajectories. We introduce HintMR, a hint-assisted reasoning framework that elicits stronger reasoning in SLMs by providing localized guidance throughout multi-step problem solving. At each reasoning step, a separate language model generates a context-aware hint conditioned on the problem and the solver's accumulated reasoning history. Rather than revealing the full solution, these hints provide targeted intermediate guidance that helps the SLM stay on a productive reasoning trajectory while continuing to generate the solution itself. We further introduce Adaptive HintMR (AdaHintMR), an efficient variant that selectively invokes the hint model only when additional guidance is likely to be useful. AdaHintMR estimates the solver's step-level uncertainty using token-level Shannon entropy and requests a new hint only when uncertainty exceeds a predefined threshold, allowing the SLM to reason independently on confident steps. Experiments across six mathematical reasoning benchmarks show that HintMR consistently improves reasoning accuracy over standard prompting and independent SLM reasoning. When paired with GPT-5.4 as the frontier hinter, HintMR substantially improves the reasoning performance of SLM solvers across multiple benchmarks, outperforming GPT-5.4 used directly as a standalone solver by up to 6.67 percentage points. These gains show that using a frontier model for targeted intermediate guidance can be more effective than relying on it to solve the entire problem directly. AdaHintMR further reduces reliance on the frontier model by selectively requesting hints only when the solver exhibits high uncertainty. Across the evaluated solvers and benchmarks, this reduces API cost by up to 72.4% relative to HintMR while maintaining strong reasoning performance. Overall, our results show that targeted intermediate hints can effectively elicit stronger mathematical reasoning from SLMs, while uncertainty aware hinting preserves much of this benefit with substantially lower frontier model inference cost.

## 1 INTRODUCTION

Large language models (LLMs) have made substantial progress in mathematical reasoning, from elementary arithmetic to competition-level mathematics (Hendrycks et al., 2021; Cobbe et al., 2021; Lewkowycz et al., 2022). Chain-of-thought (CoT) reasoning has contributed to this progress by allowing models to generate intermediate reasoning steps before producing a final answer (Wei et al., 2022). Despite these advances, small language models (SLMs) remain attractive because their lower computational and memory requirements support local, mobile, and edge deployment, where cloud-hosted models may be limited by cost, latency, or connectivity (Liu et al., 2024; Van Nguyen et al., 2025). Improving SLM reasoning therefore provides a complementary direction to relying exclusively on increasingly large frontier models.

However, SLMs often struggle with complex multi-step mathematical reasoning. An incorrect intermediate step can propagate through the remaining trajectory and lead to an incorrect final answer. Existing approaches improve reasoning through inference-time search (Yao et al., 2023), external tools (Gao et al., 2023; Schick et al., 2023), or specialized training (Ouyang et al., 2022), but these may introduce additional computational or implementation overhead. We instead investigate whether localized guidance can support the solver's intermediate reasoning while leaving it respon-



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

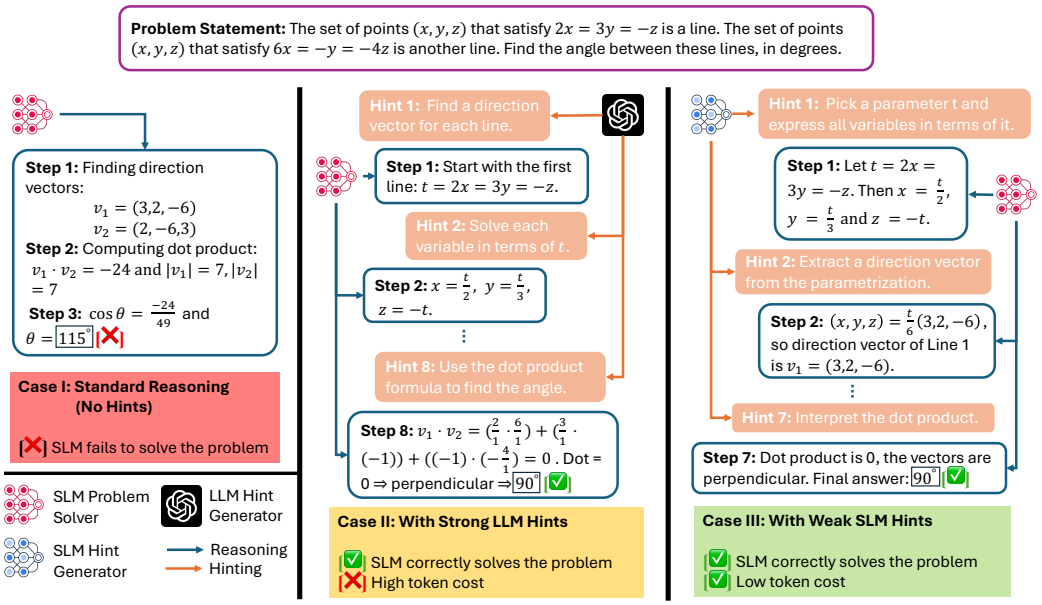


Figure 1: Overview of unassisted and hint-assisted SLM reasoning. HintMR uses a local or frontier-model hinter to provide localized guidance conditioned on the problem and accumulated reasoning, while the SLM constructs the reasoning trajectory and final answer.

sible for constructing the solution. This idea is motivated by human problem solving, where a learner may possess much of the required knowledge but benefit from a hint that redirects an unproductive reasoning path without revealing the complete solution (Chi et al., 2001). For language models, such guidance can be conditioned on the reasoning generated so far. This motivates our central question: *Can targeted, context-aware intermediate hints elicit stronger mathematical reasoning from SLMs while leaving them responsible for constructing their own solutions?*

We introduce **HintMR** (Figure 1), a hint-assisted reasoning framework that interleaves a solver's stepwise reasoning with guidance from a separate hinter. Before each reasoning step, the hinter receives the original problem and the accumulated reasoning history. It is prompted to suggest a useful next direction, such as a concept, transformation, or intermediate action, rather than provide a complete derivation or final answer. The solver incorporates this guidance and generates the next step itself. Thus, the hinter acts as a collaborative tutor rather than a replacement solver, with hints adapted to the trajectory actually produced instead of following a predefined sequence.

We study two complementary hinter settings. A frontier-model hinter evaluates how much strong intermediate guidance can improve an SLM without solving the entire problem itself, while an open-source local hinter tests whether hint-assisted reasoning remains effective without external API access. Together, these settings examine both the effectiveness of stronger guidance and the feasibility of fully local collaboration. Because not every reasoning step requires a new hint, we further introduce adaptive HintMR (**AdaHintMR**, Figure 2). AdaHintMR retains the initial hint but selectively requests later guidance based on the solver's step-level token entropy. This entropy serves as an uncertainty heuristic rather than a measure of mathematical correctness, allowing us to study whether targeted hinting can preserve reasoning gains while reducing unnecessary hinter calls. Our main contributions are:

- **Context-aware hint-assisted reasoning.** We introduce HintMR and study both frontier-model and open-source local hinters for strengthening SLM mathematical reasoning through intermediate guidance.
- **Adaptive hint invocation.** We develop AdaHintMR, which uses an entropy-based gate to selectively request subsequent hints rather than invoking the hinter at every reasoning step.
- **Evaluation of effectiveness and efficiency.** Experiments across six mathematical reasoning benchmarks show improved SLM performance with hint assistance. With GPT-5.4 as the hinter,

HintMR outperforms standalone GPT-5.4 by up to 6.67 percentage points. AdaHintMR reduces external API cost by up to 72.4% relative to HintMR while preserving strong reasoning performance across different solver settings.

Beyond SLMs, this study motivates investigating intermediate guidance as a resource for mathematical reasoning more broadly. Recent progress in competition-level and research-oriented mathematics makes this direction increasingly relevant (Luong & Lockhart, 2025; OpenAI, 2026). Although our experiments focus on SLM solvers, a natural next question is whether already-capable frontier models can also benefit from context-aware hints on harder problems. Such collaboration could support the exploration, verification, and revision of intermediate ideas without requiring one model to supply a complete solution. Investigating this possibility would extend hint-assisted reasoning from improving small-model reliability toward more challenging and potentially research-level mathematical problem solving.

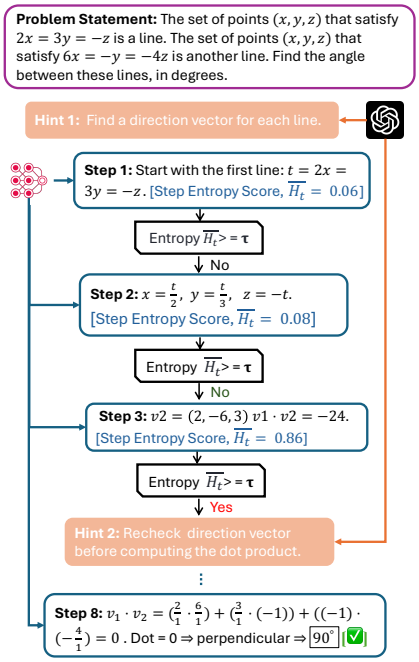


Figure 2: AdaHintMR framework. An initial hint guides the first solver step. After each step, the solver's token entropy determines whether a new hint is requested for the next step. A hint is triggered when the step-level entropy exceeds the threshold $\tau = 0.1$. All generated steps remain in the reasoning history; the gate controls hint invocation rather than step acceptance.

## 2 Related Work

We briefly summarize the work most closely related to HintMR, with a more detailed discussion provided in Appendix A.

**Data and training for mathematical reasoning.** A major direction in mathematical reasoning has been to scale supervision using synthetic problems and step-by-step solutions. MetaMath (Yu et al., 2024), MathInstruct (Yue et al., 2024), OpenMathInstruct (Toshniwal et al., 2024; 2025), and NuminaMath (Li et al., 2024) demonstrate the value of large reasoning-oriented training corpora. More recent efforts continue to expand both the scale and quality of mathematical data. MegaMath (Zhou et al., 2025) combines web-scale mathematical text, code, and synthetic data, while AoPS-Instruct (Mahdavi et al., 2025) extracts large-scale Olympiad-level problem–solution pairs and introduces a continuously updated benchmark for contamination-resistant evaluation. TMATH (Qi et al., 2025) is particularly relevant to our setting because it explicitly studies problem–hint pairs, highlighting the value of well-scoped intermediate guidance.

**Hint-guided and adaptive reasoning.** Prior work has shown that intermediate guidance can improve mathematical reasoning. Progressive-Hint Prompting (Zheng et al., 2023) and Hint-before-Solving (Fu et al., 2024) introduce hints through multi-turn or hint-first prompting strategies. More recently, ConciseHint (Tang et al., 2025) injects adaptive textual hints during reasoning to reduce overly long reasoning traces, while Hint Tuning (Fan et al., 2026) uses no-hint, sparse-hint, and full-hint training examples to teach models to adjust reasoning depth according to problem difficulty. Our setting differs in that HintMR uses a separate *hinter* conditioned on the solver's evolving reasoning trajectory to provide localized guidance during inference. AdaHintMR further uses the solver's step-level uncertainty to determine when a new hint should be requested, without requiring additional training of either model.

**Efficient reasoning in smaller models.** Recent work has also shown that strong mathematical reasoning can be elicited from relatively compact models. DeepSeek-R1 (Guo et al., 2025) transfers reasoning capabilities to smaller distilled models, while Phi-4-mini-reasoning (Xu et al., 2025) develops a dedicated training recipe for a 3.8B-parameter reasoning model. rStar-Math (Guan et al., 2025) combines small policy models with search and process-level guidance, showing that SLMs

can achieve strong competition-math performance. Other approaches focus directly on inference efficiency: s1 (Muennighoff et al., 2025) controls test-time reasoning through budget forcing, while SABER (Zhao et al., 2026) trains models to operate under different reasoning budgets. In contrast, our approach does not increase search depth or retrain the solver for a fixed reasoning budget. Instead, it selectively introduces external intermediate guidance when the solver is likely to benefit from it.

# 3 METHODOLOGY

Our framework separates *how to provide intermediate guidance* from *when to request it*. HintMR interleaves context-aware hints with stepwise reasoning, while its adaptive variant, AdaHintMR, selectively requests subsequent hints using an entropy-based invocation gate. Both methods use a solver to generate the reasoning trajectory and final answer, with a separate hinter providing localized guidance.

## 3.1 HINTMR: CONTEXT-AWARE STEPWISE HINTING

Given a mathematical problem $P$, a solver $f_\theta$ constructs a solution as a sequence of reasoning steps $W = (W_1, \ldots, W_T)$. Let $W_{<t} = (W_1, \ldots, W_{t-1})$ denote the reasoning history before step $t$, with $W_{<1} = \varnothing$. Each solver invocation is prompted to generate one reasoning step. HintMR augments this process with a separate hinter $g_\phi$, which receives the problem and the current reasoning history and provides a hint for the next step:

$$h_t = g_\phi(P, W_{<t}), \tag{1}$$

$$W_t = f_\theta(P, W_{<t}, h_t), \qquad t = 1, \ldots, T. \tag{2}$$

The first interaction is the special case $h_1 = g_\phi(P, \varnothing)$: the hinter provides an initial direction based only on the problem, which the solver then uses to generate $W_1$.

The hinter is prompted to provide concise instructional guidance, such as a useful concept, transformation, or next action, rather than a complete derivation or final answer. Because each hint is conditioned on the accumulated reasoning, the guidance adapts to the trajectory actually generated by the solver rather than following a predefined sequence. The solver determines how to use each hint and carries out the corresponding mathematical reasoning.

After each invocation, $W_t$ is appended to the reasoning history. The solver receives the original problem, the accumulated reasoning, and the current hint; earlier hints are not separately retained in its prompt, although their effects may persist through the generated reasoning. The process terminates when the solver produces a final answer or reaches the configured generation limit, without requesting another hint after termination. We omit fixed instruction templates from the notation for clarity.

Standard HintMR requests a hint before every reasoning step. This provides continuous guidance but also invokes the hinter at steps where additional assistance may be unnecessary. AdaHintMR retains the same interaction protocol while making subsequent hint requests conditional on the solver's uncertainty.

## 3.2 ADAHINTMR: ENTROPY-GATED HINT INVOCATION

AdaHintMR begins with the same initial hint as HintMR. After each nonterminal solver step, it uses token-prediction entropy to decide whether to request guidance for the next step (Figure 2). Thus, adaptivity changes the frequency of hint invocation, not the roles of the solver and hinter.

**Step-level uncertainty.** For token position $j$ in step $W_t$, let $p_{t,j,k}$ denote the probability assigned by the solver to its $k$-th most likely candidate token. We retain the top $K$ candidates and renormalize their probabilities:

$$\hat{p}_{t,j,k} = \frac{p_{t,j,k}}{\sum_{\ell=1}^{K} p_{t,j,\ell}}. \tag{3}$$

**Algorithm 1** HintMR and AdaHintMR

**Require:** Problem $P$, solver $f_\theta$, hinter $g_\phi$, mode $m \in \{\mathrm{HintMR}, \mathrm{AdaHintMR}\}$, generation limits; top-$K$ parameter $K$ and threshold $\tau$ for AdaHintMR
**Ensure:** Reasoning trajectory $W$ and final answer, if produced

$W \leftarrow ()$
$h \leftarrow g_\phi(P, W)$ ▷ Initial hint in both modes
**while** generation budget remains **do**
  Generate one step $w \leftarrow f_\theta(P, W, h)$ ▷ $h = \varnothing$ means no hint
  Append $w$ to $W$
  **if** $w$ contains a final answer or the generation limit is reached **then**
    **break**
  **end if**
  **if** $m = \mathrm{HintMR}$ **then**
    $z \leftarrow 1$
  **else**
    Compute $\bar{H}(w)$ using equation 3–equation 5
    $z \leftarrow \mathbf{1}\left[\bar{H}(w) > \tau\right]$
  **end if**
  **if** $z = 1$ **then**
    $h \leftarrow g_\phi(P, W)$
  **else**
    $h \leftarrow \varnothing$
  **end if**
**end while**
**return** $W$ and final answer, if available

We compute entropy over this truncated distribution and average it across the tokens in the generated step:

$$H_{t,j} = -\sum_{k=1}^{K} \hat{p}_{t,j,k} \log \hat{p}_{t,j,k}, \tag{4}$$

$$\bar{H}_t = \frac{1}{|W_t|} \sum_{j=1}^{|W_t|} H_{t,j}, \tag{5}$$

where $|W_t|$ is the number of generated tokens in $W_t$. Higher $\bar{H}_t$ indicates greater dispersion in the solver's token predictions. We use this score as a heuristic for hint invocation, not as a direct measure of mathematical correctness.

**Selective hint invocation.** After generating a nonterminal step $W_t$, AdaHintMR compares $\bar{H}_t$ with a predefined threshold $\tau$:

$$z_t = \mathbf{1}\left[\bar{H}_t > \tau\right], \tag{6}$$

where $z_t = 1$ indicates that a new hint is requested. The hint for the next reasoning step is

$$h_{t+1} = \begin{cases} g_\phi(P, W_{\le t}), & z_t = 1, \\ \varnothing, & z_t = 0, \end{cases} \tag{7}$$

where $W_{\le t} = (W_1, \ldots, W_t)$. The solver then generates

$$W_{t+1} = f_\theta(P, W_{\le t}, h_{t+1}). \tag{8}$$

Here, $f_\theta(P, W_{\le t}, \varnothing)$ denotes a solver invocation without a hint. In particular, skipping a hint request does not cause the previous hint to be reused.

The score $\bar{H}_t$ therefore controls guidance for $W_{t+1}$, rather than whether to accept or regenerate $W_t$. All generated steps remain in the reasoning history: a high-entropy step is not explicitly discarded or regenerated, and a low-entropy step is not treated as verified. A triggered hint guides subsequent reasoning, which may include revisiting earlier conclusions. As in HintMR, termination is checked before any further hint request.

Algorithm 1 summarizes the shared inference procedure. HintMR uses unconditional hint invocation, whereas AdaHintMR applies the entropy gate after each nonterminal step. Prompt templates and inference settings are provided in Appendix B.

# 4 Experimental Setup

We evaluate HintMR and AdaHintMR on six mathematical reasoning benchmarks. Below, we describe the datasets, model configurations, baselines and HintMR variants, adaptive hinting setup, and evaluation protocol. Additional implementation details are provided in Appendix B.

## 4.1 Datasets

Our evaluation covers six benchmarks spanning competition mathematics and broader mathematical reasoning: AIME-2024 (Zhang & Math-AI, 2024), AIME-2025 (Zhang & Math-AI, 2025), AIME-2026 (Zhang & Math-AI, 2026), AMC-2023 (Math-AI, 2023), MATH-500 (Hendrycks et al., 2021), and Minerva-Math (Lewkowycz et al., 2022). These benchmarks cover a broad range of reasoning difficulty, problem types, and solution lengths. Additional benchmark details, including evaluation size, benchmark type, approximate difficulty, and mathematical coverage, are provided in Appendix B.3 and Table 5.

## 4.2 Models

Our framework separates the roles of the *solver* and the *hinter*. The solver generates the reasoning trajectory and final answer, while the hinter provides intermediate guidance during reasoning. We use *Qwen3.5-9B* (Qwen Team, 2026a) as the primary solver in our main experiments. To examine whether the framework generalizes across different solver architectures and scales, we additionally evaluate *Phi-4-mini* (Xu et al., 2025) and *Gemma-2-9B* (Team et al., 2024).

We consider both frontier and local hinter settings. For frontier-model guidance, we use *GPT-5.4* through the OpenAI API and evaluate it with all three SLM solvers. For fully local collaboration, we use *Qwen3.8-27B* (Qwen Team, 2026b) as the hinter with Qwen3.5-9B as the solver. This setting allows us to evaluate HintMR without relying on an external API.

## 4.3 Baselines and HintMR Variants

We compare HintMR and AdaHintMR against standalone reasoning, inference-time scaling, and prior hint-based approaches, including No Hint, GPT-5.4 Direct Solver, Self-Consistency (Wang et al., 2022), HSP (Fu et al., 2024), and TMATH (Qi et al., 2025). No Hint uses independent solver reasoning, GPT-5.4 serves as a frontier standalone reference, Self-Consistency aggregates multiple independent generations by majority vote, HSP provides a single hint before reasoning, and TMATH uses learned hint generation. We also evaluate Entropy-Gated HintMR, which invokes the hinter only when uncertainty is high, and Roadmap-Guided HintMR, which combines an initial solution plan with entropy-triggered hints. Detailed descriptions are provided in Appendix B.5.

## 4.4 Evaluation Protocol

**Answer accuracy.** For each problem, we extract the final predicted answer and compare it with the reference answer. We use a three-stage verification process. We first apply normalized *Exact Match*. Predictions that do not match textually are then checked with *SymPy Match* to identify mathematically equivalent expressions with different surface forms. Any cases that remain unresolved are manually reviewed for mathematical equivalence. The resulting binary correctness labels are used to compute the final accuracy for each method and benchmark.

**API cost.** For methods that invoke GPT-5.4, we also report API cost in U.S. cents. We compute the cost from the prompt and completion tokens consumed across frontier-model calls and report the average cost per problem for each benchmark (¢/problem). Local open-source hinters do not incur API cost. Together, accuracy and API cost let us study the two main questions in our evaluation: whether intermediate hints improve SLM reasoning, and whether uncertainty-aware hinting can retain those gains while reducing reliance on frontier-model inference.

Table 1: Accuracy (%) across six mathematical reasoning benchmarks using Qwen3.5-9B as the solver and Qwen3.8-27B as the local hinter. We compare the No-Hint baseline, standard HintMR, AdaHintMR, Entropy-Gated HintMR (E-G HintMR), and Roadmap-Guided HintMR (R-G HintMR). Bold values indicate the highest accuracy for each benchmark, including ties.

| Dataset | No Hint | HintMR | AdaHintMR | E-G HintMR | R-G HintMR |
|---|---|---|---|---|---|
| AIME-2024 | 26.67 | **40.00** | **40.00** | 36.67 | 36.67 |
| AIME-2025 | 13.33 | 26.67 | **30.00** | 26.67 | 26.67 |
| AIME-2026 | 26.67 | 30.00 | **40.00** | 30.00 | 30.00 |
| AMC-2023 | 55.00 | **60.00** | **60.00** | 57.50 | 57.50 |
| MATH-500 | 74.40 | 79.20 | 80.60 | **81.00** | 79.60 |
| Minerva-Math | 53.68 | **58.09** | **58.09** | 55.51 | 54.04 |

## 5 Results and Analyses

We evaluate HintMR and AdaHintMR under both local- and frontier-hinter settings. We first consider a fully local configuration with Qwen3.5-9B as the solver and Qwen3.8-27B as the hinter. We then use GPT-5.4 as the frontier hinter and evaluate the framework with three different SLM solvers: Qwen3.5-9B, Phi-4-mini, and Gemma-2-9B. This allows us to examine both the effectiveness of intermediate guidance and the accuracy–cost trade-off under different solver configurations. Additional adaptive-hinting results and qualitative examples are provided in Appendices C and D, respectively.

### 5.1 HintMR and AdaHintMR

Table 1 reports results for the fully local configuration, where Qwen3.5-9B is used as the solver and Qwen3.8-27B as the hinter. Across all six benchmarks, HintMR and all adaptive variants improve over independent No-Hint reasoning, demonstrating that effective intermediate guidance does not require access to a frontier model.

The gains are particularly clear on the AIME benchmarks. Relative to No Hint, AdaHintMR improves accuracy from 26.67% to 40.00% on AIME-2024, from 13.33% to 30.00% on AIME-2025, and from 26.67% to 40.00% on AIME-2026. The same trend extends to the broader benchmarks, improving from 55.00% to 60.00% on AMC-2023, from 74.40% to 80.60% on MATH-500, and from 53.68% to 58.09% on Minerva-Math.

AdaHintMR matches or improves upon standard HintMR on every benchmark, matching it on AIME-2024, AMC-2023, and Minerva-Math while improving AIME-2025, AIME-2026, and MATH-500. Among the adaptive variants, AdaHintMR achieves the highest accuracy on five of the six benchmarks, while E-G HintMR performs best on MATH-500 with 81.00%. E-G HintMR and R-G HintMR also consistently outperform No Hint, showing that multiple forms of selective guidance can benefit local solver–hinter collaboration. Overall, these results show that adaptive hint selection can retain, and in several cases improve, the gains of standard HintMR while using entirely locally deployed models.

### 5.2 Accuracy–Cost Efficiency

We next examine HintMR and AdaHintMR with GPT-5.4 as the frontier hinter. Table 2 reports accuracy and average API cost for all three SLM solvers. Across every solver–benchmark combination, AdaHintMR requires less frontier-model expenditure than standard HintMR, showing that entropy-based gating consistently reduces the cost of intermediate guidance.

For the primary Qwen3.5-9B solver, AdaHintMR provides a favorable accuracy–cost trade-off. It matches HintMR on AIME-2024 (76.67%) while reducing cost from 1.21¢ to 1.01¢, and improves AIME-2025 accuracy from 50.00% to 63.33% while lowering cost from 1.44¢ to 1.04¢. On AIME-2026, both methods reach 66.67%, with AdaHintMR reducing cost from 1.57¢ to 1.03¢. The largest reduction occurs on AMC-2023, where both achieve 92.50% accuracy while cost decreases from 0.83¢ to 0.32¢, a 61.4% reduction. AdaHintMR also reduces cost on MATH-500 and Minerva-Math,

Table 2: Accuracy and API cost per problem (¢) for Qwen3.5-9B, Phi-4-mini (3.8B), and Gemma-2-9B (9B) paired with GPT-5.4 as the frontier hinter. GPT-5.4 used directly as a standalone solver is included as a reference. Within each SLM solver, bold denotes the better result between HintMR and AdaHintMR for each metric, including ties. ↑ higher is better; ↓ lower is better.

| | **GPT-5.4** | | **Qwen3.5-9B** | | | | **Phi-4-mini** | | | | **Gemma-2-9B** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Standalone** | | **HintMR** | | **AdaHintMR** | | **HintMR** | | **AdaHintMR** | | **HintMR** | | **AdaHintMR** | |
| **Dataset** | **Acc.↑** | **¢↓** | **Acc.↑** | **¢↓** | **Acc.↑** | **¢↓** | **Acc.↑** | **¢↓** | **Acc.↑** | **¢↓** | **Acc.↑** | **¢↓** | **Acc.↑** | **¢↓** |
| AIME-2024 | 76.67 | 1.44 | **76.67** | 1.21 | **76.67** | **1.01** | **80.00** | 1.02 | 70.00 | **0.38** | **76.67** | 1.16 | 70.00 | **0.32** |
| AIME-2025 | 53.33 | 1.84 | 50.00 | 1.44 | **63.33** | **1.04** | **60.00** | 1.00 | **60.00** | **0.36** | 60.00 | 1.01 | **66.67** | **0.40** |
| AIME-2026 | 60.00 | 1.88 | **66.67** | 1.57 | **66.67** | **1.03** | 63.30 | 1.02 | **73.33** | **0.37** | 56.67 | 1.01 | **66.67** | **0.39** |
| AMC-2023 | 92.50 | 0.88 | **92.50** | 0.83 | **92.50** | **0.32** | **97.50** | 0.96 | 92.50 | **0.35** | 87.50 | 0.95 | **90.00** | **0.32** |
| MATH-500 | 84.40 | 0.50 | **88.80** | 0.27 | 86.40 | **0.20** | **87.80** | 0.82 | 81.40 | **0.26** | **87.60** | 0.82 | 84.40 | **0.26** |
| Minerva-Math | 68.02 | 0.47 | **66.18** | 0.31 | **66.18** | **0.28** | 57.40 | 0.90 | **58.10** | **0.31** | **62.50** | 0.89 | 57.70 | **0.32** |

although HintMR achieves higher accuracy on MATH-500. Overall, adaptive hinting consistently reduces frontier-model usage without uniformly maximizing accuracy.

Figure 3 illustrates this trade-off across all six benchmarks. AdaHintMR generally moves toward the more cost-efficient region while maintaining accuracy close to, matching, or exceeding HintMR. The effect is especially clear on AIME-2025 and AMC-2023, where substantial cost reductions are achieved without sacrificing accuracy.

The results with Phi-4-mini and Gemma-2-9B show that this behavior extends beyond Qwen3.5-9B. AdaHintMR consistently lowers API cost across both solver architectures, while the accuracy advantage between HintMR and AdaHintMR varies by benchmark. For example, AdaHintMR performs better on AIME-2026 with Phi-4-mini and on AIME-2025, AIME-2026, and AMC-2023 with Gemma-2-9B, whereas denser HintMR guidance remains beneficial on several other benchmarks. These results suggest that the optimal amount of guidance depends on the solver and problem distribution, while entropy gating provides a consistent mechanism for reducing frontier-model usage.

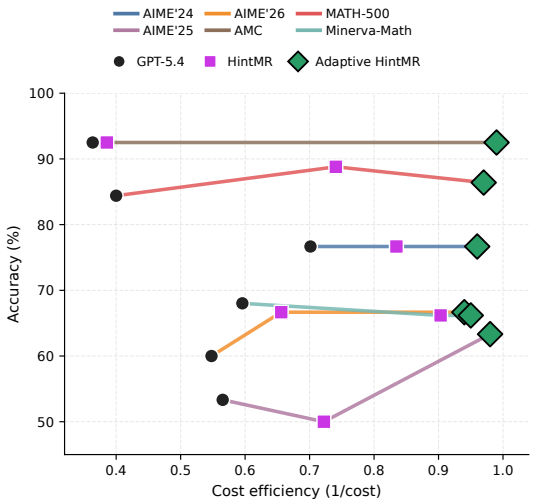


Figure 3: Accuracy–cost efficiency across six benchmarks. Cost efficiency is computed from the reciprocal of inference cost and normalized within each benchmark. Higher values of both accuracy and cost efficiency indicate better performance.

### 5.3 Comparison with Baselines

Finally, Table 3 compares AdaHintMR against four baselines: Self-Consistency Wang et al. (2022), TMATH Qi et al. (2025), HSP Fu et al. (2024), and Qwen3.8-27B Qwen Team (2026b) as a standalone solver without any hinter. Several observations emerge from this comparison.

First, additional inference-time sampling provides a competitive baseline on some benchmarks. Self-Consistency reaches 80.00% on AMC-2023 and 65.07% on Minerva-Math, but performs substantially worse on AIME, where accuracy ranges from 26.67% to 33.33%. In contrast, frontier-hinter versions of HintMR and AdaHintMR achieve much higher AIME accuracy, suggesting that targeted intermediate guidance provides benefits beyond generating multiple independent solutions. Increasing the standalone local solver size also does not reproduce these gains: Qwen3.8-27B reaches 33.33% on AIME-2025 and 36.67% on AIME-2026, whereas Qwen3.5-9B with AdaHintMR and GPT-5.4 reaches 63.33% and 66.67%, respectively.

Table 3: Accuracy comparison across six mathematical reasoning benchmarks. We compare HintMR and AdaHintMR against Self-Consistency, TMATH, HSP, and Qwen3.8-27B used as a standalone solver. HintMR and AdaHintMR are evaluated with Qwen3.5-9B, Phi-4-mini, and Gemma-2-9B as solvers. For Qwen3.5-9B, we consider both Qwen3.8-27B as a local hinter and GPT-5.4 as a frontier hinter, while Phi-4-mini and Gemma-2-9B use GPT-5.4 as the frontier hinter. Bold denotes the highest accuracy on each benchmark, including ties.

| **Method** | **AIME-2024** | **AIME-2025** | **AIME-2026** | **AMC-2023** | **MATH-500** | **Minerva-Math** |
|---|---|---|---|---|---|---|
| Self-Consistency | 33.33% | 26.67% | 30.00% | 80.00% | 82.00% | 65.07% |
| TMATH | 26.67% | 30.00% | 26.67% | 57.50% | 75.80% | 58.50% |
| HSP | 36.67% | 20.00% | 20.00% | 77.50% | 83.80% | 44.49% |
| Qwen3.8-27B | 26.67% | 33.33% | 36.67% | 80.00% | 80.40% | 64.39% |
| *Qwen3.5-9B solver + Qwen3.8-27B hinter* | | | | | | |
| HintMR | 40.00% | 26.67% | 30.00% | 60.00% | 79.20% | 58.09% |
| AdaHintMR | 40.00% | 30.00% | 40.00% | 60.00% | 80.60% | 58.09% |
| *Qwen3.5-9B solver + GPT-5.4 hinter* | | | | | | |
| HintMR | 76.67% | 50.00% | 66.67% | 92.50% | **88.80%** | **66.18%** |
| AdaHintMR | 76.67% | 63.33% | 66.67% | 92.50% | 86.40% | **66.18%** |
| *Phi-4-mini solver + GPT-5.4 hinter* | | | | | | |
| HintMR | **80.0%** | 60.0% | 63.3% | **97.5%** | 87.8% | 57.4% |
| AdaHintMR | 70.0% | 60.0% | **73.3%** | 92.5% | 81.4% | 58.1% |
| *Gemma-2-9B solver + GPT-5.4 hinter* | | | | | | |
| HintMR | 76.67% | 60.0% | 56.7% | 87.5% | 87.6% | 62.5% |
| AdaHintMR | 70.0% | **66.7%** | 66.7% | 90.0% | 84.4% | 57.7% |

Performance also varies across solver architectures and benchmarks. Phi-4-mini with HintMR achieves the highest accuracy on AIME-2024 (80.00%) and AMC-2023 (97.50%), while Phi-4-mini with AdaHintMR performs best on AIME-2026 (73.33%). Gemma-2-9B with AdaHintMR achieves the best AIME-2025 result at 66.67%. On MATH-500, Qwen3.5-9B with HintMR reaches 88.80%, while on Minerva-Math, Qwen3.5-9B with HintMR and AdaHintMR tie at 66.18%.

Overall, these results show that intermediate guidance can substantially strengthen SLM mathematical reasoning across architectures, while the relative benefit of dense versus adaptive hinting depends on the solver and benchmark. AdaHintMR additionally offers a more selective alternative that lowers frontier-model cost while often retaining or improving HintMR accuracy.

## 6 Conclusion

We introduced HintMR, a solver–hinter framework for improving mathematical reasoning in small language models through targeted intermediate guidance, together with AdaHintMR, which uses step-level uncertainty to determine when additional hints are needed. Rather than relying on a larger model to solve the entire problem, HintMR uses the hinter to provide concise, context-aware guidance while the SLM solver remains responsible for constructing the reasoning trajectory and final answer. Across six mathematical reasoning benchmarks, our HintMR and AdaHintMR configurations outperform the strongest results from Self-Consistency, TMATH, HSP, and Qwen3.8-27B on every benchmark. With GPT-5.4 as the frontier hinter, HintMR can also outperform GPT-5.4 used directly as a standalone solver by up to 6.67 percentage points. AdaHintMR further reduces reliance on the frontier model by selectively requesting additional guidance, lowering API cost by up to 72.4% relative to HintMR across the evaluated solver and benchmark settings while maintaining strong reasoning performance. These benefits are observed across Qwen3.5-9B, Phi-4-mini, and Gemma-2-9B, indicating that the framework can be applied to different SLM solvers rather than depending on a particular model.

Overall, our findings demonstrate that separating guidance from execution provides an effective and efficient way to strengthen mathematical reasoning in SLMs, allowing more capable models to assist selectively rather than performing the entire reasoning process themselves.

## AI USE STATEMENT

Generative AI tools were used during the preparation of this work to assist with language editing, improving clarity and organization, LaTeX formatting, code debugging, and drafting or refining experimental scripts, tables, and figures. All AI-assisted text, code, and artifacts were manually reviewed by the authors, and experimental code was executed and checked against the reported outputs before inclusion in the paper. Generative AI was not used to independently determine the paper's scientific conclusions or to fabricate experimental results, citations, or empirical claims. In addition, frontier language models are explicitly used as components of the proposed framework and evaluation, as described in the methodology and experimental setup. The authors take full responsibility for the final content of the paper, including all text, claims, code, results, and artifacts produced with the assistance of generative AI.

## ETHICS STATEMENT

This work studies how small language models can be supported with targeted intermediate guidance to improve mathematical reasoning while reducing unnecessary reliance on larger frontier models. All experiments use publicly available mathematical reasoning benchmarks and do not involve private, personally identifiable, or sensitive data. The main ethical risks arise from incorrect, incomplete, or misleading model-generated hints and reasoning steps, particularly in educational settings where users may place undue trust in generated explanations. Because the framework relies on existing solver and hinter models, it may also inherit their biases and failure modes. To mitigate these concerns, we evaluate across multiple benchmarks and model configurations, report both accuracy and inference cost, and explicitly treat entropy-based uncertainty as a heuristic for deciding when additional guidance may be useful rather than as a guarantee of mathematical correctness. Future work should further investigate verification, robustness, calibration, and responsible deployment of model-generated mathematical guidance.

## REPRODUCIBILITY STATEMENT

We highly recognize the importance of reproducibility in research. We provide a complete description of our experimental setup, including datasets, model configurations, training procedures, and evaluation protocols, to facilitate reproducibility. All hyperparameters and implementation details necessary to replicate our results are specified in the main text and appendix. We will release our code and relevant resources upon acceptance.

## REFERENCES

Alon Albalak, Duy Phung, Nathan Lile, Rafael Rafailov, Kanishk Gandhi, Louis Castricato, Anikait Singh, Chase Blagden, Violet Xiang, Dakota Mahan, and Nick Haber. Big-math: A large-scale, high-quality math dataset for reinforcement learning in language models. *arXiv preprint arXiv:2502.17387*, 2025.

Michelene TH Chi, Stephanie A Siler, Heisawn Jeong, Takashi Yamauchi, and Robert G Hausmann. Learning from human tutoring. *Cognitive science*, 25(4):471–533, 2001.

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.

Siqi Fan, Minghao Li, Xiaoqian Ma, Xiusheng Huang, Zhuo Chen, Bowen Qin, Liujie Zhang, Shuo Shang, and Weihang Chen. Hint tuning: Less data makes better reasoners. *arXiv preprint arXiv:2605.08665*, 2026.

Jinlan Fu, Shenzhen Huangfu, Hang Yan, See-Kiong Ng, and Xipeng Qiu. Hint-before-solving prompting: Guiding llms to effectively utilize encoded knowledge. *arXiv preprint arXiv:2402.14310*, 2024.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Zhengyang Tang, et al. Omni-math: A universal olympiad level mathematic benchmark for large language models. In *International Conference on Learning Representations*, volume 2025, pp. 100540–100569, 2025.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. Pal: Program-aided language models. In *International conference on machine learning*, pp. 10764–10799. PMLR, 2023.

Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and Mao Yang. rStar-math: Small LLMs can master math reasoning with self-evolved deep thinking. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 20640–20661. PMLR, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

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. *arXiv preprint arXiv:2103.03874*, 2021.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. Numinamath, 2024.

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. arXiv:2305.20050.

Zechun Liu, Changsheng Zhao, Forrest Iandola, Chen Lai, Yuandong Tian, Igor Fedorov, Yunyang Xiong, Ernie Chang, Yangyang Shi, Raghuraman Krishnamoorthi, et al. Mobilellm: Optimizing sub-billion parameter language models for on-device use cases. *arXiv preprint arXiv:2402.14905*, 2024.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, Yansong Tang, and Dongmei Zhang. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*, 2023.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, and Abhinav Rastogi. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*, 2024.

Thang Luong and Edward Lockhart. Advanced version of gemini with deep think officially achieves gold-medal standard at the international mathematical olympiad. Google DeepMind, July 2025. URL `https://deepmind.google/blog/advanced-version-of-gemini-with-deep-think-officially-achieves-gold-medal-standard-at-the-international-mathematical-olympiad/`. Published July 21, 2025.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*, 2023.

Sadegh Mahdavi, Muchen Li, Kaiwen Liu, Christos Thrampoulidis, Leonid Sigal, and Renjie Liao. Leveraging online olympiad-level math problems for llms training and contamination-resistant evaluation. *arXiv preprint arXiv:2501.14275*, 2025.

Math-AI. Amc23. https://huggingface.co/datasets/math-ai/amc23, 2023. Hugging Face dataset.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20286–20332, 2025.

OpenAI. Ten advances in mathematics and theoretical computer science. OpenAI, August 2026. URL https://openai.com/index/ten-advances-in-mathematics/. Published August 1, 2026.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

Changyong Qi, Yuang Wei, Haoxin Xu, Longwei Zheng, Peiji Chen, and Xiaoqing Gu. TMATH: A dataset for evaluating large language models in generating educational hints for math word problems. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 5082–5093, 2025.

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

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork, August 2026b. URL https://qwen.ai/blog?id=qwen3.8.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36:68539–68551, 2023.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*, 2023.

Siao Tang, Xinyin Ma, Gongfan Fang, and Xinchao Wang. Concisehint: Boosting efficient reasoning via continuous concise hints during generation. *arXiv preprint arXiv:2506.18810*, 2025.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Shubham Toshniwal, Ivan Moshkov, Sean Narenthiran, Daria Gitman, Fei Jia, and Igor Gitman. Openmathinstruct-1: A 1.8 million math instruction tuning dataset. *Advances in Neural Information Processing Systems*, 37:34737–34774, 2024.

Shubham Toshniwal, Wei Du, Ivan Moshkov, Branislav Kisacanin, Alexan Ayrapetyan, and Igor Gitman. Openmathinstruct-2: Accelerating ai for math with massive open-source instruction data. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2410.01560.

Chien Van Nguyen, Xuan Shen, Ryan Aponte, Yu Xia, Samyadeep Basu, Zhengmian Hu, Jian Chen, Mihir Parmar, Sasidhar Kunapuli, Joe Barrow, et al. A survey on small language models. In *Proceedings of the 15th International Conference on Recent Advances in Natural Language Processing-Natural Language Processing in the Generative AI Era*, pp. 807–821, 2025.

Ke Wang, Houxing Ren, Aojun Zhou, Zimu Lu, Sichun Luo, Weikang Shi, Renrui Zhang, Linqi Song, Mingjie Zhan, and Hongsheng Li. Mathcoder: Seamless code integration in llms for enhanced mathematical reasoning. *arXiv preprint arXiv:2310.03731*, 2023.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Haoran Xu, Baolin Peng, Hany Awadalla, Dongdong Chen, Yen-Chun Chen, Mei Gao, Young Jin Kim, Yunsheng Li, Liliang Ren, Yelong Shen, Shuohang Wang, Weijian Wu, Jianfeng Gao, and Weizhu Chen. Phi-4-mini-reasoning: Exploring the limits of small reasoning language models in math. *arXiv preprint arXiv:2504.21233*, 2025.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2309.12284.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. Mammoth: Building math generalist models through hybrid instruction tuning. *arXiv preprint arXiv:2309.05653*, 2023.

Xiang Yue, Tianyu Zheng, Ge Zhang, and Wenhu Chen. Mammoth2: Scaling instructions from the web. *Advances in Neural Information Processing Systems*, 37:90629–90660, 2024.

Yating Zhang, Wei Dong, Jiaxin Liu, Shangwen Wang, Deze Wang, Tiecheng Ma, Yiwei Li, and Kang Yang. A little help goes a long way: Tutoring llms in solving competitive programming through hints. *IEEE Transactions on Software Engineering*, 2025.

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2024. `https://huggingface.co/datasets/math-ai/aime24`, 2024.

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2025. `https://huggingface.co/datasets/math-ai/aime25`, 2025.

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2026. `https://huggingface.co/datasets/math-ai/aime26`, 2026.

Kai Zhao, Yanjun Zhao, Jiaming Song, Shien He, Lusheng Zhang, Qiang Zhang, and Tianjiao Li. Saber: Switchable and balanced training for efficient llm reasoning. In *AAAI Conference on Artificial Intelligence*, pp. 34950–34958, 2026.

Chuanyang Zheng, Zhengying Liu, Enze Xie, Zhenguo Li, and Yu Li. Progressive-hint prompting improves reasoning in large language models. *arXiv preprint arXiv:2304.09797*, 2023.

Fan Zhou, Zengzhi Wang, Nikhil Ranjan, Zhoujun Cheng, Liping Tang, Guowei He, Zhengzhong Liu, and Eric P Xing. Megamath: Pushing the limits of open math corpora. *arXiv preprint arXiv:2504.02807*, 2025.

# A Related Work

**Data and training for mathematical reasoning.** A central driver of progress in mathematical reasoning has been the construction of large-scale corpora containing mathematical problems paired with detailed reasoning traces. MetaMath (Yu et al., 2024) constructs MetaMathQA by rewriting seed problems from multiple perspectives, demonstrating that diversifying problem formulations can improve downstream mathematical reasoning. MathInstruct (Yue et al., 2024) similarly aggregates step-by-step mathematical solutions designed for chain-of-thought fine-tuning. OpenMathInstruct (Toshniwal et al., 2024; 2025) scales this direction to millions of problem–solution pairs and studies how data quality, diversity, and teacher strength affect reasoning performance. Competition-oriented collections such as NuminaMath (Li et al., 2024) provide additional supervision for challenging multi-step mathematical problems.

Recent work has continued to increase both the scale and diversity of mathematical training data. MegaMath (Zhou et al., 2025) constructs a large open mathematical corpus by combining mathematics-focused web documents, code, and synthetic data. AoPS-Instruct (Mahdavi et al., 2025) instead leverages problems and community solutions from the Art of Problem Solving forum to construct large-scale Olympiad-level supervision. The same work introduces LiveAoPSBench, which draws from newer problems to reduce the risk that evaluation performance is driven by contamination from pretraining data. These efforts highlight the continuing importance of high-quality supervision for improving mathematical reasoning.

Other work focuses on the structure and reliability of synthetic supervision. Big-Math (Albalak et al., 2025) emphasizes verifiable mathematical data, while MathCoder (Wang et al., 2023) interleaves natural-language reasoning with executable code. TMATH (Qi et al., 2025) is especially relevant to our work because it explicitly represents mathematical problems together with pedagogical hints. Whereas these approaches primarily improve a model through training data or fine-tuning, HintMR studies a different question: whether a solver can be assisted *during inference* by context-aware hints generated from its current reasoning state.

**Hint-guided mathematical reasoning.** A growing body of work suggests that mathematical reasoning can benefit from guidance that intervenes before or during the reasoning process. Progressive-Hint Prompting (Zheng et al., 2023) uses previous attempts as progressively stronger hints in a multi-turn interaction, while Hint-before-Solving (Fu et al., 2024) first elicits high-level guidance before asking the model to construct a complete solution. These methods establish that exposing a model to intermediate guidance can improve reasoning compared with fully independent generation.

Recent work has moved toward more adaptive forms of guidance. ConciseHint (Tang et al., 2025) intervenes during generation with concise textual hints and adjusts their intensity according to problem complexity, with the goal of reducing excessively long reasoning while preserving performance. Hint Tuning (Fan et al., 2026) approaches the problem from a training perspective: it constructs examples with no hints, sparse hints, or complete reasoning and teaches a reasoning model to calibrate its reasoning depth according to difficulty. These studies reinforce the idea that guidance need not be applied uniformly to every reasoning problem or reasoning step.

HintMR differs from these approaches in both the source and timing of the guidance. We explicitly separate reasoning into two model roles: a *solver*, which remains responsible for constructing the solution, and a *hinter*, which observes the problem together with the solver's accumulated reasoning and provides localized guidance for the next stage. Hints are therefore generated dynamically from the solver's current state rather than being a fixed prefix or an internal control signal. AdaHintMR further makes this interaction selective by estimating the solver's step-level uncertainty and requesting a new hint only when the entropy exceeds a predefined threshold. Importantly, this mechanism operates entirely at inference time and does not require training or fine-tuning the hinter for uncertainty-aware invocation.

Related ideas also appear in iterative feedback and refinement. Self-Refine (Madaan et al., 2023) and Reflexion (Shinn et al., 2023) use natural-language feedback to iteratively improve model outputs, while process supervision (Lightman et al., 2024; Luo et al., 2024) provides signals at intermediate stages rather than only evaluating the final answer. These approaches share the broader motivation of acting on intermediate reasoning states. HintMR, however, does not require an explicit

verifier, reward model, or critique-and-rewrite cycle; instead, a separate model provides a concise instructional hint that the solver incorporates into its continuing reasoning trajectory.

A similar separation between guidance and execution has been explored outside mathematical reasoning. For example, TeaCH (Zhang et al., 2025) uses a specialized hint generator to support competitive-programming problem solving. This provides further evidence that separating the generation of useful guidance from the generation of a complete solution can be beneficial. Our work develops this idea specifically for step-level mathematical reasoning and studies both local open-weight and frontier-model hinters.

**Improving reasoning in small language models.** The performance gap between frontier models and smaller open models has motivated substantial work on improving the reasoning capabilities of SLMs. DeepSeekMath (Shao et al., 2024) combines mathematics-focused continued pretraining with post-training for mathematical reasoning, while Qwen2.5-Math (Yang et al., 2024) develops specialized mathematical models at several parameter scales. Earlier instruction-tuned models such as WizardMath (Luo et al., 2023) and MAmmoTH (Yue et al., 2023; 2024) likewise show that carefully designed mathematical supervision can substantially improve smaller models.

More recent reasoning models push this direction further. DeepSeek-R1 (Guo et al., 2025) demonstrates that reasoning behavior learned by a large model can be transferred to substantially smaller Qwen- and Llama-based models through distillation. Phi-4-mini-reasoning (Xu et al., 2025) develops a dedicated multi-stage training recipe for a 3.8B-parameter model using long chain-of-thought data, preference optimization, and reinforcement learning. rStar-Math (Guan et al., 2025) takes a different approach, combining an SLM policy with Monte Carlo Tree Search and a process preference model to enable deeper test-time exploration. Together, these results show that model size alone does not determine mathematical reasoning ability and that suitable training or inference mechanisms can substantially strengthen smaller models.

Another recent line of work focuses on controlling the amount of computation used during reasoning. s1 (Muennighoff et al., 2025) introduces budget forcing, which explicitly extends or terminates a model's reasoning trajectory to study test-time scaling. SABER (Zhao et al., 2026) trains models to operate under multiple reasoning budgets, allowing inference to switch between shorter and deeper reasoning modes. Hint Tuning (Fan et al., 2026) similarly targets reasoning efficiency by learning when limited or extensive guidance is appropriate. These methods highlight an emerging shift from simply maximizing reasoning length toward allocating reasoning computation more selectively.

HintMR approaches efficiency from a different direction. Rather than increasing the number of sampled solutions, expanding search, or retraining the solver to operate under a predefined reasoning budget, we introduce a second model that provides targeted information during the solver's trajectory. AdaHintMR then uses uncertainty to decide when this external assistance should be invoked. This formulation allows the solver to reason independently on confident steps while reserving additional model computation for points at which intermediate guidance is more likely to be useful.

# B Additional Experimental Setup Details

## B.1 Entropy Threshold Selection

The entropy threshold $\tau$ determines how frequently AdaHintMR invokes the hinter. Since a hint is requested when the step-level uncertainty exceeds $\tau$, lower threshold values result in more frequent interventions, whereas higher values allow the solver to reason independently for more steps. We select $\tau$ using a held-out calibration set of 50 problems from Omni-MATH (Gao et al., 2025), which is separate from the six benchmarks used for final evaluation. We evaluate candidate thresholds $\tau \in \{0.05, 0.1, 0.3, 0.5, 0.8\}$ based on both reasoning accuracy and average hinter usage. Among these values, $\tau = 0.1$ provides the best overall trade-off between reasoning performance and hinter usage on the calibration set. We therefore fix $\tau = 0.1$ before final evaluation and use the same threshold for all reported AdaHintMR experiments without benchmark-specific tuning.

Table 4: Step-level entropy statistics for 2,590 reasoning steps generated by AdaHintMR across all six benchmarks. Steps are grouped by hint invocation and by the final outcome of the corresponding reasoning trajectory.

| Step Category | Mean Entropy | Count |
|---|---|---|
| Step from solved trajectory, hint triggered | 0.161 | 189 |
| Step from solved trajectory, no hint triggered | 0.068 | 1,268 |
| Step from unsolved trajectory, hint triggered | 0.199 | 146 |
| Step from unsolved trajectory, no hint triggered | 0.072 | 987 |

### B.2 Analysis of the Entropy Gate

To examine how step-level entropy relates to selective hint invocation and reasoning outcomes, we analyze 2,590 reasoning steps generated by AdaHintMR across all six benchmarks. Each step is grouped according to whether the entropy gate triggered a hint and whether it belongs to a trajectory that ultimately produced a correct final answer. Table 4 summarizes the resulting statistics.

As expected from the threshold-based gate, non-triggered steps have substantially lower entropy than triggered steps. Among steps from trajectories that ultimately solve the problem, the mean entropy is 0.068 when no hint is triggered and 0.161 when a hint is triggered. A similar pattern appears in unsuccessful trajectories, where the corresponding mean entropies are 0.072 and 0.199. The small difference between the two non-triggered groups indicates that low step-level entropy alone does not determine the final outcome of a trajectory. In contrast, among triggered steps, the higher mean entropy in unsuccessful trajectories suggests that greater predictive uncertainty is associated with more difficult reasoning states.

Overall, these results support entropy as a lightweight signal for selective hint invocation while also highlighting its limitations. Entropy reflects local predictive uncertainty rather than mathematical correctness: low-entropy steps can occur in trajectories that ultimately fail, while high-entropy steps can still occur in trajectories that recover and reach the correct answer. Accordingly, AdaHintMR uses entropy to decide when additional guidance may be useful rather than treating it as a correctness verifier.

### B.3 Benchmark Details

**AIME.** We evaluate on the 2024, 2025, and 2026 editions of the American Invitational Mathematics Examination (AIME). AIME problems typically require multi-step reasoning across algebra, geometry, combinatorics, and number theory. Their difficulty and relatively long solution trajectories make them well suited for studying whether intermediate guidance can help SLMs on challenging reasoning problems. Table 5 summarizes the evaluation benchmarks in terms of the number of problems, general difficulty, benchmark type, and mathematical coverage.

**AMC-2023.** The AMC-2023 benchmark contains problems from the 2023 American Mathematics Competitions. Compared with AIME, these problems are generally shorter while still requiring non-trivial mathematical reasoning, providing a useful setting for evaluating hint-assisted reasoning when less intervention may be needed.

**MATH-500.** MATH-500 contains 500 mathematical reasoning problems spanning multiple domains and difficulty levels. We use it to evaluate whether the benefits of solver–hinter collaboration extend beyond recent competition-style benchmarks.

**Minerva-Math.** Minerva-Math contains challenging quantitative reasoning problems from a diverse set of mathematical topics. It provides a different problem distribution and allows us to further evaluate the robustness of the framework.

### B.4 Computational Environment

All experiments were conducted on a GPU computing cluster using NVIDIA A100 GPUs, each equipped with 80 GB of GPU memory. The environment provided approximately 1 TB of stor-

Table 5: Summary of the mathematical reasoning benchmarks used in our evaluation. Size denotes the number of problems evaluated in our experiments.

| Benchmark | Size | Difficulty | Type | Primary Coverage |
|---|---|---|---|---|
| AIME-2024 | 30 | High | Competition | Algebra, geometry, combinatorics, number theory |
| AIME-2025 | 30 | High | Competition | Multi-step competition mathematics across diverse topics |
| AIME-2026 | 30 | High | Competition | Multi-step competition mathematics across diverse topics |
| AMC-2023 | 40 | Moderate | Competition | Shorter competition problems across algebra, geometry, and counting |
| MATH-500 | 500 | Mixed | General | Algebra, geometry, calculus, probability, and number theory |
| Minerva-Math | 272 | High | General | Diverse quantitative and advanced mathematical reasoning |

age, enabling efficient model inference and consistent experimental conditions across all evaluated datasets.

### B.5 Baseline and Variant Details

**No Hint.** The solver follows the same stepwise generation protocol as HintMR but receives no external guidance. We evaluate both Qwen3.5-9B and Qwen3.8-27B to separate the effect of intermediate hints from the stepwise reasoning protocol and from gains due solely to increased model size.

**GPT-5.4 Direct Solver.** GPT-5.4 is used directly as a standalone solver, providing a frontier-model reference for both accuracy and API cost.

**Self-Consistency.** (Wang et al., 2022) Multiple independent solutions are generated with Qwen3.5-9B, and the final answer is selected by majority vote. This baseline measures the benefit of additional inference-time sampling without a separate hinter.

**HSP.** (Fu et al., 2024) Hint-before-Solving Prompting provides a single hint before reasoning begins, allowing comparison between one-time guidance and HintMR's iterative, context-dependent hints.

**TMATH.** (Qi et al., 2025) TMATH is included as an additional hint-based baseline using learned hint generation for mathematical reasoning.

**Entropy-Gated HintMR.** The solver begins without an initial hint, and the hinter is invoked only when step-level entropy exceeds the predefined threshold.

**Roadmap-Guided HintMR.** The hinter first provides a concise solution roadmap, after which additional hints are triggered using the same entropy-based criterion.

### B.6 Prompt Templates

We provide the complete prompt templates below. Variables enclosed in braces, such as {`problem`} and {`current step`}, are replaced at inference time with the corresponding problem-specific content.

#### B.6.1 No-Hint Prompt

In the no-hint setting, the solver receives only the problem statement and generates the complete reasoning trajectory without external guidance.

Solver Prompt

**System:**

```
You are an expert math problem solver.  Solve the problem
step by step.  Put your final answer in \boxed{}.
```

**User:**

```
Problem:  {problem}
Solve step by step:
```

### B.6.2 HINTMR PROMPTS

HintMR uses two prompt templates: one for the *hinter* and one for the *solver*. At each reasoning step, the hinter receives the problem together with the solver's current reasoning state and generates a concise, context-aware hint. The solver then conditions on this hint and its accumulated reasoning to produce the next step.

Hinter Prompt

**System:**

```
You are a math tutor.  Based on the reasoning generated
so far and the problem context, provide one concise
instructional hint that guides the solver toward the next
useful reasoning step.  Focus only on what the solver should
consider next.  Do not reveal the final answer or provide the
complete solution.
```

**User:**

```
Problem:  {problem}
Reasoning so far:  {reasoning context}
```

Solver Prompt

**System:**

```
You are an expert mathematical reasoning assistant.  Solve
the problem step by step.  Generate exactly one reasoning
step at a time.  Use the provided hint to guide your next
step, while carrying out the reasoning yourself.  End with
\boxed{} only when you have reached the final answer.
```

**User:**

```
Problem:  {problem}
Hint:  {hint}
Reasoning so far:  {accumulated reasoning}
Next step:
```

### B.6.3 ADAPTIVE HINTMR PROMPTS

Adaptive HintMR uses the same solver and hinter prompt structure as HintMR, but invokes the hinter selectively according to the entropy-based gate. After each nonterminal solver step, the step-level entropy is computed. If it exceeds the threshold $\tau$, the hinter receives the original problem together with the solver's accumulated reasoning trajectory and generates a concise, context-aware hint for the next reasoning step. Otherwise, no new hint is provided and the solver continues using only the problem and accumulated reasoning.

Table 6: Hyperparameters used for Adaptive HintMR inference.

| Hyperparameter | Value |
|---|---|
| Top-$K$ probabilities for entropy | 20 |
| Entropy threshold $\tau$ | 0.1 |
| Maximum hinter response length | 80 tokens |
| Maximum solver generation budget | 16,000 tokens |
| Random seed | 42 |
| Entropy aggregation | Mean token entropy |
| Hinter input | Problem + reasoning context |

**Hinter Prompt**

**Invocation condition:** $\bar{\mathcal{H}} > \tau$

**System:**

```
You are a math tutor. Based on the reasoning generated
so far and the problem context, provide one concise
instructional hint that guides the solver toward the next
useful reasoning step. Focus only on what the solver should
consider next. Do not reveal the final answer or provide the
complete solution.
```

**User:**

```
Problem: {problem}
Reasoning so far: {accumulated reasoning}
```

**Solver Prompt**

**System:**

```
You are an expert mathematical reasoning assistant. Solve
the problem step by step. Generate exactly one reasoning
step at a time. Use the provided hint, when available,
to guide your next step while carrying out the reasoning
yourself. End with \boxed{} only when you have reached the
final answer.
```

**User:**

```
Problem: {problem}
Hint: {hint}
Reasoning so far: {accumulated reasoning}
Next step:
```

Table 6 summarizes the inference and uncertainty-estimation settings used for AdaHintMR, including the entropy threshold, generation limits, and maximum hint length.

# C Additional Experimental Results

## C.1 Solver Token Efficiency

Table 7 reports the average number of tokens generated by the solver under HintMR and AdaHintMR. Across most solver–hinter settings, AdaHintMR reduces solver-side generation, although the magnitude of the reduction varies across benchmarks.

With Qwen3.5-9B as the solver and Qwen3.8-27B as the local hinter, AdaHintMR uses fewer solver tokens on five of six benchmarks. The largest reduction occurs on AMC-2023, decreasing from 8,510 to 4,939 tokens per problem (42.0%). Reductions are also observed on AIME-2024 (12,848

Table 7: Average solver tokens generated per problem under HintMR and AdaHintMR. Counts include solver generation only and exclude tokens generated by the hinter. Lower values indicate lower solver-side generation cost.

(a) Qwen3.5-9B as Solver

| | **Qwen3.8-27B Hinter** | | **GPT-5.4 Hinter** | |
|---|---|---|---|---|
| **Dataset** | **HintMR** | **AdaHintMR** | **HintMR** | **AdaHintMR** |
| AIME-2024 | 12,848 | **11,152** | 8,642 | **6,343** |
| AIME-2025 | 12,427 | **9,618** | 11,968 | **8,178** |
| AIME-2026 | 11,711 | **8,755** | 11,176 | **8,755** |
| AMC-2023 | 8,510 | **4,939** | 3,983 | **2,783** |
| MATH-500 | 5,934 | **4,243** | **1,956** | 2,016 |
| Minerva-Math | **1,203** | 1,267 | **1,307** | 1,401 |

(b) Phi-4-mini and Gemma-2-9B as Solvers with GPT-5.4 Hinter

| | **Phi-4-mini** | | **Gemma-2-9B** | |
|---|---|---|---|---|
| **Dataset** | **HintMR** | **AdaHintMR** | **HintMR** | **AdaHintMR** |
| AIME-2024 | 8,168 | **8,050** | 7,993 | **7,728** |
| AIME-2025 | 10,659 | **7,756** | 10,842 | **8,763** |
| AIME-2026 | 10,438 | **8,111** | 9,996 | **9,699** |
| AMC-2023 | **3,140** | 3,450 | 3,956 | **2,602** |
| MATH-500 | 1,951 | **1,594** | 1,931 | **1,591** |
| Minerva-Math | 1,285 | **1,186** | 1,409 | **1,350** |

Bold denotes the lower solver token count between HintMR and AdaHintMR within each solver–hinter setting.

to 11,152), AIME-2025 (12,427 to 9,618), AIME-2026 (11,711 to 8,755), and MATH-500 (5,934 to 4,243). Minerva-Math is the only exception, increasing slightly from 1,203 to 1,267 tokens.

The same overall trend holds with GPT-5.4 as the hinter. For Qwen3.5-9B, AdaHintMR reduces solver generation on AIME-2024, AIME-2025, AIME-2026, and AMC-2023, while producing slightly more tokens on MATH-500 and Minerva-Math. Across the additional solvers, Phi-4-mini uses fewer tokens with AdaHintMR on five of six benchmarks, whereas Gemma-2-9B uses fewer tokens on all six. Overall, adaptive hinting often shortens the solver's reasoning trajectory, although this benefit is dependent on the solver and benchmark.

# D Qualitative Example Analysis

## D.1 HintMR and AdaHintMR

Figures 4–7 provide a qualitative example of how HintMR changes the solver's reasoning trajectory under different hinting settings.

Without hints, the solver produces several locally plausible reasoning steps but fails to incorporate the broader structure required to reach the correct solution. As a result, the reasoning gradually drifts and terminates with an incorrect answer.

With HintMR, the solver receives localized guidance throughout the reasoning process. These hints help redirect the solver toward the relevant algebraic and structural relationships while leaving the solver responsible for carrying out the derivation itself. As the reasoning progresses, the trajectory becomes more focused and coherent, allowing the solver to recover from intermediate mistakes and eventually reach the correct answer.

The example also illustrates that useful guidance can be provided by both local and frontier-model hinters. The local SLM hinter is able to steer the solver toward a productive reasoning path, while the

frontier-model hinter provides more explicit intermediate guidance. Overall, the example highlights the main effect of HintMR: targeted intermediate hints can keep the solver on a productive trajectory and improve the quality of multi-step mathematical reasoning without replacing the solver's own reasoning process.

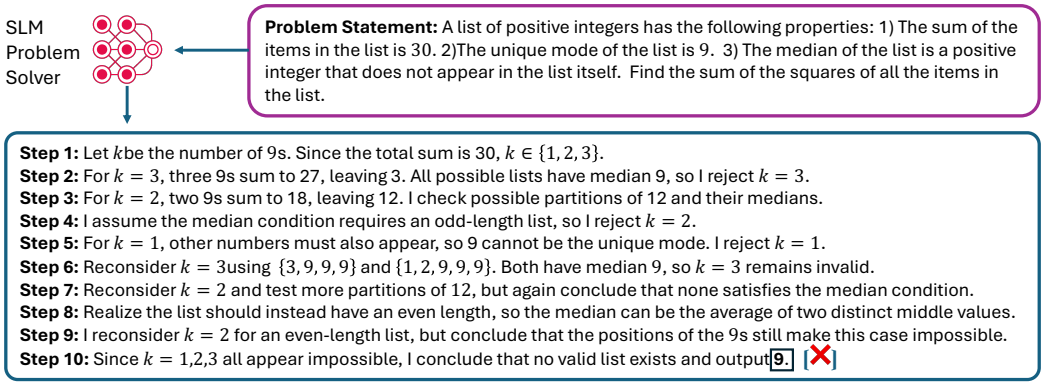


Figure 4: Qualitative reasoning trajectory of Qwen3.5-9B without hint assistance. The solver initially makes an incorrect assumption about the list length and, although it later recognizes that the list should have even length, fails to recover the correct solution and produces an incorrect final answer.

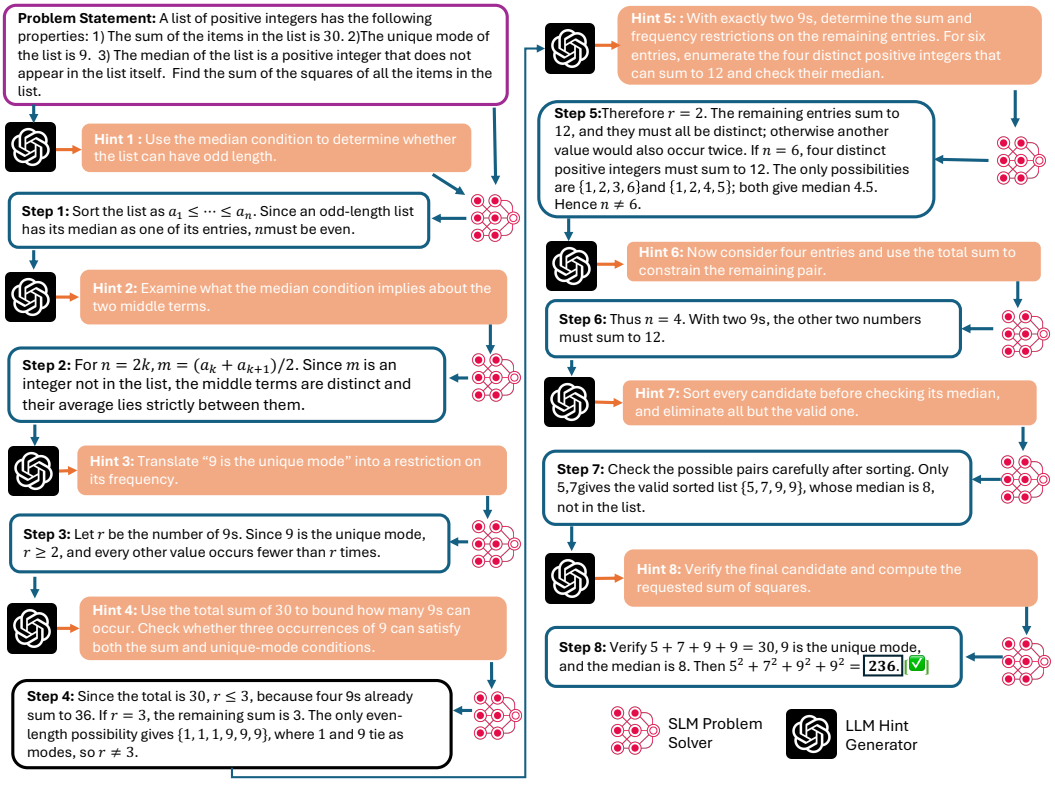


Figure 5: Qualitative reasoning trajectory of Qwen3.5-9B with HintMR using GPT-5.4 as the frontier hinter. Step-wise hints guide the solver through the median, mode-frequency, and candidate-selection constraints, leading to the valid list $\{5, 7, 9, 9\}$ and the correct final answer of 236.

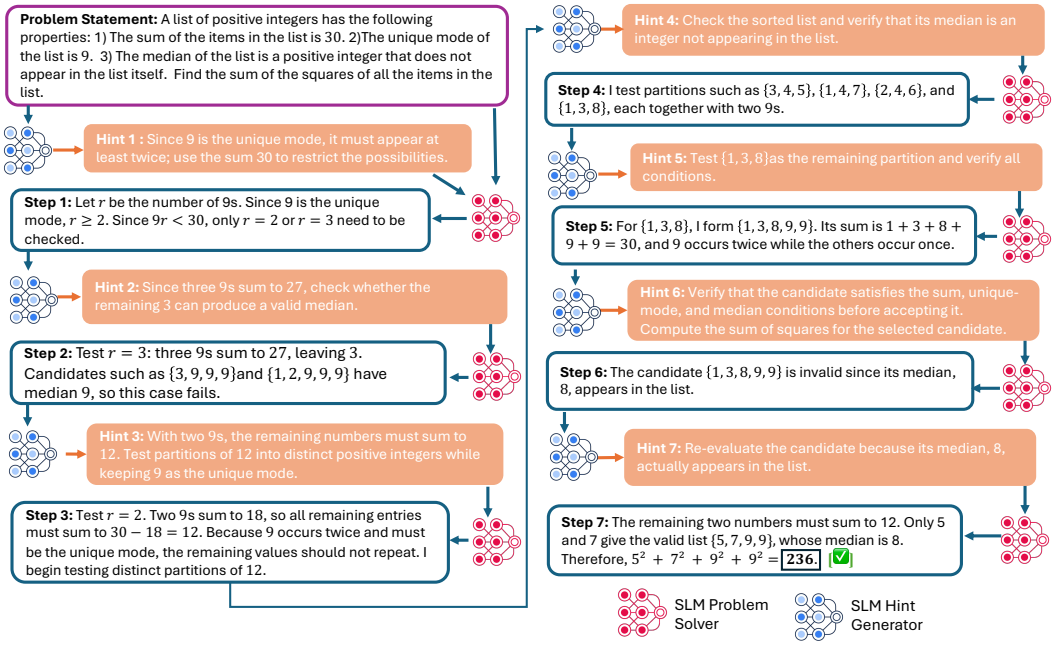


Figure 6: Qualitative reasoning trajectory of Qwen3.5-9B with HintMR using a local SLM hinter. The solver initially follows an invalid candidate, but subsequent hint feedback identifies the violated median condition and redirects the reasoning toward the valid list $\{5, 7, 9, 9\}$. The solver then verifies the constraints and reaches the correct final answer of 236.

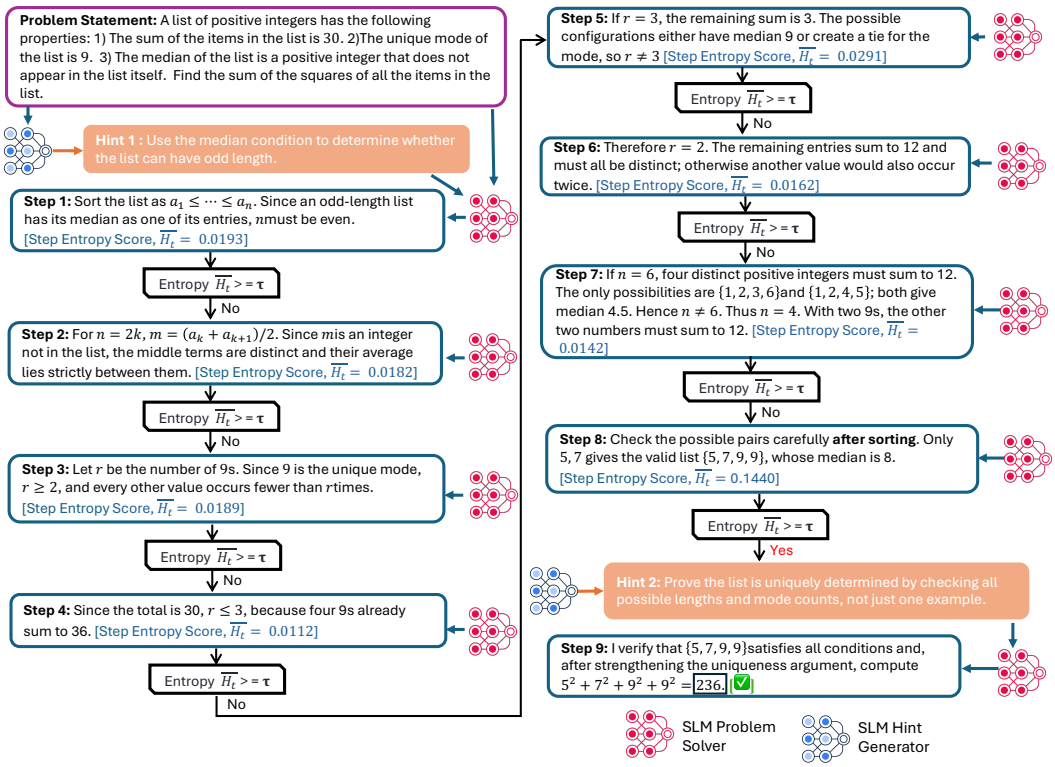


Figure 7: Qualitative reasoning trajectory of Qwen3.5-9B with AdaHintMR using a local hinter. The solver proceeds independently through low-uncertainty steps and selectively receives additional guidance when the step-level entropy exceeds the threshold $\tau = 0.1$. A later hint encourages verification of uniqueness, after which the solver confirms the valid list $\{5, 7, 9, 9\}$ and obtains the correct answer of 236.