CAMA: Enhancing Mathematical Reasoning in Large Language Models with Causal Knowledge

Lei Zan[*†1], **Keli Zhang**[*†1], **Ruichu Cai**[2,3], **Lujia Pan**[1]
[1]Huawei Noah's Ark Lab
[2]Guangdong University of Technology
[3]Peng Cheng Laboratory
{xdzanlei, cairuichu}@gmail.com, {zhangkeli1, panlujia}@huawei.com

Abstract

Large Language Models (LLMs) have demonstrated strong performance across a wide range of tasks, yet they still struggle with complex mathematical reasoning, a challenge fundamentally rooted in deep structural dependencies. To address this challenge, we propose **CA**usal **MA**thematician (**CAMA**), a two-stage causal framework that equips LLMs with explicit, reusable mathematical structure. In the learning stage, CAMA first constructs the **M**athematical **C**ausal **G**raph (**MCG**), a high-level representation of solution strategies, by combining LLM priors with causal discovery algorithms applied to a corpus of question-solution pairs. The resulting MCG encodes essential knowledge points and their causal dependencies. To better align the graph with downstream reasoning tasks, CAMA further refines the MCG through iterative feedback derived from a selected subset of the question-solution pairs. In the reasoning stage, given a new question, CAMA dynamically extracts a task-relevant subgraph from the MCG, conditioned on both the question content and the LLM's intermediate reasoning trace. This subgraph, which encodes the most pertinent knowledge points and their causal dependencies, is then injected back into the LLM to guide its reasoning process. Empirical results on real-world datasets show that CAMA significantly improves LLM performance on challenging mathematical problems. Furthermore, our experiments demonstrate that structured guidance consistently outperforms unstructured alternatives, and that incorporating asymmetric causal relationships yields greater improvements than using symmetric associations alone.

Code — https://github.com/huawei-noah/trustworthyAI/tree/master/research/CAMA

Introduction

Large Language Models (LLMs), such as the GPT series (OpenAI et al. 2024), DeepSeek series (DeepSeek-AI et al. 2025b,a), and Pangu series (Yin et al. 2025; Tang et al. 2025), have achieved remarkable advances across a wide spectrum of language tasks, including question answering, code synthesis, and information retrieval (Ouyang et al. 2022; Gu 2023; Dai et al. 2024).

Despite these advances, LLMs still struggle with challenging mathematical problems (Han, Puduppully, and Chen 2023), where solutions require formal rigor, symbolic manipulation, and multi-step deduction (Cobbe et al. 2021). This limitation can be attributed primarily to two key challenges. First, the inherent architectural constraints of transformers impose a fixed depth on reasoning, limiting the model's ability to carry out deep and interdependent logical inferences (Merrill and Sabharwal 2023; Liu, Lin, and Liu 2024). Second, LLMs primarily rely on statistical pattern recognition, making them sensitive to minor changes in problem phrasing and prone to brittle or inconsistent outputs (Mirzadeh et al. 2024; Jiang et al. 2024).

To address these challenges, we advocate for a shift from purely data-driven prediction toward structured reasoning guidance. Drawing inspiration from the dictum "teach a man to fish rather than give him a fish," we argue that equipping a model with explicit, reusable problem-solving strategies is an efficient way to strengthen its reasoning abilities. To this end, we formalize such strategies as a **M**athematical **C**ausal **G**raph (**MCG**): a directed acyclic graph in which each node is a knowledge point (*e.g.*, *Area of a circle*) and each edge encodes a causal dependency (*e.g.*, computing *Volume of a cylinder* first requires the circle's area). The MCG stores information that is generally applicable across contexts, and the edge direction specifies the order of reasoning, which is especially valuable when the required chain of knowledge spans many steps. Integrating such a graph into the prompt helps decompose complex problems into coherent subtasks, reduce reliance on implicit reasoning, and increase intermediate accuracy, while curbing hallucinations and redundancy.

Building on these insights, we present **CA**usal **MA**thematician (**CAMA**), a plug-and-play framework that integrates causal discovery with LLMs to construct and exploit MCGs. CAMA operates in two stages: (1) in the learning stage, CAMA parses LLM-generated chain-of-thought solutions to extract underlying knowledge points, then infers the causal dependencies among them. The graph is subsequently refined via feedback from the model's own question-answering accuracy. (2) In the reasoning stage, CAMA feeds this structured information back into the LLM via natural-language prompts to boost problem-solving performance, without the need for any parameter updates,

[*]These authors contributed equally.
[†]Corresponding authors.

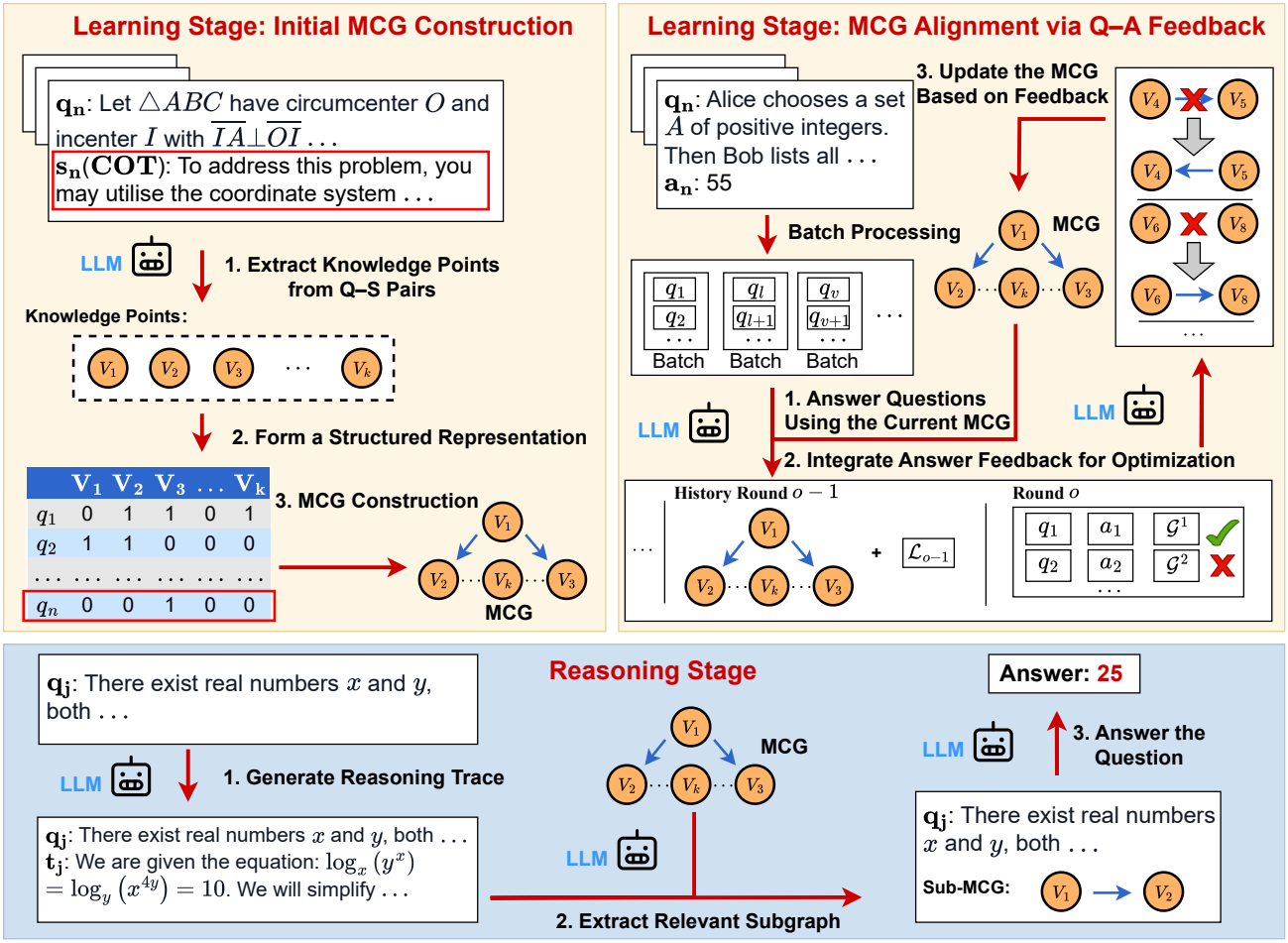

Figure 1: The CAMA framework consists of two stages: learning and reasoning. In the learning stage, (1) CAMA constructs an initial Mathematical Causal Graph (MCG) from question–solution pairs by combining LLM outputs with classical causal discovery methods to identify key knowledge points and their causal dependencies; (2) the MCG is then refined using feedback from the LLM's answers to better align with the downstream reasoning task. In the reasoning stage, the optimized MCG is used to solve new questions through a three-step process: generating a reasoning trace, extracting a relevant subgraph, and guiding the LLM to produce the final answer.

such as supervised fine-tuning, making it lightweight and adaptable.

Our main contributions are as follows:

- We propose the **M**athematical **C**ausal **G**raph (**MCG**), a reusable, high-level representation that captures the causal structure of mathematical solution strategies.
- We develop **CAMA**, a lightweight and plug-and-play framework that combines LLMs with causal discovery to automatically derive and utilize MCGs for reasoning-enhanced problem solving.
- We empirically demonstrate that CAMA outperforms standard prompting methods on real-world datasets. In particular, structured, directed guidance via MCGs leads to more reliable and accurate solutions than unstructured or symmetric alternatives.

Related Works

Causal Discovery

Identifying causal relationships is a fundamental problem across many empirical sciences. Traditionally, such relationships are inferred through interventions or randomized controlled experiments. However, these approaches are often costly, time-consuming, or even infeasible in practice. As an alternative, causal discovery algorithms aim to infer a causal graph, such as a directed acyclic graph (DAG), from passively collected observational data that are typically easier to obtain. A wide range of causal discovery methods have been developed, falling into several main categories: constraint-based methods (*e.g.*, PC, FCI) (Spirtes, Glymour, and Scheines 2000; Spirtes, Meek, and Richardson 1995), which rely on conditional independence tests; noise-based methods (*e.g.*, LiNGAM) (Shimizu et al. 2011; Peters, Janzing, and Schölkopf 2017), which exploit asym-

metric patterns of noise in the data; score-based methods (*e.g.*, GES) (Chickering 2002b), which search for high-scoring graph structures; and optimization-based methods (*e.g.*, NOTEARS) (Zheng et al. 2018), which frame DAG learning as a continuous optimization problem. For a comprehensive review of these methods, please refer to these surveys (Assaad, Devijver, and Gaussier 2022; Glymour, Zhang, and Spirtes 2019). In addition, several practical Python libraries have emerged to support causal discovery (Zhang et al. 2021; Zheng et al. 2024). Despite their theoretical appeal, practitioners should remain mindful that each method's assumptions may not always hold in complex environments (Aït-Bachir et al. 2023). For example, the PC algorithm (Spirtes, Glymour, and Scheines 2000) assumes *faithfulness* and *causal sufficiency* (Spirtes, Glymour, and Scheines 2000), meaning that all statistical independencies within the observed data are encoded in the causal graph and there are no unobserved hidden common causes. Moreover, some methods, such as PC and GES, typically recover only a Markov equivalence class (MEC) of DAGs, which includes all graphs that encode the same set of conditional independencies. These equivalence classes are commonly represented by a completed partially directed acyclic graph (CPDAG) (Chickering 2002a).

Recently, the rapid advancement of LLMs and the availability of vast amounts of unstructured data, particularly text, have drawn significant attention to leveraging LLMs to extract causal concepts and identify the relationships among them (Schölkopf et al. 2021; Liu et al. 2024; Wang et al. 2025).

Reinforcing Mathematical Reasoning Ability of LLMs

Improving the mathematical reasoning ability of LLMs has recently attracted significant attention, with advancements emerging from both pre-training and post-training strategies. Math-specific pre-trained models such as Llemma (Azerbayev et al. 2023), DeepSeekMath (Shao et al. 2024), InternLM-Math (Ying et al. 2024), and Qwen2-Math (Yang et al. 2024) enhance performance by training on curated, math-rich corpora. These models benefit from data sourced from scientific texts, programming content, and formal mathematical proofs, enabling them to develop a deeper domain understanding. In addition, post-training techniques further refine reasoning through supervised fine-tuning on specialized datasets. Program-of-Thought (PoT) (Chen et al. 2022), evol-Instruct (Luo et al. 2023), and Tool-Integrated Reasoning (TIR) (Gou et al. 2023) are representative approaches that teach models to solve problems step-by-step, often by generating executable code or integrating external tools such as Python for reliable calculations. Models such as WizardMath (Luo et al. 2023) and MetaMath (Yu et al. 2023) exemplify how instruction tuning and reasoning format alignment significantly improve math benchmark performance. Beyond supervised learning, preference-based methods like Step-DPO (Lai et al. 2024) and online RLHF (Wang et al. 2025) focus on optimizing the reasoning process itself by learning from step-level feedback. These strategies train models to prefer correct intermediate steps, rather than only focusing on final answers, resulting in more robust and interpretable solutions. While these efforts have led to substantial improvements, they primarily rely on statistical learning or preference signals.

The CAMA Framework

Problem Setup

Dataset Description We are given a dataset $\mathbf{D} = \{(q_i, s_i, a_i)\}_{i=1}^n$ of n independently drawn triples from the product space $\mathbf{Q} \times \mathbf{S} \times \mathbf{A}$, where $q_i \in \mathbf{Q}$ is an unstructured question, $s_i \in \mathbf{S}$ is the corresponding detailed solution, and $a_i \in \mathbf{A}$ is the ground-truth answer (integer or symbolic).

Learning Objective The goal of CAMA is to learn a mapping

$$f : (\mathbf{Q} \times \mathbf{S} \times \mathbf{A})^n \to \mathbf{G}, \qquad f(\{(q_i, s_i, a_i)\}_{i=1}^n) = \mathcal{G}.$$

However, most existing mathematical datasets do not provide detailed solution steps. To address this limitation, we leverage an LLM to generate a chain-of-thought solution s_i for each question q_i, thereby constructing the solution set $\{s_i\}_{i=1}^n$.

We denote the Mathematical Causal Graph (MCG) as $\mathcal{G} = (\mathbf{V}, \mathbf{E}) \in \mathbf{G}$, with its components described in detail in the following section. In this graph, $\mathbf{V} = \{V_1, \ldots, V_k\}$ is the set of k nodes, each corresponding to a distinct mathematical knowledge point (*e.g.*, *Volume of a cylinder*, *Area of a circle*), and $\mathbf{E}$ is the set of edges capturing the causal relationships among these knowledge points.

To construct $\mathcal{G}$, we first learn an intermediate mapping

$$h : (\mathbf{Q} \times \mathbf{S})^n \to \{0,1\}^{n \times k}, \qquad h(\{(q_i, s_i)\}_{i=1}^n) = \mathbf{Z},$$

where $\mathbf{Z} \in \{0,1\}^{n \times k}$. Each row corresponds to a question–solution pair, and each column to a knowledge point. The element $Z_{i,j} \in \{0,1\}$ indicates whether knowledge point V_j is required to solve question q_i: $Z_{i,j} = 1$ if the knowledge point V_j is necessary for solving question q_i; otherwise $Z_{i,j} = 0$. In practice, $\mathbf{Z}$ is obtained through a multi-step pipeline consisting of knowledge point extraction, deduplication, and parsing (via the function $l(\cdot)$ defined later). Thus, the mapping $h(\cdot)$ should be regarded as a composite procedure rather than a separate model. To extract and formalize the set of knowledge points $\mathbf{V} = \{V_1, \ldots, V_k\}$ from the input pairs $\{(q_i, s_i)\}_{i=1}^n$, we leverage the language understanding and domain knowledge capabilities of the LLM. Once the binary matrix $\mathbf{Z}$ is obtained, a causal discovery algorithm (denoted by CD) is applied to infer the structure of the graph. This graph is then iteratively refined using feedback from a selected subset of question–answer pairs, allowing it to better align with downstream reasoning tasks.

Mathematical Causal Graph (MCG) We represent high-level solution strategies for mathematical problems using a Mathematical Causal Graph (MCG), denoted as $\mathcal{G} = (\mathbf{V}, \mathbf{E})$. In this graph, nodes correspond to relevant theorems or definitions, while edges capture causal relationships and the potential order in which these knowledge points are applied during problem solving. For simplicity and without

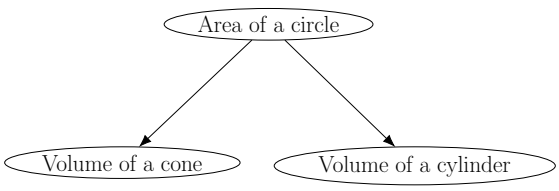

Figure 2: An example of a Mathematical Causal Graph (MCG) is shown, illustrating three knowledge points: *Area of a circle*, *Volume of a cylinder*, and *Volume of a cone*. The edges indicate that understanding the *Area of a circle* is required to compute both the *Volume of a cylinder* and the *Volume of a cone*.

loss of generality, we assume that the MCG is a directed acyclic graph. Figure 2 illustrates an example MCG with three knowledge points: *Area of a circle*, *Volume of a cylinder*, and *Volume of a cone*. A directed edge from *Area of a circle* to *Volume of a cylinder* (and similarly to *Volume of a cone*) indicates that computing the area of the circle is a prerequisite for determining the cylinder's volume. More generally, each edge represents a possible (though not necessarily unique) reasoning path.

Mathematical Dataset Construction To construct the Mathematical Causal Graph (MCG), it is crucial to curate the dataset $\mathbf{D} = \{(q_i, s_i, a_i)\}_{i=1}^n$. We begin with a readily available dataset $\mathbf{QA} = \{(q_i, a_i)\}_{i=1}^n$. For each question q_i, we employ an LLM (denoted $\mathcal{LLM}$) together with a prompt template p_g to generate a detailed chain-of-thought solution s_i and a predicted answer $\hat{a}_i$:

$$s_i, \hat{a}_i = \mathcal{LLM}(q_i, p_g).$$

The solution s_i is retained and paired with q_i only if the predicted answer matches the ground truth, *i.e.*, $\hat{a}_i = a_i$. By concatenating each retained solution with its corresponding question, we obtain the set of question–solution pairs $\mathbf{QS} = \{(q_i, s_i)\}_{i=1}^n$.

Overview of the CAMA Framework

This section introduces the two-stage CAMA framework, illustrated in Figure 1, comprising a learning stage and a reasoning stage. In the learning stage, CAMA first builds an initial Mathematical Causal Graph (MCG) from a set of question–solution pairs and then refines it using feedback from a downstream reasoning task. In the reasoning stage, a new question is solved in three sequential steps guided by the MCG, ultimately producing the final answer.

Learning Stage

The learning stage of the CAMA framework consists of two key components: (1) constructing an initial Mathematical Causal Graph (MCG) from question–solution pairs, and (2) refining the graph by aligning it with the reasoning task through LLM-based feedback.

Initial MCG Construction This step involves three main sub-tasks: extracting knowledge points from question–solution pairs, forming a structured representation, and applying causal discovery to infer the initial MCG. The input consists of n pairs: $\mathbf{QS} = \{(q_i, s_i)\}_{i=1}^n$. For each pair (q_i, s_i), $\mathcal{LLM}$ is prompted with p_p to extract up to λ relevant knowledge points:

$$\mathbf{V}^i = \mathcal{LLM}(q_i, s_i, \lambda, p_p).$$

Each knowledge point $V_j^i \in \mathbf{V}^i$ is a tuple: $V_j^i = (V_{j,key}^i, V_{j,des}^i)$, where $V_{j,key}^i$ is a short label, and $V_{j,des}^i$ is a detailed description. The parameter λ controls the granularity of extracted knowledge points: smaller λ leads the LLM to produce broader, more general knowledge points, while larger λ allows for more specific and detailed extractions tailored to each question–solution pair. Given the sensitivity of LLMs to prompt phrasing and input order (Razavi et al. 2025; Mirzadeh et al. 2025), knowledge extraction is performed independently for each question to avoid prompt interference. This results in a set of extracted knowledge points for each item in the dataset. The union of all extracted sets is:

$$\mathbf{V}' = \bigcup_{i=1}^{n} \mathbf{V}^i.$$

The aggregated set $\mathbf{V}'$ may contain redundant or semantically overlapping entries. To address this, we employ another $\mathcal{LLM}$ with prompt template p_r to identify and remove redundant knowledge points:

$$\mathbf{V}, \mathbf{O} = \mathcal{LLM}(\mathbf{V}', p_r),$$

where $\mathbf{V}$ is the deduplicated set, and each pair $(V_i, V_j) \in \mathbf{O}$ indicates that V_j, an element in $\mathbf{V}' \setminus \mathbf{V}$, can be replaced by $V_i \in \mathbf{V}$. Finally, we reconstruct the structured representation $\mathbf{Z}$ by parsing the original extracted sets $\{\mathbf{V}^1, \ldots, \mathbf{V}^n\}$ with $\mathbf{V}$ using the replacement mapping $\mathbf{O}$. Specifically, each row of $\mathbf{Z}$ corresponds to a question–solution pair (q_i, s_i), and each column represents a knowledge point in $\mathbf{V}$. A column is set to 1 if the corresponding knowledge point either appears in $\mathbf{V}^i$ or replaces an element of $\mathbf{V}^i$ according to $\mathbf{O}$; otherwise, it is set to 0. This parsing process is defined by the function $l(\cdot)$:

$$\mathbf{Z} = l(\{\mathbf{V}^1, \ldots, \mathbf{V}^n\}, \mathbf{V}, \mathbf{O}).$$

After obtaining the binary matrix $\mathbf{Z}$ from the question–solution pairs $\mathbf{QS}$, we apply a causal discovery method (CD), to infer the MCG:

$$\mathcal{G} = \mathrm{CD}(\mathbf{Z}).$$

In our experiments, we employ the PC algorithm, which yields a completed partially directed acyclic graph (CPDAG) comprising both directed and undirected edges. The resulting edges should be viewed as dataset-level averages, links appearing in only a few instances are generally pruned. This method relies on two standard assumptions: *causal sufficiency* — no unobserved knowledge point acts as a common prerequisite for any pair in $\mathbf{V}$, and *faithfulness* — the inferred graph $\mathcal{G}$ captures all dependencies among the knowledge points. Although we adopt PC here for concreteness, our framework is agnostic to the specific causal discovery method and can accommodate alternatives such as FCI, provided that their assumptions align with the data.

MCG Alignment via Question–Answer Feedback To adapt the edge set $\mathbf{E}$ of the MCG $\mathcal{G}$ to downstream reasoning tasks, we iteratively update $\mathcal{G}$ with feedback from $\mathcal{LLM}$, which generates answers conditioned on the current graph. The alignment aims to maximize $\mathcal{LLM}$'s precision on a designated subset of question–answer pairs. Formally we seek

$$\mathcal{G}^* = \arg\max_{\mathcal{G}} \ \mathbb{E}_{(q_i,s_i,a_i)\sim\mathbf{D}^s}\left[\mathbf{1}_{\{\mathcal{LLM}(\mathcal{G},q_i,p_a)=a_i\}}\right],$$

where $\mathbf{D}^s = \{(q_i, s_i, a_i)\}_{i=1}^m$ denotes a subset of m samples drawn from $\mathbf{D}$, for each question q_i the model uses $\mathcal{G}$ through the prompt template p_a and returns

$$\hat{a}_i = \mathcal{LLM}(\mathcal{G}, q_i, p_a),$$

the indicator function $\mathbf{1}_{\{\cdot\}}$ outputs 1 when $\hat{a}_i$ matches the ground truth a_i and 0 otherwise.

To solve this problem, we adopt a batch-based iterative optimization procedure over n_e epochs. At each optimization round o, we draw a batch of s_b samples from $\mathbf{D}^s$:

$$\mathbf{D}'_o = \{(q_j, s_j, a_j)\}_{j=1}^{s_b} \subset \mathbf{D}^s.$$

Instead of using the full graph $\mathcal{G}_o$, we extract a subgraph $\mathcal{G}_o^j \subseteq \mathcal{G}_o$ for each question q_j that includes only relevant knowledge points. To do this, we first ask the $\mathcal{LLM}$ using prompt template p_t to analyze the question and produce a candidate chain-of-thought solution:

$$t_j = \mathcal{LLM}(q_j, p_t).$$

We then use the generated reasoning trace t_j along with q_j and the full graph $\mathcal{G}_o$ to generate the matched subgraph using prompt p_m:

$$\mathcal{G}_o^j = \mathcal{LLM}(\mathcal{G}_o, t_j, q_j, p_m).$$

The model is then queried as follows:

$$\hat{a}_j = \mathcal{LLM}(\mathcal{G}_o^j, q_j, p_a).$$

We record the quadruple $(q_j, s_j, \mathcal{G}_o^j, \mathbf{1}_{\{\hat{a}_j=a_j\}})$ and repeat the process for all s_b questions.

After the batch is finished, these quadruples together with an optimization history over r previous rounds:

$$\mathbf{H} = \{(\mathcal{G}_{o-1}, \mathcal{L}_{o-1}), \ldots, (\mathcal{G}_{o-r}, \mathcal{L}_{o-r})\}$$

are given to the model through the update prompt p_u to obtain the next graph:

$$\mathcal{G}_{o+1} = \mathcal{LLM}(\{(q_j, s_j, \mathcal{G}_o^j, \mathbf{1}_{\{\hat{a}_j=a_j\}})\}_{j=i}^{s_b}, \mathbf{H}, p_u),$$

Edges that support correct reasoning are reinforced, whereas those linked to errors are revised. Here, $\mathcal{L}_{o-1}$ represents the precision achieved in the previous round with graph $\mathcal{G}_{o-1}$:

$$\mathcal{L}_{o-1} = \frac{1}{s_b}\sum_{j=1}^{s_b} \mathbf{1}_{\{\hat{a}_j=a_j\}}.$$

At the end of each epoch, we evaluate the precision of $\mathcal{G}_{o+1}$ over $\mathbf{D}^s$. After all epochs, the graph with the highest precision on $\mathbf{D}^s$ is selected as the final optimized graph $\mathcal{G}^*$. To improve efficiency, the optimization process is terminated early if the graph remains unchanged for c_{stop} consecutive batches. We further assume that MCGs derived from the alignment and downstream samples share partial structural overlap, enabling strategy preferences learned during alignment to transfer effectively to downstream reasoning.

Reasoning Stage

In the reasoning stage, a new question is processed in three successive steps using the MCG $\mathcal{G}^*$, ultimately yielding the final answer. This procedure mirrors the alignment process used during training.

Generate Reasoning Trace Given a new question q_j, we first prompt $\mathcal{LLM}$ using a template p_t to produce a candidate chain-of-thought solution:

$$t_j = \mathcal{LLM}(q_j, p_t).$$

Extract Relevant Subgraph Next, the generated reasoning trace t_j is used, along with the question q_j and the full graph $\mathcal{G}^*$, to extract a relevant subgraph via the prompt template p_m:

$$\mathcal{G}^j = \mathcal{LLM}(\mathcal{G}^*, t_j, q_j, p_m).$$

Answer the Question Finally, the model is prompted with the matched subgraph $\mathcal{G}^j$, the question q_j, and the answering prompt p_a to generate the predicted answer:

$$\hat{a}_j = \mathcal{LLM}(\mathcal{G}^j, q_j, p_a).$$

To enable effective reasoning, in practice, the subgraph $\mathcal{G}^j$ is encoded into natural language. Each edge between knowledge points is verbalized. For instance, a directed edge from *Area of a circle* to *Volume of a cylinder* is expressed as: "*Area of a circle* is a prerequisite for *Volume of a cylinder*. If *Volume of a cylinder* is used, then *Area of a circle* could also be used." If the edge is undirected, we phrase it as: "*Area of a circle* and *Volume of a cylinder* are associated, but the direction of dependency is unclear. Either could be a prerequisite for the other." This verbalization follows directly from the way we build the structured matrix $\mathbf{Z}$. For each question–solution pair, a knowledge point's presence is encoded as 1 and its absence as 0. A directed edge from one knowledge point to another therefore means the source point always appears whenever the target does, allowing us to interpret the source as a prerequisite for the target.

Experiments

To assess the effectiveness of CAMA in enhancing the LLM's mathematical reasoning, we conduct the following experiments.

Datasets We evaluate our framework on three mathematical benchmarks: AIME, Omni-MATH (Gao et al. 2024), and OlympiadBench (He et al. 2024). The American Invitational Mathematics Examination (AIME) is a high-level competition for high school students, with 30 questions per year spanning four categories: algebra, geometry, number theory, and combinatorics. Omni-MATH is a recent Olympiad-style benchmark containing 4,428 problems across nine subdomains. For our experiments, we use a 200-question subset, referred to as Omni-MATH-200, which focuses on the same four categories. OlympiadBench includes 8,476 international Olympiad problems. We evaluate on a filtered subset of 674 English, non-proof questions, called OlympiadBench-674, which is also grouped into the same four categories.

Evaluation Setup and Compared Methods We adopt DeepSeek-V3-0324 (DeepSeek-AI et al. 2025b) (**DSV3**) and Qwen3-32B (Yang et al. 2025) (**Qwen3**) as our base LLMs, adopting a sampling temperature of 0.6 following prior work (DeepSeek-AI et al. 2025a). DSV3 is accessed via API, whereas Qwen3 is deployed locally. The version enhanced with our proposed CAMA framework is denoted **CAMA**. Unless otherwise specified, we set the knowledge point granularity to $\lambda = 3$, batch size to $s_b = 5$, number of epochs to $n_e = 10$, history length to $r = 7$, and early stopping threshold to $c_{stop} = 3$. We also evaluate two ablated variants: **CAMA *w/o* Alignment**, which excludes the alignment stage, and **CAMA *w/o* Directed Edge**, in which all directed edges in the MCG are replaced with undirected edges. For comparison, we include two baselines based on the chain-of-thought prompting (Wei et al. 2022): **COT-ZeroShot**, which uses no examples, and **COT-FewShot**, which uses the full AIME2023 dataset along with detailed solutions as in-context examples (due to input token length limitations, AIME2022 questions are excluded). To construct the MCG, we apply the PC algorithm from the `gCastle` library (Zhang et al. 2021) with the G-squared test and default settings.

Evaluation Metric Model performance is evaluated with the Pass@1 metric, defined as the fraction of test questions answered correctly on the first attempt. A prediction is considered correct if it exactly matches or is mathematically equivalent to the ground truth. Formally, given a test set $\mathbf{Q} = \{q_1, \ldots, q_n\}$, let $\hat{a}_i$ be the model's answer and a_i the ground truth for q_i. An indicator function $\text{judge}(\hat{a}_i, a_i)$ returns 1 if the answer is correct, and 0 otherwise, leading to

$$\text{Pass@1} = \frac{1}{n}\sum_{i=1}^{n} \text{judge}(\hat{a}_i, a_i).$$

Experimental Setup To construct the MCG, we use a training set of 60 questions sourced from AIME2022 and AIME2023. We prompt DeepSeek-R1 (DeepSeek-AI et al. 2025a) to produce detailed chain-of-thought solutions for each question. Alignment is performed using the same data as the training set. Model performance is then evaluated on four test sets: AIME2024, AIME2025, Omni-MATH-200, and OlympiadBench-674. Given that answers in the AIME datasets are purely numeric, correctness is evaluated using a strict equality check: $\text{judge}(\hat{a}_i, a_i) = 1$ if and only if $\hat{a}_i = a_i$. For Omni-MATH-200 and OlympiadBench-674, where answers may include symbolic expressions, we employ Omni-Judge (Gao et al. 2024), an evaluator built on LLaMA-3.1-8B-Instruct, to assess correctness.

Results Table 1 presents the mean Pass@1 scores for each method on four benchmark datasets using two base LLMs, with each result averaged over three repetitions. Overall, **CAMA** achieves the strongest performance in most cases. The comparisons across method variants reveal several key observations. First, **CAMA** consistently outperforms **COT-FewShot**, indicating that structured information encoded in the MCG is more effective for enhancing mathematical reasoning than raw text prompting. Second, the comparison between **CAMA** and **CAMA w/o Directed Edge** demonstrates the importance of asymmetric (directed) relations in the MCG over purely symmetric ones, confirming the value of capturing causal directionality. Third, the comparison with **CAMA w/o Alignment** underscores the benefit of the alignment step in refining the graph and adapting it more closely to the LLM's reasoning needs. Finally, the relatively moderate performance of **CAMA** based on **DSV3** on OlympiadBench-674 can be partially attributed to limited utilization of the benchmark's knowledge components in the MCG. This limitation can be eased by adopting a coarser knowledge point granularity: with $\lambda = 2$, **CAMA** attains a Pass@1 score of 67%, nearly matching the 67.2% achieved by the **COT-ZeroShot**.

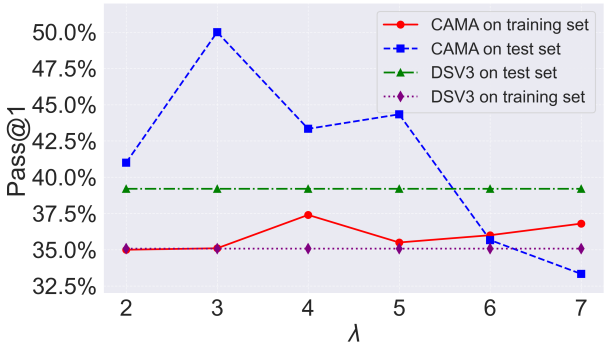

Figure 3: This figure shows the Pass@1 scores of **CAMA** using MCGs built with different knowledge point granularities, controlled by the parameter λ (ranging from 2 to 7). Each λ produces a distinct MCG from the AIME2022 and AIME2023 training data, and results are averaged over three repetitions. Performance is reported on both the training set (AIME2022 and AIME2023) and the test set (AIME2024). The base LLM is **DSV3**, with its scores included as references, shown with purple diamonds for the training sets and a green triangle for the test set.

Impact of Knowledge Point Granularity λ In this part, we investigate how the granularity of knowledge points, controlled by λ, influences the performance of **CAMA**. For this experiment, we use AIME2022 and AIME2023 as the training set and AIME2024 as the test set. The value of λ is varied from 2 to 7 in increments of 1. For each setting, an MCG is constructed from the training data and then reused to answer both training and test questions. Figure 3 reports the Pass@1 scores of **CAMA** based on **DSV3** on both sets, with each result averaged over three repetitions. The results reveal an interesting finding: as λ increases, performance improves on the training set but declines on the test set. This suggests a trade-off between generalization and specificity of the extracted knowledge points. When λ is small, the extracted knowledge points are coarser and more broadly applicable across datasets. In contrast, larger λ values lead to finer-grained, more detailed knowledge that better fits the training questions but may overfit, reducing robustness on

Base LLM	Method	AIME2024 Pass@1(↑)	AIME2025 Pass@1(↑)	Omni-MATH-200 Pass@1(↑)	OlympiadBench-674 Pass@1(↑)
DSV3	**COT-ZeroShot**	42.2%	35.6%	42.2%	**67.2%**
	COT-FewShot	43.3%	31.0%	41.0%	65.2%
	DSV3	39.2%	28.8%	42.0%	65.0%
	CAMA *w/o* Directed Edge	43.3%	33.3%	43.7%	65.8%
	CAMA *w/o* Alignment	47.8%	**38.9%**	43.0%	66.8%
	CAMA (ours)	**50.0%**	**38.9%**	**45.0%**	66.4%
Qwen3	**COT-ZeroShot**	75.6%	72.2%	68.7%	81.4%
	COT-FewShot	66.7%	61.1%	58.5%	77.1%
	Qwen3	74.4%	65.6%	64.2%	81.2%
	CAMA *w/o* Directed Edge	75.6%	72.2%	67.0%	82.0%
	CAMA *w/o* Alignment	78.9%	74.4%	67.3%	82.2%
	CAMA (ours)	**82.2%**	**76.7%**	**69.0%**	**83.3%**

Table 1: This table reports the mean Pass@1 scores for each method on four datasets using DeepSeek-V3-0324 (**DSV3**) and Qwen3-32B (**Qwen3**), with each result averaged over three repetitions. Omni-MATH-200 is a 200-question subset of Omni-MATH covering four categories. Likewise, OlympiadBench-674 denotes a 674-question English, non-proof subset of OlympiadBench. Bold values indicate better performance.

unseen problems. Notably, the λ that maximizes training performance is not necessarily optimal for the test set.

Case Study We apply **CAMA** with **DSV3** and focus on Problem 14 from Exam II of AIME2024, which states:

> Let $b \geq 2$ be an integer. Call a positive integer n *b-eautiful* if it has exactly two digits when expressed in base b, and these two digits sum to $\sqrt{n}$. For example, 81 is 13-eautiful because $81 = \underline{63}_{13}$ and $6+3 = \sqrt{81}$. Find the least integer $b \geq 2$ for which there are more than ten b-eautiful integers.

The reasoning trace produced by **DSV3** matches two knowledge points in the MCG: *Modular arithmetic for integer solutions* and *Quadratic polynomial systems*, with the former serving as a prerequisite for the latter. This dependency plays an important role in guiding **DSV3**'s problem-solving approach. **DSV3** first represents the two-digit number in base b as $a_1 \cdot b + a_0$, where $1 \leq a_1 \leq b-1$ and $0 \leq a_0 \leq b-1$, and imposes the constraint $(a_1 + a_0)^2 = a_1 \cdot b + a_0$. Leveraging *Modular arithmetic for integer solutions*, **DSV3** introduces $s = a_1 + a_0$, leading to the equation $a_1 = \frac{s(s-1)}{b-1}$. Since s and $s-1$ are consecutive integers and hence coprime, this implies that the denominator $b-1$ should factor into a coprime pair (d, e) such that $s \equiv 0 \pmod{d}$ and $s - 1 \equiv 0 \pmod{e}$. Subsequently, within the framework of *Quadratic polynomial systems*, **DSV3** invokes the Chinese Remainder Theorem, and **DSV3** deduces that each coprime pair (d, e) yields a distinct solution for s that repeats every $b-1$ values. Since the number of such coprime factorizations (d, e) is 2^γ, where γ is the number of distinct prime factors of $b-1$, **DSV3** continues with additional derivations and ultimately returns the correct solution 211. However, without the incorporation of this prior knowledge, **DSV3** does not apply modular arithmetic and the Chinese Remainder Theorem, but instead resorts to an exhaustive brute-force search over possible digit pairs and ultimately returns an incorrect answer. Moreover, modular operations are generally essential for solving constrained quadratic polynomial functions, highlighting their broader relevance beyond this specific context.

Conclusions

In this paper, we present the Mathematical Causal Graph (MCG) as a high-level representation of reusable solution strategies for mathematical problems and introduce CAusal MAthematician (CAMA), a plug-and-play framework that learns and exploits these graphs. CAMA extracts knowledge points from chain-of-thought solutions generated by a large language model, infers the causal relations among them, and refines the resulting graph with feedback from its own question-answering accuracy. Experiments on real-world benchmarks show that incorporating structured guidance consistently outperforms unstructured alternatives, and that incorporating asymmetric causal relationships yields greater improvements than relying on symmetric associations alone.

Although our proposed framework shows significant improvement, it still has some limitations. CAMA's effectiveness depends on the granularity parameter λ, which controls how knowledge points are extracted. The optimal value of λ on training data may not transfer well to unseen benchmarks, posing challenges for practical deployment. Future work could explore strategies for automatically selecting an appropriate λ for new tasks, potentially with assistance from the LLM itself. Finally, the MCG remains static during the reasoning stage. Enabling CAMA to dynamically update the graph while answering questions, particularly by adding novel knowledge points not observed during training, could improve both its coverage and robustness.

References

Aït-Bachir, A.; Assaad, C. K.; de Bignicourt, C.; Devijver, E.; Ferreira, S.; Gaussier, E.; Mohanna, H.; and Zan, L. 2023. Case studies of causal discovery from it monitoring time series. *arXiv preprint arXiv:2307.15678*.

Assaad, C. K.; Devijver, E.; and Gaussier, E. 2022. Survey and evaluation of causal discovery methods for time series. *Journal of Artificial Intelligence Research*, 73: 767–819.

Azerbayev, Z.; Schoelkopf, H.; Paster, K.; Santos, M. D.; McAleer, S.; Jiang, A. Q.; Deng, J.; Biderman, S.; and Welleck, S. 2023. Llemma: An open language model for mathematics. *arXiv preprint arXiv:2310.10631*.

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

Chickering, D. M. 2002a. Learning equivalence classes of Bayesian-network structures. *Journal of machine learning research*, 2(Feb): 445–498.

Chickering, D. M. 2002b. Optimal structure identification with greedy search. *Journal of machine learning research*, 3(Nov): 507–554.

Cobbe, K.; Kosaraju, V.; Bavarian, M.; Chen, M.; Jun, H.; Kaiser, L.; Plappert, M.; Tworek, J.; Hilton, J.; Nakano, R.; et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Dai, S.; Xu, C.; Xu, S.; Pang, L.; Dong, Z.; and Xu, J. 2024. Bias and unfairness in information retrieval systems: New challenges in the llm era. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 6437–6447.

DeepSeek-AI et al. 2025a. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. arXiv:2501.12948.

DeepSeek-AI et al. 2025b. DeepSeek-V3 Technical Report. arXiv:2412.19437.

Gao, B.; Song, F.; Yang, Z.; Cai, Z.; Miao, Y.; Dong, Q.; Li, L.; Ma, C.; Chen, L.; Xu, R.; et al. 2024. Omni-math: A universal olympiad level mathematic benchmark for large language models. *arXiv preprint arXiv:2410.07985*.

Glymour, C.; Zhang, K.; and Spirtes, P. 2019. Review of causal discovery methods based on graphical models. *Frontiers in genetics*, 10: 524.

Gou, Z.; Shao, Z.; Gong, Y.; Shen, Y.; Yang, Y.; Huang, M.; Duan, N.; and Chen, W. 2023. Tora: A tool-integrated reasoning agent for mathematical problem solving. *arXiv preprint arXiv:2309.17452*.

Gu, Q. 2023. Llm-based code generation method for golang compiler testing. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2201–2203.

Han, V. T. Y.; Puduppully, R.; and Chen, N. F. 2023. Veritymath: Advancing mathematical reasoning by self-verification through unit consistency. *arXiv preprint arXiv:2311.07172*.

He, C.; Luo, R.; Bai, Y.; Hu, S.; Thai, Z. L.; Shen, J.; Hu, J.; Han, X.; Huang, Y.; Zhang, Y.; Liu, J.; Qi, L.; Liu, Z.; and Sun, M. 2024. OlympiadBench: A Challenging Benchmark for Promoting AGI with Olympiad-Level Bilingual Multimodal Scientific Problems. arXiv:2402.14008.

Jiang, B.; Xie, Y.; Hao, Z.; Wang, X.; Mallick, T.; Su, W. J.; Taylor, C. J.; and Roth, D. 2024. A peek into token bias: Large language models are not yet genuine reasoners. *arXiv preprint arXiv:2406.11050*.

Lai, X.; Tian, Z.; Chen, Y.; Yang, S.; Peng, X.; and Jia, J. 2024. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*.

Liu, C.; Chen, Y.; Liu, T.; Gong, M.; Cheng, J.; Han, B.; and Zhang, K. 2024. Discovery of the Hidden World with Large Language Models. In Globerson, A.; Mackey, L.; Belgrave, D.; Fan, A.; Paquet, U.; Tomczak, J.; and Zhang, C., eds., *Advances in Neural Information Processing Systems*, volume 37, 102307–102365. Curran Associates, Inc.

Liu, J.; Lin, J.; and Liu, Y. 2024. How much can rag help the reasoning of llm? *arXiv preprint arXiv:2410.02338*.

Luo, H.; Sun, Q.; Xu, C.; Zhao, P.; Lou, J.; Tao, C.; Geng, X.; Lin, Q.; Chen, S.; and Zhang, D. 2023. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. *arXiv preprint arXiv:2308.09583*.

Merrill, W.; and Sabharwal, A. 2023. The parallelism tradeoff: Limitations of log-precision transformers. *Transactions of the Association for Computational Linguistics*, 11: 531–545.

Mirzadeh, I.; Alizadeh, K.; Shahrokhi, H.; Tuzel, O.; Bengio, S.; and Farajtabar, M. 2024. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. *arXiv preprint arXiv:2410.05229*.

Mirzadeh, I.; Alizadeh, K.; Shahrokhi, H.; Tuzel, O.; Bengio, S.; and Farajtabar, M. 2025. GSM-Symbolic: Understanding the Limitations of Mathematical Reasoning in Large Language Models. In *The Thirteenth International Conference on Learning Representations (ICLR)*. OpenReview. Poster.

OpenAI et al. 2024. GPT-4 Technical Report. arXiv:2303.08774.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744.

Peters, J.; Janzing, D.; and Schölkopf, B. 2017. *Elements of causal inference: foundations and learning algorithms*. The MIT Press.

Razavi, A.; Soltangheis, M.; Arabzadeh, N.; Salamat, S.; Zihayat, M.; and Bagheri, E. 2025. Benchmarking Prompt Sensitivity in Large Language Models. In Hauff, C.; Macdonald, C.; Jannach, D.; Kazai, G.; Nardini, F. M.; Pinelli, F.; Silvestri, F.; and Tonellotto, N., eds., *Advances in Information Retrieval*, 303–313. Cham: Springer Nature Switzerland. ISBN 978-3-031-88714-7.

Schölkopf, B.; Locatello, F.; Bauer, S.; Ke, N. R.; Kalchbrenner, N.; Goyal, A.; and Bengio, Y. 2021. Toward causal representation learning. *Proceedings of the IEEE*, 109(5): 612–634.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y.; Wu, Y.; et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Shimizu, S.; Inazumi, T.; Sogawa, Y.; Hyvarinen, A.; Kawahara, Y.; Washio, T.; Hoyer, P. O.; Bollen, K.; and Hoyer, P. 2011. DirectLiNGAM: A direct method for learning a linear non-Gaussian structural equation model. *Journal of Machine Learning Research-JMLR*, 12(Apr): 1225–1248.

Spirtes, P.; Glymour, C. N.; and Scheines, R. 2000. *Causation, prediction, and search*. MIT press.

Spirtes, P.; Meek, C.; and Richardson, T. 1995. Causal inference in the presence of latent variables and selection bias. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, UAI'95, 499–506. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN 1558603859.

Tang, Y.; Yin, Y.; Wang, Y.; Zhou, H.; Pan, Y.; Guo, W.; Zhang, Z.; Rang, M.; Liu, F.; Zhang, N.; et al. 2025. Pangu Ultra MoE: How to Train Your Big MoE on Ascend NPUs. arXiv:2505.04519.

Wang, X.; Zhou, K.; Wu, W.; Singh, H. S.; Nan, F.; Jin, S.; Philip, A.; Patnaik, S.; Zhu, H.; Singh, S.; et al. 2025. Causal-copilot: An autonomous causal analysis agent. *arXiv preprint arXiv:2504.13263*.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Xia, F.; Chi, E.; Le, Q. V.; Zhou, D.; et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35: 24824–24837.

Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Yang, A.; Zhang, B.; Hui, B.; Gao, B.; Yu, B.; Li, C.; Liu, D.; Tu, J.; Zhou, J.; Lin, J.; et al. 2024. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Yin, Y.; Huang, W.; Song, K.; Tang, Y.; Wu, X.; Guo, W.; Guo, P.; Wang, Y.; Meng, X.; Wang, Y.; et al. 2025. Pangu Ultra: Pushing the Limits of Dense Large Language Models on Ascend NPUs. arXiv:2504.07866.

Ying, H.; Zhang, S.; Li, L.; Zhou, Z.; Shao, Y.; Fei, Z.; Ma, Y.; Hong, J.; Liu, K.; Wang, Z.; et al. 2024. Internlm-math: Open math large language models toward verifiable reasoning. *arXiv preprint arXiv:2402.06332*.

Yu, L.; Jiang, W.; Shi, H.; Yu, J.; Liu, Z.; Zhang, Y.; Kwok, J. T.; Li, Z.; Weller, A.; and Liu, W. 2023. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*.

Zhang, K.; Zhu, S.; Kalander, M.; Ng, I.; Ye, J.; Chen, Z.; and Pan, L. 2021. gCastle: A Python Toolbox for Causal Discovery. arXiv:2111.15155.

Zheng, X.; Aragam, B.; Ravikumar, P. K.; and Xing, E. P. 2018. Dags with no tears: Continuous optimization for structure learning. *Advances in neural information processing systems*, 31.

Zheng, Y.; Huang, B.; Chen, W.; Ramsey, J.; Gong, M.; Cai, R.; Shimizu, S.; Spirtes, P.; and Zhang, K. 2024. Causal-learn: Causal discovery in python. *Journal of Machine Learning Research*, 25(60): 1–8.