ReasonMap: Making Implicit Mathematical Knowledge Explicit in Long Chain-of-Thought Traces

Anonymous ACL submission

Abstract

Long chain-of-thought (CoT) traces provide observable natural-language rationales for mathematical problem solving, but they rarely expose the named definitions and theorems that justify the written solution. We treat each problem–trace pair as carrying a recoverable knowledge footprint: a trace-grounded support set of named definitions and theorems. Inspired by formal proofs, where supporting premises are recorded more explicitly than in free-form CoT, we formulate Mathematical Support Set Extraction (MSSE): given a problem and a completed long CoT trace, recover the canonical support set that supports the written solution. We propose ReasonMap, which attributes this support set to make solutions auditable, reusable, and easier to repair. ReasonMap follows a two-stage pipeline: proof-to-CoT support transfer adapts premise-explicit proof supervision to premise-implicit long CoT traces, and hard-negative support calibration applies preference optimization against plausible-but-wrong support sets. Experiments show that ReasonMap improves support-set extraction over strong baselines and that ReasonMap-generated support hints aid downstream reasoning in both frontier-model and self-generated settings. The GitHub repository is available at https://anonymous.4open.science/r/ReasonMap-6A83.

1 Introduction

Long chain-of-thought (CoT) reasoning has become central to mathematical problem solving in large language models (Wei et al., 2022; Snell et al., 2025). Although such traces expose intermediate reasoning, they remain knowledge-opaque: they rarely identify the named definitions and theorems that license a solution's steps. A completed trace can therefore be inspected verbally while its mathematical support remains difficult to diagnose, reuse, or audit. We call this trace-level knowledge footprint a Mathematical Support Set (MSS): the set of named definitions and theorems invoked by, or required to justify, a completed solution.

Existing resources do not directly provide this information. Large-scale mathematical datasets typically offer only coarse metadata, such as domain labels or difficulty estimates (He et al., 2025); these labels identify a problem's surface topic but not whether a particular solution uses induction, recurrence solving, polynomial-degree arguments, or other concrete tools. Formal mathematics provides complementary supervision: in proof systems and premise-grounded proof datasets, supporting premises are often recorded explicitly (Murphy et al., 2024; He et al., 2024; Cao et al., 2025; Welleck et al., 2022). However, these resources target short, structured proofs or formal proof environments, rather than long, informal CoT traces. Thus, neither coarse metadata nor existing proof-oriented supervision directly attributes fine-grained mathematical support to free-form reasoning traces. Additional motivating examples and dataset-level evidence appear in Appendix A.4.

We therefore formulate Mathematical Support Set Extraction (MSSE): given a problem and a completed reasoning trace, recover the canonical set of named mathematical definitions and theorems supporting the written solution. The goal is not to assign a broad problem-level topic tag, but to recover the specific mathematical knowledge used by that trace.

MSSE poses two core challenges. The first is supervision mismatch: MSS are explicit mainly in short proof-style examples, while definition entries provide auxiliary concept-level supervision; target long CoT traces usually leave the same items implicit and distributed across many steps. The second is extraction fidelity: a predicted support set may be plausible yet wrong by adding extra items, omitting necessary ones, duplicating repeated items, or selecting a nearby theorem at the wrong specificity. Such near misses are especially

harmful because they can appear useful while misleading downstream reasoning.

To address these challenges, we propose ReasonMap, a trace-to-support framework for MSSE. ReasonMap learns a canonical MSS vocabulary and its usage from premise-explicit proof supervision, transfers this knowledge to premise-implicit long CoT traces, and uses Direct Preference Optimization (DPO) (Rafailov et al., 2023) with hard negatives to suppress plausible but mathematically incorrect support sets. On in-distribution (ID) benchmarks, ReasonMap improves fine-grained MSS extraction; on out-of-distribution (OOD) benchmarks, it shows domain-relevance transfer under a proxy metric. Its recovered MSS also improve downstream problem solving when used as ReasonMap-generated hints. Our contributions are threefold:

- We formulate Mathematical Support Set Extraction (MSSE), a trace-level attribution task that recovers the named definitions and theorems supporting a completed mathematical reasoning trace for a given problem.
- We propose ReasonMap, a trace-to-support framework that bridges premise-explicit proof supervision to premise-implicit long CoT traces through proof-to-CoT support transfer and hard-negative support calibration.
- We show that extracted mathematical support sets are actionable: they improve fine-grained extraction quality on ID benchmarks, show domain-relevance transfer on OOD benchmarks, and serve as reusable support sheets for downstream mathematical reasoning.

2 Related Work

2.1 CoT and Reasoning Trace Analysis

Chain-of-thought (CoT) prompting has become a standard paradigm for eliciting mathematical reasoning in large language models (Wei et al., 2022). Recent work analyzes the internal organization of reasoning traces, including shared decision pivots, graph-structured reasoning paths, and the contribution of individual thoughts to final answers (Cho et al., 2025; Zhang et al., 2025; Bogdan et al., 2025; Venhoff et al., 2025). These studies primarily analyze, steer, or restructure the reasoning process itself. For example, Ranaldi et al. (2025) introduce quasi-symbolic abstractions that reorganize CoT reasoning into more structured paths, while CoMAT improves performance and verifiability by converting natural-language queries into symbolic form before executing reasoning (Leang et al., 2025). In contrast, ReasonMap operates post hoc: it does not construct a new reasoning process, but attributes a completed long trace by recovering the trace-level MSS of named definitions and theorems that makes the solution auditable, reusable, and easier to repair.

2.2 Mathematical Knowledge in Proofs and Formalization

Mathematical proof research offers a complementary setting where supporting knowledge is more explicit. Natural-Proofs-Gen grounds proof generation on referenced theorems and definitions (Welleck et al., 2022), but works in the reverse direction: generating proofs from known premises rather than extracting knowledge from completed solutions. Autoformalization (Polu et al., 2023) and formal proof methods map informal statements into formal settings through proof planning, premise selection, and proof-state guidance (Murphy et al., 2024; He et al., 2024; Cao et al., 2025), while statement–proof matching learns associations between proofs and theorems (Li et al., 2023). These methods typically rely on formal verifiers, short structured proofs, or narrow domains, assumptions that do not hold for long, informal CoT traces. ReasonMap instead extracts the mathematical support set from a completed long CoT trace without requiring a formal proof environment.

2.3 Mathematical Support Knowledge in Data and Supervision

Mathematical datasets have expanded through both human curation and LLM-based construction. Human-curated resources such as GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), and Natural-Proofs-Gen (Welleck et al., 2022) provide high-quality supervision but are costly to scale, especially for long solutions. Larger resources such as Omni-Math (Gao et al., 2024), Light-R1 (Wen et al., 2025), DeepMath-103K (Chen et al., 2025), and related synthesis pipelines emphasize diversity, difficulty, or broad domain coverage. However, they mostly provide domain- or difficulty-level labels rather than trace-level MSS of named theorems and definitions. ReasonMap targets this missing layer by learning to extract mathematical support sets and validating their usefulness for downstream reasoning.

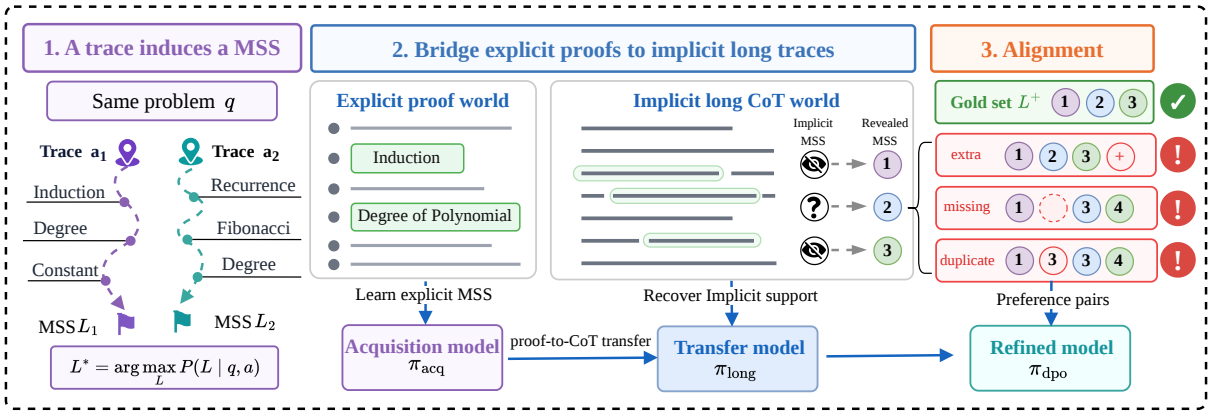

Figure 1: Overview of ReasonMap. (1) A problem–trace pair induces a trace-specific support set: different valid solution traces for the same problem may rely on different named definitions and theorems. (2) We perform proof-to-CoT support transfer, bridging premise-explicit proof data and premise-implicit long CoT traces through a sequential short-to-long curriculum. (3) We further conduct hard-negative support calibration, suppressing plausible-but-wrong support sets with DPO and yielding a faithful support sheet that can be reused as a reasoning hint.

3 Method

ReasonMap aims to recover the trace-grounded mathematical support set for a given problem–trace pair. This requires handling two challenges: transferring explicit support supervision from short proof-style data to long, premise-implicit CoT traces, and improving extraction fidelity when predicted support sets contain plausible but incorrect items. As shown in Figure 1, ReasonMap addresses these challenges with a two-stage framework: proof-to-CoT support transfer followed by hard-negative support calibration.

3.1 A Trace Induces a Support Set

Given a problem q and a completed reasoning trace a, Mathematical Support Set Extraction (MSSE) predicts a support set $L(q, a) \subseteq \mathcal{V}$, where $\mathcal{V} = \mathcal{V}_{\text{def}} \cup \mathcal{V}_{\text{thm}}$ is a canonical vocabulary of named definitions and theorems. Each vocabulary entry is a support item. The target is trace-grounded: $L(q, a)$ contains the MSS invoked by, or required to justify, the written trace, rather than the model's private internal state or a broad problem-level topic.

We train the extractor as a conditional generation model: the input is $x = (q, a)$, and the target output y is a deterministic canonical listing of the MSS $L(q, a)$. Following the definition-entry setup in Natural-Proofs-Gen (Welleck et al., 2022), we also use definition-only instances only as auxiliary vocabulary-acquisition supervision, not as trace-grounded MSSE examples:

- A problem–trace example $(q_i, a_i, L_i) \in \mathcal{D}_{\text{proof}}$, containing a problem, its completed reasoning trace, and the trace-grounded support set.
- An auxiliary definition-only example $(c_j, L_j^{\text{def}}) \in \mathcal{D}_{\text{def}}$, whose target contains the defined canonical title L_j and linked description c_j.

For problem–trace inputs, $L = \{d_1, \ldots, d_G, t_1, \ldots, t_M\}$, where d_g and t_m denote the g-th definition and m-th theorem, respectively.

3.2 From Explicit Proofs to Long Traces

The central difficulty in MSSE is a supervision mismatch. The data source where MSS are explicit consists of short, structured proof-style examples, while the inputs we ultimately care about are long CoT traces whose MSS are implicit. Training only on the former gives the model a clean support-item vocabulary but poor robustness to long, redundant reasoning.

Training only on generated long CoTs would expose the model to the target style but risks path-label misalignment, because a generated solution may solve the same problem through a different mathematical route from the original referenced proof. ReasonMap resolves this mismatch with a two-step curriculum. The first step is knowledge acquisition: we train on premise-explicit examples together with definition-only instances, so that the model learns the canonical MSS space and the semantics of individual mathematical items. The second step is trace adaptation: we convert selected short proof examples into long CoT traces while

preserving their original MSS. This teaches the model to recover the same underlying MSS under the paraphrase, redundancy, and delayed premise usage typical of long CoT reasoning.

We source our dataset from Natural-Proofs-Gen (Welleck et al., 2022), a high-quality dataset of human-written mathematical proofs. Each problem–trace entry in this source data provides a math problem q, a concise proof-style solution a, and corresponding referenced support items L from ProofWiki. To suit MSSE, we process and reconstruct these entries to create $\mathcal{D}_{\text{acq}} = \mathcal{D}_{\text{proof}} \cup \mathcal{D}_{\text{def}}$, where $\mathcal{D}_{\text{proof}}$ contains problem–trace MSSE examples and $\mathcal{D}_{\text{def}}$ contains auxiliary definition-only vocabulary examples. Examples can be found in Appendices A.10 and A.11.

To adapt this explicit supervision to informal long reasoning traces, we construct a generalization dataset designed specifically for proof-to-CoT support transfer. For each referenced triple (q, a, L), we prompt a strong reasoning model to expand the concise proof-style solution a into a long trace a^{ℓ} conditioned on both q and a. Conditioning on a is crucial: it anchors the generated trace to the referenced proof path, reducing the chance that the model solves the same problem using a different MSS. Specifically, we prompt the DeepSeek-R1 (DeepSeek-AI et al., 2025) to produce a^{ℓ}, following the instructions provided in Appendix A.3. The resulting triple (q, a^{ℓ}, L) forms $\mathcal{D}_{\text{long}}$. We further analyze the fidelity and expansion behavior of this augmentation process in Appendix D.

We utilize the two collected datasets for training and evaluation. First, we partition $\mathcal{D}_{\text{acq}}$ into training $\mathcal{D}_{\text{acq}}^{\text{train}}$ and test sets $\mathcal{D}_{\text{acq}}^{\text{test}}$, with $\mathcal{D}_{\text{proof}}^{\text{train}}$ and $\mathcal{D}_{\text{proof}}^{\text{test}}$ denoting their problem–trace portions. The generalization sets, $\mathcal{D}_{\text{long}}^{\text{train}}$ and $\mathcal{D}_{\text{long}}^{\text{test}}$, are curated by augmenting problem–trace subsets sampled from $\mathcal{D}_{\text{proof}}^{\text{train}}$ and $\mathcal{D}_{\text{proof}}^{\text{test}}$, respectively, thereby preventing data leakage. Our training protocol is a two-step curriculum process. We first warm up the base model on $\mathcal{D}_{\text{acq}}^{\text{train}}$, yielding π_{acq}. Subsequently, we continue training this model on the generalization data $\mathcal{D}_{\text{long}}^{\text{train}}$ to adapt its knowledge to complex long CoT reasoning.

The output of this transfer process is π_{long}, an extractor adapted to both concise proof-style solutions and informal long CoT traces.

3.3 Learning from Near Misses

After proof-to-CoT support transfer, π_{long} can still produce MSS from the correct mathematical neighborhood that are nevertheless flawed. The most harmful errors are near misses: an extra theorem that sounds relevant, a neighboring definition at the wrong granularity, or a duplicated item that inflates the MSS. Such outputs are difficult to eliminate with SFT alone, because they are fluent and mathematically plausible. We therefore cast high-fidelity extraction as a hard-negative support calibration problem, where the gold MSS is preferred over a self-generated MSS that is close to correct but still flawed. We use DPO for this refinement step because of its stability and efficiency compared with more complex RLHF pipelines (Yang et al., 2025a). Following Cheng et al. (2025), we construct **hard negatives** from candidates generated by π_{long} that remain plausible but contain fine-grained flaws.

We first construct a generation dataset $\mathcal{D}_{\text{gen}}$, which is composed of all samples from $\mathcal{D}_{\text{long}}^{\text{train}}$ combined with a filtered subset of $\mathcal{D}_{\text{proof}}^{\text{train}}$, excluding auxiliary definition-only instances. Note that samples from $\mathcal{D}_{\text{long}}^{\text{train}}$ make up a slightly larger proportion; the goal is to maintain a strong focus on long CoT capabilities while preserving the model's generalization to shorter cases. For $\mathcal{D}_{\text{gen}}$, we only include samples where the label length $|L| > 3$, considering that overly short labels are prone to being trivially copied by the model and fail to produce meaningful preference comparisons.

For each input $x = (q, a)$ in $\mathcal{D}_{\text{gen}}$, we use π_{long} to sample N candidate sequences $\{\hat{y}_1, \ldots, \hat{y}_N\}$ and parse each into a deduplicated MSS $\hat{L}_k$. We score each candidate using its deduplicated match rate m_k with the gold MSS and its repeat rate r_k:

$$\mathcal{S}_k = (1 - r_k) \cdot m_k. \tag{1}$$

This score prefers candidates that substantially overlap with the gold MSS while penalizing repeated MSS. The detailed calculation and distribution of repeat rate and exact-match precision are shown in Figure 4 in Appendix A.6.

To select a challenging but flawed negative example, we identify the candidate with the highest score inside a quality window $[\alpha, \gamma)$. Samples with no candidate in this window are discarded from DPO construction. The index of this "rejected" candidate k^* is thus determined by:

$$k^* = \arg\max_{k \in \{1,\ldots,N\}} \mathcal{S}_k, \quad \text{s.t. } \alpha \leq \mathcal{S}_k < \gamma. \tag{2}$$

Using this rule, we construct the preference dataset $\mathcal{D}_{\text{dpo}}$. For each tuple $(x, y^+, y^-) \in \mathcal{D}_{\text{dpo}}$, the chosen response y^+ is the canonical output sequence

of the gold support set, while the rejected response y^- is the candidate sequence $\hat{y}_{k^*}$ selected by (2). We then adopt the DPO objective. Let $\pi_{\text{ref}} = \pi_{\text{long}}$, $\Delta_\theta(y \mid x) = \log \pi_{\text{dpo}}(y \mid x) - \log \pi_{\text{ref}}(y \mid x)$, and $\Delta_\theta^+ = \Delta_\theta(y^+ \mid x)$, $\Delta_\theta^- = \Delta_\theta(y^- \mid x)$. The loss is:

$$\mathcal{L}_{\text{dpo}} = -\mathbb{E}_{\mathcal{D}_{\text{dpo}}} \left[\log \sigma\big(\beta(\Delta_\theta^+ - \Delta_\theta^-)\big) \right]. \quad (3)$$

Here, $\beta > 0$ controls the strength of the preference, and $\sigma(\cdot)$ converts the preference score into a probability. The reference model is a frozen copy of π_{long}. This calibration stage refines π_{long} by reducing subtle flaws such as hallucinations and format inconsistencies, yielding the final model π_{dpo}.

4 Experiments

We conduct experiments to evaluate ReasonMap from three perspectives: extraction quality on MSSE, the contribution of key design choices, and the downstream utility of extracted MSS.

4.1 Experimental Setup

Table 8 in Appendix A.6 summarizes the dataset statistics. Training hyperparameters, hardware, and DPO construction details are provided in Appendix A.5. For training, we use the warmup acquisition set $\mathcal{D}_{\text{acq}}^{\text{train}}$, the long-CoT generalization set $\mathcal{D}_{\text{long}}^{\text{train}}$, and the DPO datasets. For evaluation, we use two in-distribution sets, $\mathcal{D}_{\text{acq}}^{\text{test}}$ and $\mathcal{D}_{\text{long}}^{\text{test}}$, together with two external sets, $\mathcal{D}_{\text{DeepMath}}^{\text{test}}$ and $\mathcal{D}_{\text{Omni}}^{\text{test}}$, to test out-of-distribution generalization.

Baselines. Previous works mostly classify mathematical reasoning chains by prompting powerful LLMs like the ChatGPT series (Gao et al., 2024; He et al., 2025). We follow this precedent but establish our baselines based on two distinct selection criteria. First, to assess the task's inherent difficulty using cost-effective frontier models with strong instruction-following capabilities, we select GPT-4o-mini and DeepSeek-V3 (DeepSeek-AI et al., 2025) and their next-generation models, GPT-5-mini and DeepSeek-V3.1. Second, to isolate the effectiveness of our proposed framework, we evaluate the widely used models that serve as our backbone—Qwen2.5-7B-Instruct and Llama3.1-8B-Instruct. The prompt is provided in Appendix A.12.

Evaluation metrics. We use $\mathcal{D}_{\text{acq}}^{\text{test}}$ and $\mathcal{D}_{\text{long}}^{\text{test}}$ as in-distribution (ID) test data. Specifically, models take in the ID test data and predict a sequence of MSS. To measure predictions, we employ an LLM-as-Judge approach (Gu et al., 2024) where an LLM matches MSS between labels and generated outputs. We calculate the precision (P), recall (R), and their F1 score (F1 = 2P×R/(P + R)) as evaluation metrics. To avoid over-crediting overly broad labels, we separately analyze granularity mismatch errors in Appendix C.2. The detailed calculation and prompt are presented in Appendix A.13.

To assess the out-of-distribution (OOD) performance of ReasonMap, we additionally introduce two test sets for evaluation. We uniformly sampled 1,000 instances based on domain distributions from DeepMath-103K and Omni-Math datasets, denoted as $\mathcal{D}_{\text{DeepMath}}^{\text{test}}$ and $\mathcal{D}_{\text{Omni}}^{\text{test}}$, respectively. Because these OOD sets provide only broad domain labels, we report Domain Relevance Rate (DRR), a proxy that checks whether predicted MSS fall within the ground-truth domain of the problem. We further audit this proxy in Appendix C.3: GPT-4o-mini reaches over 0.90 precision and recall relative to expert item-domain judgments, and among OOD MSS judged domain-correct, only 18.3% are trivial. The full prompts and scoring details are provided in Appendices A.13 and A.14.

4.2 Main Results on MSSE

ReasonMap brings the clearest gains on long-CoT extraction. Table 1 shows that ReasonMap substantially improves both open-weight backbones, especially on $\mathcal{D}_{\text{long}}^{\text{test}}$, the setting most aligned with our task. For Qwen2.5-7B-Instruct, long-CoT F1 increases from 0.502 to 0.740 after long-CoT adaptation and remains strong after DPO. For Llama3.1-8B-Instruct, F1 increases from 0.475 to 0.733 after long-CoT adaptation and further reaches 0.750 after DPO. These gains indicate that task-specific training is crucial for recovering implicit MSS from extended reasoning traces, where prompting-only models often struggle to maintain both coverage and precision.

OOD performance suggests partial transfer. On the OOD benchmarks, ReasonMap improves over its corresponding open-weight backbones and is particularly competitive on $\mathcal{D}_{\text{DeepMath}}^{\text{test}}$: the DPO variants reach 0.455 DRR for Qwen2.5-7B and 0.432 DRR for Llama3.1-8B. However, frontier API models remain stronger on $\mathcal{D}_{\text{Omni}}^{\text{test}}$ in some cases. We therefore interpret the OOD results as evidence of domain-relevance transfer, not as uniform dominance over all frontier baselines.

Models	In-distribution (ID)						Out-of-distribution (OOD)	
	$\mathcal{D}^{\text{test}}_{\text{acq}}$			$\mathcal{D}^{\text{test}}_{\text{long}}$			$\mathcal{D}^{\text{test}}_{\text{DeepMath}}$	$\mathcal{D}^{\text{test}}_{\text{Omni}}$
	P	R	F1	P	R	F1	DRR	DRR
Frontier LLMs								
GPT-4o-mini	0.692	0.723	0.707	0.458	0.632	0.531	0.427	0.466
GPT-5-mini	0.632	0.708	0.667	0.505	0.609	0.552	0.460	0.476
DeepSeek-V3	0.756	0.717	0.735	0.676	0.682	0.678	0.420	0.411
DeepSeek-V3.1	0.779	0.671	0.721	0.523	0.588	0.553	0.420	0.428
Qwen2.5-7B-Instruct								
Base-Model	0.610	0.611	0.610	0.443	0.580	0.502	0.412	0.401
ReasonMap-7B$_{\text{acq}}$	**0.833**	**0.844**	**0.838**	0.488	**0.805**	0.607	0.357	0.376
ReasonMap-7B$_{\text{long}}$	0.804	0.798	0.800	0.716	0.766	**0.740**	0.434	0.350
ReasonMap-7B$_{\text{dpo}}$	0.786	0.766	0.775	**0.732**	0.737	0.734	**0.455**	**0.413**
Llama3.1-8B-Instruct								
Base-Model	0.577	0.649	0.611	0.359	0.702	0.475	0.321	0.365
ReasonMap-8B$_{\text{acq}}$	**0.788**	**0.801**	**0.794**	0.432	**0.791**	0.558	0.316	0.342
ReasonMap-8B$_{\text{long}}$	**0.788**	0.749	0.768	0.725	0.742	0.733	0.400	**0.389**
ReasonMap-8B$_{\text{dpo}}$	0.758	0.742	0.749	**0.789**	0.715	**0.750**	**0.432**	**0.389**

Table 1: Performance comparison of different models and training stages across multiple test sets. Metrics include precision (P), recall (R), F1 score, and Domain Relevance Rate (DRR).

DPO improves extraction fidelity by making the model more conservative. The results also reveal a calibration effect. The acquisition-only model often achieves high recall but tends to over-generate on long-CoT inputs, while long-CoT adaptation and DPO move the predicted-to-gold length ratio closer to 1, as shown in Figure 2. Accordingly, DPO should be viewed as a fidelity-oriented calibration step rather than a pure F1 maximizer: for Qwen2.5-7B, long-test F1 slightly drops after DPO, but precision and OOD DeepMath DRR improve. This trade-off is desirable for support-set extraction because hallucinated MSS can mislead downstream reasoning more severely than moderate omissions.

The training effect is consistent across backbones. Although Qwen2.5-7B-Instruct is generally stronger on the acquisition test set, Llama3.1-8B-Instruct becomes highly competitive on long-CoT extraction after ReasonMap training. The consistent improvements across both backbones suggest that the main gains come from the ReasonMap pipeline itself, rather than from a model-specific artifact.

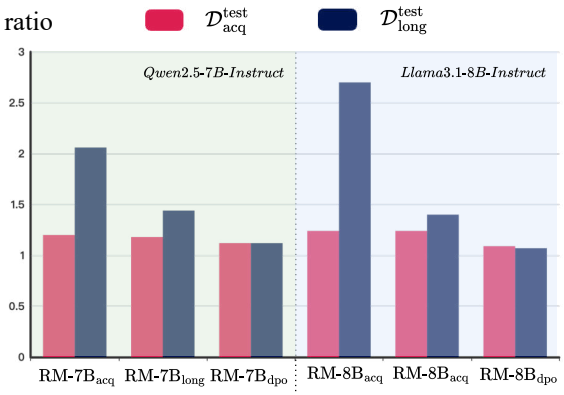

Figure 2: Predicted-to-gold MSS length ratio across training stages. Long-CoT adaptation and DPO move the ratio closer to 1, reflecting more conservative MSS extraction.

4.3 Ablation Studies

Table 2 studies the contribution of the main design choices in ReasonMap, including the proof-to-CoT support transfer curriculum, long-CoT data construction, definition-level supervision, and the DPO negative-selection threshold.

The short-to-long curriculum is important for long-CoT extraction. Sequential training achieves the best performance on the target long-CoT setting, reaching 0.740 F1 on $\mathcal{D}^{\text{test}}_{\text{long}}$, while joint training drops to 0.676. Although joint training slightly improves F1 on $\mathcal{D}^{\text{test}}_{\text{acq}}$ and OOD DRR on $\mathcal{D}^{\text{test}}_{\text{Omni}}$, it underperforms on the setting most aligned with our goal. This suggests that first learning a stable MSS vocabulary from premise-explicit examples and then adapting to long CoT traces is more effective than mixing both distributions from the beginning. The weaker result of $sequential\text{-}ep1$ further shows that the warmup stage should be sufficiently strong before long-CoT adaptation.

Both guided long-CoT construction and definition-level supervision matter. Removing the concise proof hint during long-CoT generation leads to a clear drop on both ID test sets, indicating that unguided expansion can introduce a mismatch

Training Method	In-distribution (ID)						Out-of-distribution (OOD)	
	$\mathcal{D}^{\text{test}}_{\text{acq}}$			$\mathcal{D}^{\text{test}}_{\text{long}}$			$\mathcal{D}^{\text{test}}_{\text{DeepMath}}$	$\mathcal{D}^{\text{test}}_{\text{Omni}}$
	P	R	F1	P	R	F1	DRR	DRR
ReasonMap-7B$_{\text{long}}$								
- *sequential*	**0.804**	0.798	0.800	**0.716**	**0.766**	**0.740**	**0.434**	0.350
- *joint*	0.780	**0.830**	**0.804**	0.653	0.701	0.676	0.410	**0.416**
- w/o *hint*	0.733	0.776	0.753	0.649	0.694	0.670	0.415	0.362
- *sequential-ep*1	0.732	0.780	0.755	0.649	0.751	0.691	0.431	0.406
- *drop-*1*k-def*	0.758	0.816	0.785	0.618	0.772	0.685	0.419	0.402
- *drop-*3*k-def*	0.718	0.765	0.740	0.618	0.762	0.682	0.398	0.388
ReasonMap-7B$_{\text{dpo}}$								
- $\gamma = 0.9$	**0.786**	**0.766**	**0.775**	**0.732**	**0.737**	**0.734**	**0.455**	**0.413**
- $\gamma = 0.7$	0.784	0.762	0.772	0.724	0.715	0.719	0.451	0.397
- $\gamma = 0.5$	0.781	0.740	0.759	**0.732**	0.728	0.729	0.448	0.401

Table 2: Ablation study on different variants of training strategies.

between the generated reasoning path and the original MSS labels. Similarly, dropping definition-only examples consistently degrades performance, especially when more such examples are removed. These results support two components of proof-to-CoT support transfer: the long-CoT traces should remain anchored to the referenced proof path, and explicit concept-level supervision helps the model normalize and recover MSS across different reasoning contexts.

DPO benefits from harder rejected candidates. For hard-negative support calibration, $\gamma = 0.9$ performs best overall, achieving the strongest results on both ID sets and both OOD benchmarks. Lower thresholds produce easier negatives and lead to slightly weaker performance. This indicates that DPO is most useful when rejected candidates are close to the gold MSS but still contain subtle errors, which matches the goal of reducing plausible-but-wrong MSS predictions.

4.4 Downstream Utility of Extracted MSS

We evaluate whether recovered MSS are actionable for downstream problem solving in three increasingly practical settings. First, reference-trace MSS test whether explicit MSS improve mathematical reasoning when supplied to the solver. Second, self-generated MSS test whether ReasonMap can provide useful hints without access to reference solutions at inference time. Third, AIMO experiments test whether the same type of hint remains useful for a strong frontier reasoning model.

4.4.1 Reference MSS as Controlled Hints

To isolate whether explicit MSS help problem solving, we first evaluate MSS prompting on MATH-500[1] and $\mathcal{D}^{\text{test}}_{\text{DeepMath}}$ using Qwen2.5-7B-Instruct and Llama3.1-8B-Instruct. Because these benchmarks do not provide ProofWiki-style MSS, we construct reference-trace MSS by applying ReasonMap to the available reference solutions; these hints are controlled inputs for analysis, not native benchmark labels. The prompt structure is given in Appendix A.8.

Method	MATH-500	$\mathcal{D}^{\text{test}}_{\text{DeepMath}}$
Qwen2.5 Model	0.740	0.506
+ Ref.-MSS	0.759	0.518
Llama3.1 Model	0.483	0.302
+ Ref.-MSS	0.497	0.322

Table 3: Accuracy with and without ReasonMap MSS extracted from reference solutions on MATH-500 and $\mathcal{D}^{\text{test}}_{\text{DeepMath}}$.

Table 3 shows consistent gains across both models and datasets. The effect is larger on the more challenging $\mathcal{D}^{\text{test}}_{\text{DeepMath}}$ benchmark, where Qwen2.5-7B gains 1.2% and Llama3.1-8B gains 2.0%. This controlled comparison establishes that MSS can improve downstream reasoning, especially when problems are more knowledge-intensive. A case study in Appendix A.7 further illustrates how MSS guide the model toward a more rigorous reasoning path.

4.4.2 Self-Generated MSS as Practical Hints

Controlled reference-trace hints establish that MSS are useful, but inference-time use is more compelling when no reference solution is available. We therefore evaluate a self-generated setting in which the base model first samples $k = 4$ candidate reasoning paths, ReasonMap extracts MSS from these

[1] https://huggingface.co/datasets/HuggingFaceH4/MATH-500

rollouts, and the aggregated MSS is fed back as a hint for a final reasoning pass.

Method Setting	MATH-500	$\mathcal{D}^{\text{test}}_{\text{DeepMath}}$
Base Model	0.740	0.506
+ SG MSS	**0.751**	**0.513**
+ Ref.-MSS	0.759	0.518

Table 4: Performance with self-generated (SG) MSS on Qwen2.5-7B-Instruct, with reference-trace MSS included as a controlled comparison.

As shown in Table 4, self-generated MSS hints improve over the base model on both test sets and recover much of the reference-trace hint gain without access to reference solutions at inference time. This demonstrates a practical inference loop, although the gain is not isolated from the additional rollout budget.

4.4.3 MSS Benefit Strong Reasoning Models

Finally, we test whether explicit MSS remain useful when the solver is already strong. For AIMO[2], MSS are generated by ReasonMap from sampled reasoning rollouts without access to reference MSS or final answers. The results in Table 5 show that these hints improve both Qwen2.5-7B-Instruct and DeepSeek-V3, including a **5.56% gain for DeepSeek-V3**.

Model	Base	+ MSS	Δ
Qwen2.5-7B	10.00%	14.44%	+4.44%
DeepSeek-V3	51.11%	56.67%	**+5.56%**

Table 5: Downstream performance on the AIMO validation set. ReasonMap-generated support hints improve both open-weight and frontier reasoning models.

4.5 Further Analysis

We include four additional analyses to clarify why ReasonMap works and how reliable the reported gains are.

Table 6 validates the automatic judge used in our ID and OOD evaluations. Across both domain-level and fine-grained support-item matching, GPT-4o-mini reaches average precision and recall above 0.90 against expert judgments; the sampling protocol is in Appendix C.1.

Table 7 shows that ReasonMap reduces the most harmful failure mode, inference drift and hallucination, from 58% to 48%, while shifting more residual errors toward omission. This trade-off is consistent with suppressing misleading support hints. More details are presented in Appendix C.2. We further audit OOD predictions judged as domain-correct: only **18.3%** of 300 accepted support items are trivial (Appendix C.3). Cost and Qwen3-8B transfer analyses are provided in Appendices E and B.

Task	Expert	P	R
Domain	A	0.9083	0.9033
	B	0.9100	0.9070
	C	0.9050	0.9080
	Avg.	*0.9078*	*0.9061*
Fine-grained	A	0.9240	0.9330
	B	0.9190	0.9280
	C	0.9300	0.9260
	Avg.	*0.9243*	*0.9290*

Table 6: Agreement between GPT-4o-mini judgments and three independent math experts.

Model	A: Gran.	B: Omit.	C: Drift
DeepSeek-V3	10%	32%	58%
ReasonMap	12%	40%	48%

Table 7: Distribution of failure modes on 100 low-overlap cases.

5 Conclusion

We introduced MSSE, a task for recovering the mathematical support set behind long CoT traces, and proposed ReasonMap, a trace-to-support framework that combines proof-to-CoT support transfer with hard-negative support calibration. Experiments show that ReasonMap is especially effective for ID long-CoT extraction and remains competitive under OOD domain-relevance evaluation. Downstream studies further show that reference-trace support hints improve mathematical problem solving, that ReasonMap can produce useful self-generated hints at inference time, and that explicit support hints can still benefit a strong frontier reasoning model. These results suggest that fine-grained support-set extraction is both an interpretability tool and a practical ingredient for knowledge-grounded reasoning.

[2] https://huggingface.co/datasets/AI-MO/aimo-validation-aime

Limitations

This work has several scope limitations. First, MSSE assumes a canonical vocabulary of named definitions and theorems. This makes extraction well-defined and comparable, but it may not capture informal heuristics, unnamed facts, or alternative concept granularity that human solvers sometimes use. Second, fine-grained gold MSS annotations are available only for the ID setting; for OOD datasets, we therefore report domain-relevance evaluation as a proxy rather than exact MSS accuracy. Future work could explore larger fine-grained MSS benchmarks across diverse mathematical domains, and extend MSS extraction to noisier, partially written, or interactive reasoning traces that better reflect real-world mathematical workflows.

Ethical Considerations

This work adheres to standard ethical guidelines for academic research. All datasets used for training and evaluation in this study were constructed from publicly available sources, ensuring that no private or sensitive data was involved. The base large language models we utilized for fine-tuning are also publicly accessible. The research focuses on MSS extraction, a task with no foreseeable negative societal impacts or ethical concerns. We are committed to transparency and responsible research practices.

References

Paul C Bogdan, Uzay Macar, Neel Nanda, and Arthur Conmy. 2025. Thought anchors: Which llm reasoning steps matter? *arXiv preprint arXiv:2506.19143*.

Jialun Cao, Yaojie Lu, Meiziniu Li, Haoyang Ma, Haokun Li, Mengda He, Cheng Wen, Le Sun, Hongyu Zhang, Shengchao Qin, Shing-Chi Cheung, and Cong Tian. 2025. From informal to formal – incorporating and evaluating LLMs on natural language requirements to verifiable formal proofs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 26984–27003, Vienna, Austria. Association for Computational Linguistics.

Sirui Chen, Changxin Tian, Binbin Hu, Kunlong Chen, Ziqi Liu, Zhiqiang Zhang, and Jun Zhou. 2025. Arrows of math reasoning data synthesis for large language models: Diversity, complexity and correctness. *Preprint*, arXiv:2508.18824.

Jiale Cheng, Xiao Liu, Cunxiang Wang, Xiaotao Gu, Yida Lu, Dan Zhang, Yuxiao Dong, Jie Tang, Hongning Wang, and Minlie Huang. 2025. Spar: Self-play with tree-search refinement to improve instruction-following in large language models. In *International Conference on Learning Representations*, volume 2025, pages 68675–68694.

Dongkyu Cho, Amy BZ Zhang, Bilel Fehri, Sheng Wang, Rumi Chunara, Rui Song, and Hengrui Cai. 2025. Correct reasoning paths visit shared decision pivots. *arXiv preprint arXiv:2509.21549*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, and Bingxuan Wang et al. 2025. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, Zhengyang Tang, Benyou Wang, Daoguang Zan, Shanghaoran Quan, Ge Zhang, Lei Sha, Yichang Zhang, Xuancheng Ren, Tianyu Liu, and Baobao Chang. 2024. Omni-math: A universal olympiad level mathematic benchmark for large language models. *Preprint*, arXiv:2410.07985.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, and 1 others. 2024. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*.

Yuhang He, Jihai Zhang, Jianzhu Bao, Fangquan Lin, Cheng Yang, Bing Qin, Ruifeng Xu, and Wotao Yin. 2024. BC-prover: Backward chaining prover for formal theorem proving. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3059–3077, Miami, Florida, USA. Association for Computational Linguistics.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2025. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *Preprint*, arXiv:2504.11456.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Joshua Ong Jun Leang, Aryo Pradipta Gema, and Shay B. Cohen. 2025. CoMAT: Chain of mathematically annotated thought improves mathematical reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 20245–20274, Suzhou, China. Association for Computational Linguistics.

Weixian Waylon Li, Yftah Ziser, Maximin Coavoux, and Shay B. Cohen. 2023. BERT is not the count: Learning to match mathematical statements with proofs. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3581–3593, Dubrovnik, Croatia. Association for Computational Linguistics.

Logan Murphy, Kaiyu Yang, Jialiang Sun, Zhaoyu Li, Anima Anandkumar, and Xujie Si. 2024. Autoformalizing euclidean geometry. In *Forty-first International Conference on Machine Learning*.

Stanislas Polu, Jesse Michael Han, Kunhao Zheng, Mantas Baksys, Igor Babuschkin, and Ilya Sutskever. 2023. Formal mathematics statement curriculum learning. In *ICLR*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Leonardo Ranaldi, Marco Valentino, and Andre Freitas. 2025. Improving chain-of-thought reasoning via quasi-symbolic abstractions. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 17222–17240, Vienna, Austria. Association for Computational Linguistics.

Joar Skalse, Nikolaus Howe, Dmitrii Krasheninnikov, and David Krueger. 2022. Defining and characterizing reward gaming. *Advances in Neural Information Processing Systems*, 35:9460–9471.

Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2025. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*.

Fengwei Teng, Zhaoyang Yu, Quan Shi, Jiayi Zhang, Chenglin Wu, and Yuyu Luo. 2025. Atom of thoughts for markov llm test-time scaling. *CoRR*, abs/2502.12018.

Constantin Venhoff, Iván Arcuschin, Philip Torr, Arthur Conmy, and Neel Nanda. 2025. Understanding reasoning in thinking language models via steering vectors. In *Workshop on Reasoning and Planning for Large Language Models*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Sean Welleck, Jiacheng Liu, Ximing Lu, Hannaneh Hajishirzi, and Yejin Choi. 2022. Naturalprover: Grounded mathematical proof generation with language models. *Advances in Neural Information Processing Systems*, 35:4913–4927.

Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Tanglifu Tanglifu, Xiaowei Lv, Haosheng Zou, Yongchao Deng, Shousheng Jia, and Xiangzheng Zhang. 2025. Light-r1: Curriculum SFT, DPO and RL for long COT from scratch and beyond. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 318–327, Vienna, Austria. Association for Computational Linguistics.

Ling Yang, Zhaochen Yu, Tianjun Zhang, Minkai Xu, Joseph E. Gonzalez, Bin CUI, and Shuicheng YAN. 2025a. Supercorrect: Advancing small LLM reasoning with thought template distillation and self-correction. In *The Thirteenth International Conference on Learning Representations*.

Wang Yang, Xiang Yue, Vipin Chaudhary, and Xiaotian Han. 2025b. Speculative thinking: Enhancing small-model reasoning with large model guidance at inference time. *arXiv preprint arXiv:2504.12329*.

Yuanhe Zhang, Ilja Kuzborskij, Jason D Lee, Chenlei Leng, and Fanghui Liu. 2025. Dag-math: Graph-guided mathematical reasoning in llms. *arXiv preprint arXiv:2510.19842*.

A Supplementary Details

A.1 Reproducibility Statement

We are committed to ensuring the reproducibility of our research. We have provided a detailed description of all experimental settings, hyperparameters, and the computational platform in the Appendix A.5. Upon publication, we will release all the training data constructed for our SFT and DPO stages, the evaluation datasets, and the source code for our data processing pipeline and evaluation scripts. Furthermore, the final trained model weights for our ReasonMap models will also be made publicly available.

A.2 Use of LLMs

We used LLMs in two distinct ways. For manuscript preparation, an LLM was used only for language editing. As part of the research pipeline, we used DeepSeek-R1 to generate proof-anchored long-CoT traces and GPT-4o-mini as an automatic judge, as described in Sections A.3 and A.13. All methodological decisions, experiments, analyses, and final claims were made by the authors.

A.3 Generation Prompt

Generation Prompt

You are an advanced reasoning assistant tasked with generating a mathematical proof for a given problem. I will provide you with a human-written, concise proof outline. Your task is to write a self-contained expanded solution that follows the logical structure and key steps of the provided outline, without copying its surface wording or explicitly mentioning the outline. Your proof should: 1) Adhere to the reasoning style of a logical, step-by-step mathematical proof. 2) Incorporate all relevant mathematical concepts and principles implied by the problem. 3) Avoid any direct mention of the human proof; treat it as if you are deriving the proof entirely independently. 4) Maintain clarity and rigor, ensuring that each step is justified and flows naturally from the previous.

A.4 Application to a Large-Scale Mathematical Dataset

To further demonstrate the practical importance of ReasonMap, we perform a case study using $\text{ReasonMap}_{\text{dpo}}$ to extract MSS from each sample in the DeepMath-103K dataset. For each difficulty level, we computed the average number of extracted MSS N_{MSS}. We also calculated the average MSS density for each difficulty, defined as:

$$\text{Density} = \frac{N_{\text{MSS}}}{\sqrt{\text{steps}}}, \tag{4}$$

where "steps" are determined by splitting the solution at each occurrence of "\n\n" (Yang et al., 2025b). Note that we omitted the easiest and hardest difficulty groups due to insufficient data in these categories. As shown in Figure 3 (b), our analysis reveals that the average number and density of MSS are higher in more difficult problems.

A.5 Experimental setting

The DPO dataset was constructed by generating N=4 responses per prompt and retaining rejected candidates in the hard-negative score window $\alpha \leq \mathcal{S}_k < \gamma$, with $\alpha = 0.3$ and $\gamma = 0.9$ in the main setting. Here $m_k = 2|\hat{L}_k \cap L|/(|\hat{L}_k| + |L|)$, and $r_k = 1 - u_k/n_k$, where u_k and n_k are the numbers of unique and total parsed MSS in $\hat{y}_k$. In the evaluation process, we selected GPT-4o-mini as our judge. Recent work has demonstrated its strong capabilities to handle complex tasks (Gu et al., 2024; Yang et al., 2025a; Teng et al., 2025) and its efficiency in terms of faster response and output times, making it a highly suitable choice for robust and scalable evaluation. Crucially, to validate its reliability for our ID and OOD MSSE evaluation task, we conducted a human evaluation (detailed in Appendix C.1). The results demonstrated a high agreement rate ($> 90\%$) between GPT-4o-mini and human experts, confirming it as a highly suitable choice for robust and scalable evaluation.

We selected Qwen2.5-7B-Instruct and Llama3.1-8B-Instruct as the backbone models for ReasonMap, and conducted full-parameter training with our two-stage pipeline. To further validate the adaptability of our framework to next-generation architectures, we also conducted experiments using the more recent Qwen3-8B as a backbone (results detailed in Appendix B). All experiments were performed on an Ascend Atlas 800T A2 server equipped with 8 × 64GB NPUs, utilizing the MindSpeed-LLM training framework for model optimization and acceleration. In the first stage of training, we set the learning rate to 1e-5, the batch size to 32, and the maximum sequence length to 1k tokens. Training was performed for 3 epochs using a cosine decay learning rate schedule, with a warmup ratio of 0.01. During the second stage, the learning rate was adjusted to 5e-6, and the maximum sequence length was increased to 8k tokens, while all other hyperparameters remained unchanged. For the DPO training stage, we set β to 0.1 and used the sigmoid function as the reward transformation. To mitigate reward hacking (Skalse et al., 2022) and stabilize the training, we incorporated SFT loss into the DPO training, with a loss coefficient of 0.5. The learning rate was set to 5e-7, and training was conducted for one epoch.

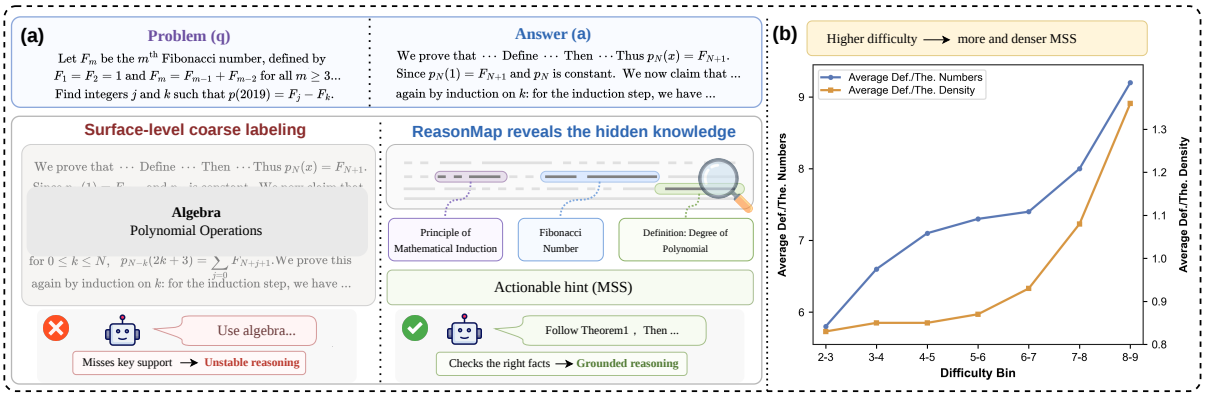

Figure 3: Motivating evidence for trace-grounded MSS. (a) Coarse domain labels such as "Algebra" identify the surface topic of a problem but do not reveal the concrete definitions and theorems used by a particular solution trace. (b) On DeepMath-103K, harder problems tend to induce larger and denser extracted MSS, suggesting that fine-grained MSS recovery becomes more important as reasoning difficulty increases.

We use the same system prompt across all stages, as shown in Appendix A.15.

A.6 Supplementary table and figure

Dataset	Total	Avg MSS	Avg len
$\mathcal{D}^{\text{train}}_{\text{acq}}$	39,885	2.63	90.17
$\mathcal{D}^{\text{train}}_{\text{long}}$	5,802	6.30	1,103.26
$\mathcal{D}_{\text{dpo-Llama}}$	7,637	–	735.20
$\mathcal{D}_{\text{dpo-Qwen}}$	6,655	–	705.66
$\mathcal{D}^{\text{test}}_{\text{acq}}$	1,135	6.17	184.86
$\mathcal{D}^{\text{test}}_{\text{long}}$	562	6.29	1190.47
$\mathcal{D}^{\text{test}}_{\text{DeepMath}}$	1,000	–	3069.56
$\mathcal{D}^{\text{test}}_{\text{Omni}}$	1,000	–	365.32

Table 8: Statistics of different datasets.

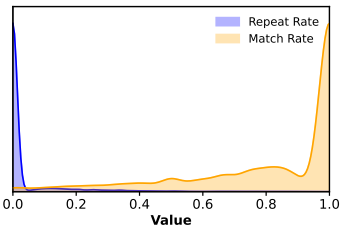

Figure 4: The distribution of repeat rate and exact-match precision in $\mathcal{D}_{\text{gen}}$.

A.7 Case Study

Question

Evaluate the contour integral $\int_\gamma \tanh(z)\, dz$ using the residue theorem, where $\gamma(t) = e^{it}$ for $0 \leq t \leq 2\pi$.
Relevant theorems and concepts:

- Residue Theorem
- Poles of Hyperbolic Tangent Function
- Definition: Order of Pole/Simple Pole
- Definition: Isolated Singularity
- Definition: Analytic Function/Complex Plane
- Contour Integral of Hyperbolic Tangent Function

Wrong Answer (w/o MSS)

To evaluate the contour integral $\int_\gamma \tanh(z)\, dz$ where $\gamma(t) = e^{it}$ for $0 \leq t \leq 2\pi$, we will use the residue theorem. The residue theorem states that for a function $f(z)$ that is analytic inside and on a simple closed contour γ except for a finite number of singular points inside γ, the integral is given by

$$\int_\gamma f(z)\, dz = 2\pi i \sum \text{Res}(f, a_i)$$

where the sum is over all the singular points a_i inside γ.
First, we identify the singular points of the function $\tanh(z)$. The hyperbolic tangent function $\tanh(z) = \frac{\sinh(z)}{\cosh(z)}$ has singular points where $\cosh(z) = 0$. The function $\cosh(z) = \frac{e^z + e^{-z}}{2}$ has zeros at $z = i\left(\frac{\pi}{2} + k\pi\right)$ for any integer k. The only singular points of $\tanh(z)$ inside the unit circle $|z| = 1$ are $z = i\frac{\pi}{2}$. Next, we need to find the residue of $\tanh(z)$ at $z = i\frac{\pi}{2}$. The residue of a function $f(z)$ at a simple pole $z = a$ is given by

$$\text{Res}(f, a) = \lim_{z \to a} (z - a) f(z)$$

For $\tanh(z)$ at $z = i\frac{\pi}{2}$,

$$\begin{aligned}\text{Res}\left(\tanh(z), i\frac{\pi}{2}\right) &= \lim_{z \to i\frac{\pi}{2}} (z - i\frac{\pi}{2}) \tanh(z) \\ &= \lim_{z \to i\frac{\pi}{2}} (z - i\frac{\pi}{2}) \frac{\sinh(z)}{\cosh(z)} \\ &= \lim_{z \to i\frac{\pi}{2}} \frac{\sinh(z)}{\cosh'(z)}.\end{aligned}$$

Since $\cosh'(z) = \sinh(z)$, we have

$$\text{Res}\left(\tanh(z), i\frac{\pi}{2}\right) = \lim_{z \to i\frac{\pi}{2}} \frac{\sinh(z)}{\sinh(z)} = 1$$

Thus, the residue of $\tanh(z)$ at $z = i\frac{\pi}{2}$ is 1. By the residue theorem, the value of the integral is

$$\int_\gamma \tanh(z)\, dz = 2\pi i \cdot 1 = 2\pi i$$

Therefore, the answer is

$$\boxed{2\pi i}$$

Correct Answer (w/ MSS)

To evaluate the contour integral $\int_\gamma \tanh(z)\,dz$ where $\gamma(t) = e^{it}$ for $0 \le t \le 2\pi$, we will use the residue theorem. The residue theorem states that for a function $f(z)$ that is analytic inside and on a simple closed contour γ except for a finite number of singular points inside γ, the integral of $f(z)$ around γ is given by

$$\int_\gamma f(z)\,dz = 2\pi i \sum_k \text{Res}(f, a_k)$$

where the sum is over all the singular points a_k inside γ.
First, we identify the singular points of the function $\tanh(z)$. The hyperbolic tangent function $\tanh(z) = \frac{\sinh(z)}{\cosh(z)}$ has singularities where $\cosh(z) = 0$. The solutions to $\cosh(z) = 0$ are $z = \frac{(2n+1)\pi i}{2}$ for $n \in \mathbb{Z}$. These are the points where $\cosh(z)$ has simple zeros.
Next, we need to determine which of these singular points lie inside the contour $\gamma(t) = e^{it}$ for $0 \le t \le 2\pi$. The contour γ is the unit circle in the complex plane, so we need to check if any of the points $z = \frac{(2n+1)\pi i}{2}$ lie within the unit disk $|z| < 1$.
For $n = 0$, we have $z = \frac{\pi i}{2}$, and $|\frac{\pi i}{2}| = \frac{\pi}{2} > 1$. For $n = -1$, we have $z = -\frac{\pi i}{2}$, and $|-\frac{\pi i}{2}| = \frac{\pi}{2} > 1$. For $n \neq 0, -1$, the magnitudes $|\frac{(2n+1)\pi i}{2}| = \frac{|2n+1|\pi}{2} > 1$.
Thus, there are no singular points of $\tanh(z)$ inside the unit disk. Since the function $\tanh(z)$ is analytic inside and on the unit circle, the integral of $\tanh(z)$ over the unit circle is zero.
Therefore, the value of the contour integral is

$$\boxed{0}$$

Guided by the theorem, the model strictly adhered to all its application conditions. It correctly calculated all the poles of $\tanh(z)$ as $z = \frac{(2n+1)\pi i}{2}$. Following this, it performed the most critical logical step: verifying whether any of these poles were located inside the path of integration (the unit circle). It correctly found that $|\frac{\pi i}{2}| \approx 1.57 > 1$, and therefore, no poles reside within the contour. According to the Residue Theorem, the integral is correctly determined to be zero.

This example shows that MSS can help even when the problem already mentions the main theorem: the additional concepts remind the solver to check the theorem's applicability conditions, including singularity locations and whether poles lie inside the contour. Without this richer MSS sheet, the model still invokes the Residue Theorem but incorrectly treats $z = \frac{i\pi}{2}$ as lying inside the unit circle, leading it to compute an irrelevant residue and return a non-zero answer.

A.8 System prompt for Problem-solving

Prompt with MSS

Your task is to solve the Math problem. You will be provided with relevant theorems and concepts as a reference to assist you. Please use this information to guide your reasoning and solution. If applicable, include the final answer in \boxed{} for closed-form results like multiple choices or mathematical solutions.
{question}
Relevant theorems and concepts:
{MSS}

Prompt without MSS

Your task is to solve the Math problem. Please reason carefully and include the final answer in \boxed{} when applicable.

A.9 Instances in Natural-Proofs-Gen

A definition-only instance in Natural-Proofs-Gen

```
<prompt>
<definition>
<title> Definition:Finite Group/Axioms </title>
<content>
A [[Definition:Finite Group|finite group]] is an
[[Definition:Algebraic Structure|algebraic structure]]
$\struct {G, \circ}$ which satisfies the following four
conditions:
{{begin-axiom}}
{{axiom | n = \text{FG} 0
      | lc= [[Definition:Closed Algebraic Structure|Closure]]
        | q = \forall a, b \in G
        | m = a \circ b \in G
}}
{{axiom | n = \text{FG} 1
      | lc= [[Definition:Associative Operation|Associativity]]
        | q = \forall a, b, c \in G
      | m = a \circ \paren{b \circ c} = \paren{a \circ b} \circ c
}}
{{axiom | n = \text{FG} 2
        | lc= [[Definition:Finite Set|Finiteness]]
        | q = \exists n \in \N
        | m = \order G = n
}}
{{axiom | n = \text{FG} 3
   | lc= [[Definition:Cancellable Operation|Cancellability]]
        | q = \forall a, b, c \in G
        | m = c \circ a = c \circ b \implies a = b
}}
{{axiom | m = a \circ c = b \circ c \implies a = b
}}
{{end-axiom}}
These four stipulations are called the finite group axioms.
</content>
</definition>
```

A problem–trace instance in Natural-Proofs-Gen

```
<prompt>
<theorem>
<title>
Real Number Subtracted from Itself leaves Zero
</title>
<content>
Let $x \in \R$ be a real number. Then:$x - x = 0$
where $x - x$ denotes the operation of real subtraction.
</content>
</theorem>
<reference> Inverses for Real Addition </reference>
<proof>
{{begin-eqn}}
{{eqn | l = x - x
      | r = x + \paren{-x}
      | c = {{Defof|Real Subtraction}}
}}
{{eqn | r = 0
      | c = [[Inverses for Real Addition]]
}}
{{end-eqn}}
{{qed}}
</proof>
```

A.10 An example of $\mathcal{D}_{\text{acq}}^{\text{train}}$

An example of $\mathcal{D}_{\text{acq}}^{\text{train}}$

Question (q)
Let G be a cyclic group. Let g be a generator of G. Let ϕ be an automorphism on G. Then $\phi(g)$ is also a generator of G.
Proof-Style Trace (a)
By definition of automorphism, ϕ is a homomorphism. It follows that this result is a specific instance of Homomorphic Image of Cyclic Group is Cyclic Group. □
MSS (L)

- Homomorphic Image of Cyclic Group is Cyclic Group
- Definition:Group Automorphism
- Definition:Cyclic Group/Generator
- Definition:Group homomorphism

A.11 An example of $\mathcal{D}_{\text{long}}^{\text{train}}$

An example of $\mathcal{D}_{\text{long}}^{\text{train}}$

Question (q)
Let G be a cyclic group. Let g be a generator of G. Let ϕ be an automorphism on G. Then $\phi(g)$ is also a generator of G.
Long CoT Trace (a^{ℓ})
Okay, so I need to prove that if G is a cyclic group generated by an element g, and ϕ is an automorphism of G, then the image of g under ϕ, which is $\phi(g)$, is also a generator of G. Let me start by recalling some basic definitions and properties here.
First, a cyclic group is a group that can be generated by a single element. That means every element in G can be written as some power of g, right? So $G = \{g^0, g^1, g^2, \ldots, g^{n-1}\}$ if G is finite of order n, or all integer powers if it's infinite cyclic.
An automorphism ϕ is a bijective homomorphism from G to itself. Since ϕ is a homomorphism, it preserves the group operation. So for any integers k and m, $\phi(g^k) = \phi(g)^k$, because homomorphisms respect exponents. Also, being bijective means ϕ is both injective and surjective.
Now, the goal is to show that $\phi(g)$ generates G. Since G is cyclic and generated by g, every element in G is of the form g^k for some integer k. Because ϕ is surjective, every element in G must be the image of some element under ϕ. But since ϕ is a homomorphism, the image of any element in G under ϕ can be expressed as $\phi(g)$ raised to some power. Wait, let me think again.
Since ϕ is an automorphism, it's surjective, so every element in G must be hit by ϕ. But ϕ is a homomorphism, so for any h in G, there exists some a in G such that $\phi(a) = h$. But since G is cyclic, $a = g^m$ for some m. Therefore, $h = \phi(g^m) = \phi(g)^m$. That means every element h in G is some power of $\phi(g)$, which would imply that $\phi(g)$ generates G. Therefore, $\phi(g)$ is a generator.
Alternatively, maybe I should approach this by considering that the image of a cyclic group under a homomorphism is cyclic. Since ϕ is surjective (because it's an automorphism), the image of G under ϕ is G itself. Then, since G is cyclic, the image must be generated by the image of the generator g, which is $\phi(g)$. Therefore, $\phi(g)$ must generate G.
Wait, but the homomorphic image of a cyclic group is cyclic, yes. Since ϕ is a homomorphism from G to G, and ϕ is surjective, then $\phi(G) = G$. Therefore, $\phi(G)$ is cyclic and generated by $\phi(g)$. Hence, $\phi(g)$ is a generator of G.
I think that covers it. Let me make sure there's no gap here. The key points are that automorphisms are surjective homomorphisms, and the image of a generator under a surjective homomorphism must generate the image group, which in this case is G itself. So yeah, $\phi(g)$ generates G. That should do it.
Proof:
Let G be a cyclic group generated by an element g, and let $\phi : G \to G$ be an automorphism. We aim to show that $\phi(g)$ is also a generator of G.
1. Properties of Automorphisms: Since ϕ is an automorphism, it is a bijective homomorphism. Bijectivity ensures ϕ is surjective, meaning $\phi(G) = G$. Additionally, homomorphisms preserve group operations, so for any integer k,

$$\phi(g^k) = \phi(g)^k.$$

2. Image of G Under ϕ: As ϕ is surjective, every element $h \in G$ can be expressed as $h = \phi(a)$ for some $a \in G$. Since G is cyclic and generated by g, there exists an integer m such that $a = g^m$. Thus,

$$h = \phi(g^m) = \phi(g)^m.$$

3. Generating G via $\phi(g)$: The equality $h = \phi(g)^m$ demonstrates that every element $h \in G$ is a power of $\phi(g)$. Consequently, $\phi(g)$ generates G, making $\phi(g)$ a generator of G.
4. Conclusion: Therefore, the automorphism ϕ maps the generator g to another generator $\phi(g)$, preserving the cyclic structure of G.

□

MSS (L)

- Homomorphic Image of Cyclic Group is Cyclic Group
- Definition:Group Automorphism
- Definition:Cyclic Group/Generator
- Definition:Group homomorphism

A.12 The system prompt for Baseline LLMs

The system prompt for Baseline LLMs

You are an MSS extractor. Given a problem and a completed solution trace, output only the canonical names of the definitions and theorems that support the written solution. If a theorem is used, label it with ## before and after the theorem name. If a concept is used, prefix the concept name with "Definition:" and enclose it with ##.
Here are some examples of how to format your output:
e.g.:
input: {example}
output: {output_example}
DO NOT include any explanations or additional text in your output, just output the theorems or definitions separate by "##". Here are new problem–trace inputs you need to extract MSS from:

A.13 The LLM-as-Judge prompt for MSS matching (ID evaluation)

The LLM-as-Judge prompt for MSS matching (ID evaluation)

You will receive two strings, A and B, each containing a list of theorem names or mathematical concepts separated by ##. Your task is to output a series of similar pairs, where:

- The left item of each pair comes from A.
- The right item of each pair comes from B.

Similarity Criteria:

- Match terms with identical or near-identical names (e.g., "Semilattice", "Semilattice")

- Pair items where one is a variant/subset of the other (e.g., “Closure”, “Closure (Abstract Algebra)”).
- Ignore minor formatting differences (e.g., “Idempotence”, “Idempotence/Operation”).
- Prioritize conceptual alignment over exact wording (e.g., “Associativity”, “Associative”).

Output Format:
Return a list of pairs in the format [[A_{item}, B_{item}]], one per line. If no match is found for an item, omit it.
Example:
Input:
A:

- Definition:Upper Semilattice
- Definition:Semilattice
- Definition:Closure
- Definition:Associativity
- Definition:Commutativity
- Definition:IdempotenceOperation
- Definition:Supremum of Set

B:

- Definition:Associative
- Definition:Algebraic Structure
- Definition:IdempotenceOperation
- Definition:Closure (Abstract Algebra)Algebraic Structure
- Definition:Upper Semilattice on Classical Set
- Definition:CommutativeOperation
- Upper Semilattice on Classical Set is Semilattice

Output:

- [[Definition:Upper Semilattice, Definition:Upper Semilattice on Classical Set]]
- [[Definition:Semilattice, Definition:Semilattice]]
- [[Definition:Closure, Definition:Closure (Abstract Algebra)Algebraic Structure]]
- [[Definition:Associativity, Definition:Associative]]
- [[Definition:Commutativity, Definition:CommutativeOperation]]
- [[Definition:Idempotence, Definition:IdempotenceOperation]]

Do not output any additional text or explanations, just the pairs. Here are the two termstheorems to compare:

For example, if the MSS in labels are “Definition:A, B, Definition:C”, and the MSS in the output are “Definition:A, B”. The LLM judge outputs the matched MSS: “Definition:A, B”, then the precision is $2/2 = 1$, the recall is $2/3$, and the F1 score is $2 \cdot (1 \cdot 2/3)/(1 + 2/3) = 4/5$.

A.14 The LLM-as-Judge prompt for domain relevance match (OOD evaluation)

The LLM-as-Judge prompt for domain relevance match (OOD evaluation)

Act as a mathematics expert. You will receive two strings, A and B. B string contains a list of theorem names or mathematical concepts separated by ##. Given two mathematical sets:

- **A:** A mathematical domain/topic (e.g., "Linear Algebra," "Topology," "Calculus").
- **B:** A list of mathematical theorems, concepts, or terms.

Perform the following:

1. Check if each item in B belongs to the domain A (based on standard mathematical classifications).
2. Return all strings from B that are definitively part of A, preserving their original formatting (e.g., capitalization, symbols).

Additional Notes:

- If no items in B match A, output `None`.

Output Format: Select the items from B that are part of A, separated by ##.
Example:

- **Input:**
  A:`<|Mathematics -> Precalculus -> Limits|>`
  B:`<|L’Hopital’s Rule##`
  `Definition:Limit of Real Function|>`
  Output: `L’Hopital’s Rule##`
  `Definition:Limit of Real Function`
- **Input:**
  A:`<|Mathematics -> Algebra -> Intermediate Algebra ->`
  `Quadratic Functions|>`
  B:`<|Solution to Quadratic Equation##`
  `Definition:Real Number##Definition:Domain (Set)|>`
  Output:
  `Solution to Quadratic Equation##`
  `Definition:Real Number##Definition:Domain (Set)`
- **Input:**
  A:`<|Mathematics -> Algebra -> Linear Algebra|>`
  B:`<|Euler’s Formula##Determinant##Fourier`
  `Transform##Eigenvector##Riemann Hypothesis|>`
  Output: `Determinant##Eigenvector`
- **Input:**
  A:`<|Mathematics -> Geometry -> Solid Geometry`
  `-> Surface Area|>`
  B:`<|Definition:Angle##Definition:Area##`
  `Definition:Circle/Center|>`
  Output: `Definition:Angle##Definition:Area##`
  `Definition:Circle/Center`
- **Input:**
  A:`<|Mathematics -> Geometry -> Solid Geometry`
  `-> Surface Area|>`
  B:`<|Intermediate Value Theorem|>`
  Output: `None`

Here are the two terms/theorems to compare:

A.15 The system prompt for ReasonMap model

The system prompt ReasonMap models

You are an MSS extractor. Given a problem and a completed solution trace, output only the canonical names of the definitions and theorems that support the written solution. Separate MSS with ##. Do not solve the problem or provide explanations.

B Extended Experimental Results

B.1 Performance on Next-Generation Models

To validate the adaptability of our framework to the rapid iteration of open-weight LLMs, we applied ReasonMap to a newer base model: **Qwen3-8B**. Unlike the instruction-tuned variants used in our main experiments, Qwen3-8B is a base model without inherent instruction-following capabilities. Despite this, after training with our ReasonMap pipeline, it demonstrates exceptional performance.

Table 9 presents the results. The **ReasonMap-Qwen3-8B$_{\mathbf{dpo}}$** variant achieves the strongest ID F1 scores and the best DeepMath OOD DRR among the compared ReasonMap variants. Its Omni-Math OOD DRR is slightly lower than the Qwen2.5-7B variant, indicating that the gain is not uniform across every OOD metric. Overall, these results suggest that our training framework can transfer to a newer backbone model and improve fine-grained knowledge extraction.

C Human Evaluation and Qualitative Analysis

We conducted a series of human evaluations involving three PhDs in Mathematics to validate our automated metrics and analyze model behavior.

C.1 Validation of LLM-as-a-Judge

To ensure the reliability of GPT-4o-mini as our automated judge, we constructed a validation set by randomly sampling 100 instance pairs from our test outputs, including 50 fine-grained MSS pairs from the In-Distribution (ID) set and 50 domain label pairs from the Out-of-Distribution (OOD) set. The human evaluators independently assessed these pairs for semantic equivalence, and their matching results are used to calculate the precision and recall with the results judged by GPT-4o-mini. As reported in Table 6, the automated judge exhibited strong alignment with human experts, achieving high Average Precision and Recall scores (> 0.90) across both fine-grained MSS matching and OOD domain categorization. These consistent high-agreement results confirm the effectiveness of GPT-4o-mini in capturing semantic correctness, validating its use as a reliable proxy for MSSE task evaluation.

C.2 Error Analysis and Taxonomy

We performed a fine-grained error analysis on 100 failure cases (where overlap $< 50\%$) from both DeepSeek-V3 and ReasonMap-7B. Errors were classified into three categories:

- **Code A (Granularity Mismatch):** The predicted concept is correct, but the level of detail/specificity is inconsistent with the label. The prediction is a generalization of the label, or the label is a generalization of the prediction.
- **Code B (Implicit Omission):** The model fails to extract one or more critical and non-trivial MSS that are present in the Ground Truth Label. (Focus on key theorems or domain-specific definitions that are entirely missing).
- **Code C (Inference Drift / Hallucination):** The model predicts MSS that are clearly irrelevant or semantically distant from the actual core concepts in the Ground Truth Label. (The prediction includes concepts that share very little semantic overlap with any MSS in the label).

Table 7 shows the distribution. The most critical success of ReasonMap lies in the suppression of Inference Drift (Code C) errors. These errors are considered the most destructive because they introduce misleading or semantically distant concepts into the recovered MSS. DeepSeek-V3's primary failure mode is Code C, accounting for **58%** of its total errors. ReasonMap successfully reduces this high-risk failure mode by 10 percentage points (from 58% to **48%**). This demonstrates that our training strategy (SFT+DPO) effectively suppresses the generation of noisy and unreliable concepts, preventing the model from "hallucinating" connections that do not logically exist.

The error distribution in ReasonMap shifts towards categories that are more recoverable and less harmful: 1) Granularity Mismatch (Code A): These errors are semantically controllable (i.e., "correct concept, wrong specificity"). ReasonMap is slightly more prone to this recoverable error type (12% vs. 10%), indicating it stays "on topic" even when the precision is imperfect. 2) Implicit Omission (Code B): The increase in omissions (40% vs. 32%) suggests a more conservative strategy. We acknowledge this increase in Code B as a side effect of our calibration algorithm. In the process of suppressing excessive model output to reduce hallucinations, the model effectively raises its internal confidence threshold, which inevitably increases the probability of missing some valid but less obvious key information.

Models	In-distribution (ID)						Out-of-distribution (OOD)	
	$\mathcal{D}^{\text{test}}_{\text{acq}}$			$\mathcal{D}^{\text{test}}_{\text{long}}$			$\mathcal{D}^{\text{test}}_{\text{DeepMath}}$	$\mathcal{D}^{\text{test}}_{\text{Omni}}$
	P	R	F1	P	R	F1	DRR	DRR
Qwen2.5-7B-Instruct								
Base-Model	0.610	0.611	0.610	0.443	0.580	0.502	0.412	0.401
ReasonMap-7B$_{\text{dpo}}$	0.786	0.766	0.775	0.732	0.737	0.734	0.455	0.413
Llama3.1-8B-Instruct								
Base-Model	0.577	0.649	0.611	0.359	0.702	0.475	0.321	0.365
ReasonMap-8B$_{\text{dpo}}$	0.758	0.742	0.749	0.789	0.715	0.750	0.432	0.389
Qwen3-8B								
ReasonMap-Qwen3-8B$_{\text{dpo}}$	0.864	0.858	0.860	0.767	0.759	0.763	0.469	0.400

Table 9: Performance comparison of ReasonMap across different backbone architectures. The Qwen3-8B variant achieves stronger ID F1 scores and higher DeepMath OOD DRR, while its Omni-Math OOD DRR is not uniformly better than all other variants.

This targeted error analysis confirms that ReasonMap does not merely improve overall F1 scores; it fundamentally improves the quality and safety of model outputs during failure cases by significantly reducing the most unreliable error pattern (Code C), even at the cost of increased omissions.

C.3 Evaluation of OOD Triviality

To address concerns that the automated "Domain Relevance" metric (judged by GPT-4o-mini) might artificially inflate performance by including many trivial tags (e.g., "Definition: Set"), we conducted a targeted audit to determine the triviality rate within the accepted predictions.

Specifically, we sampled 300 extracted MSS that were already classified by GPT-4o-mini as belonging to the given domain. Human experts then judged these validated MSS to identify those that were "trivial" or "uninformative" given the problem context. The evaluation reveals that only **18.3%** of the MSS were classified as trivial. This low ratio confirms that the high domain relevance scores reported in our main results largely reflect the retrieval of non-trivial, substantial mathematical knowledge, rather than generic or redundant concepts.

D Data Construction Details

In the proof-to-CoT MSS transfer stage, we augment the dataset by prompting an LLM to generate long-form CoT traces (a^{ℓ}) based on concise proof-style traces (a). A potential concern is whether the model simply copies or paraphrases the short trace ("Over-Imitation") rather than generating a genuinely detailed reasoning path.

The divergence between Precision and Recall provides critical insights into the generation quality: 1) The low Precision scores (e.g., ROUGE-1 ≈ 0.11) are expected and desirable. Since the generated CoT traces are approximately **30**$\times$ longer than the original proof-style traces (expanding from $<$0.1k to $\approx$3k tokens), the original tokens constitute only a small fraction of the generated text. This confirms that the model is *not* simply copying the input, but is performing substantial expansion beyond the concise trace. 2) Conversely, the relatively high Recall scores (e.g., ROUGE-1 ≈ 0.74) indicate that the vast majority of key terms and steps from the original concise proof-style trace (a) are successfully retained within the generated long CoT trace (a^{ℓ}). The results show the augmentation faithfully diversifies the reasoning data and maintains the correct answer, instead of introducing noise and bias.

Metric	Precision	Recall	F1-Score
ROUGE-1	0.1089	0.7398	0.1771
ROUGE-2	0.0578	0.4135	0.0948
ROUGE-L	0.0776	0.5720	0.1280

Table 10: ROUGE scores between original concise proof-style traces and generated long CoT traces. The low overlap indicates significant structural and content expansion.

E Cost-Effectiveness Analysis

A key motivation for developing ReasonMap is to provide a scalable, cost-effective solution for fine-grained MSS extraction. While proprietary models like the recently released **GPT-5-mini** offer competitive pricing, their costs scale linearly with dataset size, making them prohibitively expensive for extracting MSS for massive corpora required for pre-training (e.g., millions of samples).

Parameters and Assumptions.

- **Workload:** We estimate costs for extracting MSS from 1 Million (1M) samples.
- **Token Counts:** We assume an average input length of 2,000 tokens and an output length of 500 tokens per instance, totaling 2.5 billion tokens.
- **GPT-5-mini Pricing**[3]**:** $0.25 per 1M input tokens and $2.00 per 1M output tokens (August 2025 pricing).
- **ReasonMap Pricing:** Our cost model uses standard cloud pricing for NVIDIA A100 SXM GPUs ($1.79/hour/GPU[4]).
  - *Training:* We assume a fine-tuning run using 8 GPUs for 8 hours.
  - *Inference:* We assume deployment on a single GPU. With vLLM acceleration, the system achieves a throughput of ∼3,362 tokens/sec [5].

Based on these data, the inference cost is derived as follows:

$$T_{\text{sec}} = \frac{2{,}500{,}000{,}000}{3362.71} \approx 743{,}448 \text{ s},$$
$$T_{\text{hr}} = \frac{743{,}448}{3{,}600} \approx 206.51 \text{ h},$$
$$C_{\text{infer}} = 206.51 \times \$1.79 \approx \mathbf{\$369.65}.$$

Table 11 presents the economic analysis. While the proprietary API incurs linear costs reaching $1,500, the self-hosted approach—even accounting for an 8-GPU training cluster—costs significantly less.

ReasonMap demonstrates a clear economic advantage for large-scale data synthesis. The combined cost of training and inference (∼$484) is substantially lower than the fees for a comparable volume of API calls. This allows researchers to trade variable API costs for fixed compute time, making massive dataset generation feasible within academic budgets.

[3] https://openai.com/api/pricing/
[4] https://lambda.ai/instances
[5] https://www.databasemart.com/blog/vllm-gpu-benchmark-a100-80gb

Method	Cost Breakdown (1M Samples)		Total Cost
	Operational Components	Cost Calculation	
GPT-5-mini (API)	Input Processing	2,000M tok $\times$ \$0.25	\$500.00
	Output Generation	500M tok $\times$ \$2.00	\$1,000.00
		API Total:	**\$1,500.00**
ReasonMap-7B (Ours)	Training (8x A100)	8 hours $\times$ 8 $\times$ \$1.79	\$114.56
	Inference (1x A100)	$\sim$207 hours $\times$ \$1.79	\$369.66
		Self-Hosted Total:	**\$484.22**

Table 11: Cost comparison for extracting MSS from 1 million samples. Even with the upfront cost of training on an 8-GPU cluster, the open-source ReasonMap model reduces the total expense by **approx. 68%** (\$484 vs. \$1,500) compared to the proprietary frontier API. Inference costs are calculated based on a measured throughput of 3,362 tokens/s.