Universiteit Leiden

Master Computer Science

Causal Reasoning in Pieces: Modular In-Context Discovery with Large Language Models

Name:	Kacper Kadziolka
Student ID:	4115945
Date:	27/06/2025
Specialisation:	Artificial Intelligence
1st supervisor:	Dr. Saber Salehkaleybar
2nd supervisor:	Dr. Matthijs van Leeuwen

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

Causal inference remains a fundamental challenge for large language models. Recent advances in internal reasoning with large language models have sparked interest in whether state-of-the-art reasoning models can robustly perform causal discovery—a task where conventional models often suffer from severe overfitting and near-random performance under data perturbations. On the CORR2CAUSE benchmark, we first evaluate conventional LLMs (*LLaMA3.3-70B* and *GPT4* series) under Chain-of-Thought, Tree-of-Thought, and self-evaluation prompting, and observe steep performance drops as causal graph complexity increases. We then study causal discovery with the emergent OpenAI's *o-series* and *DeepSeek-R* model families and find that these reasoning-first architectures achieve significantly greater native gains than prior approaches. To capitalize on these strengths, we introduce a modular in-context pipeline inspired by the Tree-of-Thoughts and Chain-of-Thoughts methodologies, yielding nearly three-fold improvements over conventional baselines. We further probe the pipeline's impact by analyzing reasoning chain length, complexity, and conducting qualitative and quantitative comparisons between conventional and reasoning models. Our findings suggest that while advanced reasoning models represent a substantial leap forward, carefully structured in-context frameworks are essential to maximize their capabilities and offer a generalizable blueprint for causal discovery across diverse domains.

1 Introduction

Causal inference—the ability to infer direct cause-effect relationships from observational data—underpins progress across the sciences, from epidemiology to economics. While large language models (LLMs) have recently demonstrated extraordinary performance in a variety of reasoning tasks (Brown et al., 2020; Wei et al., 2023), their capacity for *robust* causal discovery remains underdeveloped. Evaluations on CORR2CAUSE reveal that even fine-tuned models collapse under minimal perturbations (Jin et al., 2024) and often resort to superficial pattern matching rather than genuine causal inference (Zečević et al., 2023). This exposes a crucial gap between prompt-level heuristics and true causal reasoning.

The CORR2CAUSE benchmark[1] (Jin et al., 2024) provides a controlled evaluation framework in which each sample encodes conditional independencies derived from synthetic DAGs. Although fine-tuned classifiers achieve high in-distribution accuracy, their performance deteriorates sharply under subtle adversarial perturbations, such as paraphrasing or variable-identifier substitutions, while in-context and chain-of-thought prompting only yields marginal, non-robust improvement.

First, we conduct a broad survey of causal discovery and large language model domains, carefully tracing their foundations, commonly used algorithms, emergent LLM applications, advanced prompting techniques, and the latest reasoning-specialist architectures. Classical structure-learning methods such as Peter-Clark (PC) (Spirtes et al., 2001) and Greedy Equivalence Search (GES) (Chickering, 2002) algorithms provide strong statistical guarantees for recovering causal graphs from observational data (Rohrer, 2018). Prompting strategies like Chain-of-Thought (CoT) (Zhang et al., 2022) and Tree-of-Thought (ToT) (Yao et al., 2023)

[1] `https://huggingface.co/datasets/causal-nlp/corr2cause`

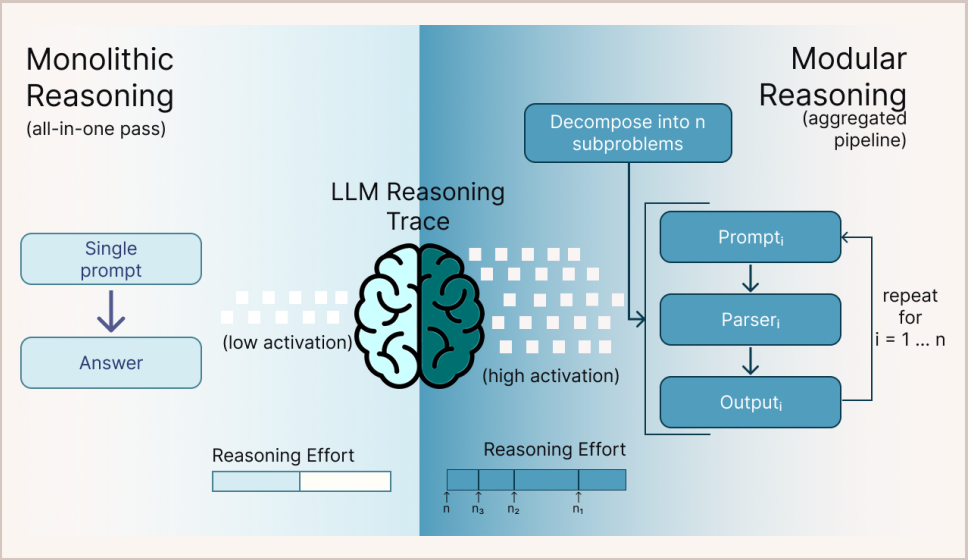


Figure 1: (Left) A single-prompt pass activates the model's internal reasoning once—shown as low activation, and consumes relatively few tokens (light bar denotes average token usage as reasoning effort). (Right) The modular pipeline splits the task into n prompt–parser–output stages; each stage re-activates the reasoning trace, yielding a cumulatively high activation and substantially greater total token usage (dark segmented bar). This richer, multi-stage reasoning increases overall reasoning effort and produces more robust answers than monolithic baseline.

have been applied to explore whether general-purpose LLMs can reproduce these results directly from synthetic independence statements, but their performance remains modest (Jin et al., 2024; Babu Shrestha et al., 2025). More recently, purpose-built "reasoning specialists" (e.g., *DeepSeek-R1* (DeepSeek-AI, 2025), *o3-mini* (OpenAI, 2025)) have been trained via reinforcement learning to perform multi-step inference and internal self-verification. Reviewing this progresses highlights both the strengths and the limitations of each approach and motivates our modular in-context pipeline, which pairs reasoning-specialist LLMs with a principled decomposition of the PC algorithm.

Prior to employing specialized reasoning models, we first benchmark general, conventional LLMs—including a self-hosted version of open-source—*LLaMA 3.3-70B* (Touvron et al., 2023) and the OpenAI's *GPT4* series (OpenAI et al., 2024) using three standard in-context prompting paradigms (Chain-of-Thought, Tree-of-Thought, and a single-iteration self-evaluation loop). As detailed in Section 5, these experiments measure `exact_match` ratio on the undirected skeleton[2] discovery stage in a zero-shot setting. The results reveal a steep decline in performance as graph complexity grows, highlighting the limitations of generic prompting strategies and motivating our transition to reasoning-specialist LLMs.

To address these shortcomings, we further evaluate the CORR2CAUSE benchmark with two state-of-the-art reasoning-specialist LLM families OpenAI's *o3-mini* and *DeepSeek-R1*, which exhibit internally disciplined inference processes via reinforcement learning self-evolution (DeepSeek-AI, 2025). By embedding a Peter-Clark causal discovery algorithm within a sin-

[2]The skeleton of a directed graph is an undirected graph resulted by removing the orientations of edges in the directed graph.

gle, unified prompt, we establish a zero-shot baseline that already surpasses all the previously reported results.

To further improve the performance, we propose a *modular in-context pipeline framework* tailored to reasoning-specialist LLMs. Rather than encoding all logic into a single monolithic prompt (Figure 1), our framework guides the model through a user-defined sequence of n subproblems, each issued as its own prompt and followed by a lightweight parser. Decomposing the reasoning task in this way yields progressively richer intermediate artifacts, sharpens the model's focus, and—by aggregating multiple sub-answers—transforms a brittle one-shot pass into a robust, multi-stage inference procedure that substantially improves accuracy.

In summary, we have developed a modular in-context pipeline tailored for PC-algorithm causal discovery as evaluated on the CORR2CAUSE benchmark. By decomposing the task into four focused prompts—undirected skeleton extraction, v-structure identification, Meek-rules orientation, and hypothesis evaluation—we achieve up to a three-fold F1 improvement without fine-tuning. While this design is optimized for causal-discovery tasks, exploring its adaptation to other structured-inference domains represents a promising avenue for future work. All code, prompt templates, and evaluation scripts are available at the following GitHub repository.[3]

Specifically, this thesis advances the field through the following key contributions:

1. We establish comprehensive baseline performance assessments for conventional LLMs (*LLaMA3.3-70B* and the OpenAI's *GPT-4* series) under widely used in-context prompting paradigms (Chain-of-Thought, Tree-of-Thought, and self-evaluation loops) on the challenging CORR2CAUSE causal discovery benchmark.

2. We conduct a detailed evaluation of reasoning-specialist LLM architectures (OpenAI's *o3-mini* and *DeepSeek-R1*), demonstrating significant native performance improvements over conventional baselines in zero-shot causal discovery scenarios.

3. We introduce a novel, modular in-context learning pipeline specifically tailored to reasoning-specialist LLMs. By decomposing the Peter Clark algorithm into a structured sequence of subprompts, this pipeline substantially improves robustness and accuracy, achieving nearly three-fold improvements in F1 scores without any additional fine-tuning.

4. We provide a rigorous qualitative and quantitative analysis of reasoning traces generated by reasoning-specialist models, clearly identifying iterative reasoning behaviors, trace complexity, and frequent re-evaluations as critical mechanisms underlying their performance improvements.

5. Finally, we release publicly accessible implementations, including all code, prompt templates, evaluation scripts, and documented inference environments, giving the community a reproducible blueprint for applying modular in-context frameworks to causal inference tasks and related structured reasoning domains.

These contributions collectively demonstrate the transformative potential of modular, structured in-context learning pipelines and open new avenues for further advancements in robust causal discovery using large language models.

[3] `https://github.com/kacperkadziolka/causal-reasoning-in-pieces`

The remainder of this thesis is organized as follows. Section 2 surveys related work, tracing the evolution from classical causal-discovery algorithms to recent prompting techniques and reasoning-specialist LLMs. Section 3 introduces the CORR2CAUSE benchmark, presents an exploratory data analysis, and states the research questions that guide the study. Section 5 describes how conventional LLMs are applied to the task, adapting Chain-of-Thought, Tree-of-Thought, and self-evaluation prompts to CORR2CAUSE and reporting their empirical performance. Section 6 proposes a modular in-context pipeline, beginning with the decomposition of the Peter-Clark algorithm and comparing baseline versus pipeline variants. Section 7 presents empirical evaluation, including results on the CORR2CAUSE benchmark, cross-model comparison, and an analysis contrasting conventional with reasoning-specialist models. Finally, Section 8 summarizes the main findings, discusses trade-offs, and outlines directions for future research.

2 Literature Review

This chapter surveys the causal discovery methods, from the classical algorithms to more recent LLM-based techniques. We begin in Section 2.1 with classical methods such as the Peter-Clark (PC) algorithm, which establish the theoretical foundations of our causal discovery framework. Section 2.2 then examines how general-purpose LLMs have been applied to causal inference tasks, highlighting both their promise and their current limitations. In Section 2.3, we review in-context prompting strategies designed to elicit better reasoning out of conventional LLMs. Section 2.4 focuses on the emergence of reasoning-specialist models that are explicitly trained for multi-step logical inference.

2.1 Overview of the Causal Discovery

Causal discovery—also known as *causal structure learning* (CSL) (Russo and Toni, 2025) is the task of inferring direct cause-and-effect relationships among variables from observational data and representing them as a graph (Woodward, 2003). Causal interpretations are an important factor of learning causal graphs: through data collection and validation, they establish the set of assumptions necessary for valid causal inference (Spirtes et al., 2001; Pearl, 2009; Peters et al., 2017). In this extensively studied domain, numerous settings and methods have been proposed to address its challenges (Zanga and Stella, 2023; Glymour et al., 2019; Vowels et al., 2021). Researchers typically classify these methods into three main methodologies: constraint-based, score-based and functional causal model (FCM)-based approaches (Nogueira et al., 2022).

In structural causal models (Pearl, 2009), the causal relationships are often represented as a directed acyclic graph (DAG) (Triantafillou and Tsamardinos, 2016). In a DAG, nodes correspond to variables and directed edges encode causal relationships (Geiger and Pearl, 1990). Causal discovery aims to infer the *existence* and *direction* of causal link (i.e., whether A causes B or vice versa) rather than merely quantifying the association between them (Lauritzen, 1996). Because observational data can be explained by multiple causal structures, the true underlying graph can be recovered up to an equivalence class (Pearl, 2009). Constraint-based methods recover the equivalence class by performing conditional independence tests (Koller and Friedman, 2009). A core concept in these methods is *d-separation*, a criterion determining whether

a set of variables Z blocks all paths between A and B in a causal graph (Pearl, 2010) (see Appendix A for a formal definition). If Z d-separates A and B, then A and B are conditionally independent given Z, under the causal Markov property (Uhler et al., 2013).

One of the earliest and most influential constraint-based algorithms is the Peter-Clark (PC) algorithm (Spirtes et al., 2001). Named after its originators, Peter Spirtes and Clark Glymour, the PC algorithm begins with a fully connected undirected graph over the variables and systematically removes edges based on conditional independence tests (Kalisch and Buehlmann, 2005; Harris and Drton, 2013). After pruning edges, it applies a set of orientation rules—commonly referred to as *Meek's rules*—to orient further undirected edges without introducing new cycles or v-structures (Meek, 2013, 2018)). The PC algorithm is provably sound and complete: it identifies all causal directions that are consistent with the data and outputs a Completed Partially Directed Acyclic Graph (CPDAG) representing the Markov equivalence class of the true DAG (Chickering, 2002).

Score-based methods form another branch of causal discovery, casting the search for the best-fitting DAG as an optimization problem over a scoring function (Malinsky and Danks, 2018). The Greedy Equivalence Search (GES) algorithm (Chickering, 2002) and its extensions (Ramsey et al., 2017; Huang et al., 2018; Tsamardinos et al., 2006) lay the foundations for score-base approaches. A parallel line of research reformulates the acyclicity constraint in continuous form (Zheng et al., 2018), leading to a substantial literature on a differentiable causal discovery methods (Vowels et al., 2021). Moreover, functional causal model (FCM) methods show that, under specific, stricter assumptions on the functional form and noise distribution of each variable, the true causal graph can be recovered uniquely from the observational data (Bühlmann et al., 2014). Other approaches exist in the literature, but they—and the broader score-based and FCM-based methodologies fall outside the main focus of this work.

2.2 LLMs in Causal Discovery

The advent of large language models has opened up new possibilities for causal discovery, giving rise to the emerging field referred to as *CausalNLP* (Jin et al., 2022, 2024, 2023). Models such as GPT (Yenduri et al., 2023) and LLaMA (Touvron et al., 2023) possess extensive pre-trained knowledge and sophisticated reasoning capabilities, enabling them to function as *automated domain experts* for causal reasoning (Wan et al., 2025). Three principal paradigms have been proposed for integrating LLMs into causal discovery workflows (Wan et al., 2025):

1. **Direct Causal Inference:** Model uses its embedded world knowledge to independently infers causal relationships directly from textual descriptions, effectively replacing human experts. Willig et al. (2022) first showed that an LLM can predict pairwise causal directions. Later work extended this approach to discovering complete and partial causal graphs (Long et al., 2024; Kıcıman et al., 2024).

2. **LLM-Assisted Causal Refinement:** A traditional statistical causal-discovery algorithm is executed first, and its output is then refined or corrected by an LLM, narrowing the search space, integrating contextual knowledge, and enriching the results with additional reasoning. Representative methods treat the model as a *posterior verifier/corrector* (Long et al., 2024; Kıcıman et al., 2024).

3. **LLM-Based Prior Integration:** Prior to any data-driven discovery, LLMs generate priors or constraints to seed the causal discovery process with expert knowledge (Ban et al., 2023; Chen et al., 2023).

This thesis focuses on the direct causal inference paradigm, which remains particularly challenging for current LLMs despite recent advances (Wan et al., 2025). A foundational requirement across all paradigms is the development of rigorous benchmarks and evaluation frameworks to assess the causal reasoning capabilities of LLMs (Yang et al., 2024). One prominent example is CORR2CAUSE, the first large-scale dataset specifically designed to evaluate pure causal inference by isolating causal reasoning from world knowledge and commonsense assumptions (Jin et al., 2024). CORR2CAUSE presents abstract, generic variable pairs, requiring models to deduce true causal relationships merely from observational data (represented in the form of conditional independencies).

Evaluation metrics are often selected to align with task granularity. For pairwise causal direction tasks, where each query is a binary classification (cause vs effect), accuracy and F1-score are employed (Bondarenko et al., 2022; Jin et al., 2024; Wan et al., 2025). For full-graph prediction tasks, structure-oriented metrics such as Structural Hamming Distance (SHD) and Normalized Hamming Distance (NHD) measure the number of edge edits needed to align the inferred graph with the ground truth (Tsamardinos et al., 2006).

Jin et al. (2024) report that LLMs perform poorly on pure causal inference benchmarks, achieving F1-scores barely above 30%. Furthermore, fine-tuning yields little improvement on perturbed or paraphrased inputs, suggesting that models learn superficial patterns rather than being able to perform causal reasoning (Joshi et al., 2024). In conclusion, while LLMs offer promising avenues for causal discovery, their performance on tasks demanding rigorous causal reasoning remains limited.

2.3 In-Context Strategies for Causal Reasoning

One line of inquiry to improve LLM-based causal discovery has been to design smarter prompt strategies that encourage the model to reason more accurately. A widely adopted technique in the domain is Chain-of-Thought (CoT) prompting (Wei et al., 2023). CoT instructs the model to "think step by step", producing intermediate reasoning before giving a final answer (Zhang et al., 2024b). By providing examples of rational, stepwise explanations, often combined with few-shot examples, CoT leverages the LLM's sequence modeling to generate a logical chain of intermediate conclusions rather than jumping straight to the answer (Zhang et al., 2022). Multiple studies outside of causality have shown that CoT prompting dramatically improves performance on various reasoning benchmarks (Wang et al., 2023). Researchers have also applied CoT to causal inference tasks. (Willig et al., 2022) found that LLM answers to pairwise causal questions became more accurate and interpretable with CoT prompts—though correctness is not guaranteed, as model can still exhibit flawed or biased logic. Subsequent work has introduced verification steps (Ji et al., 2023) or more structured prompting frameworks to enforce consistency (Zhang et al., 2024a; Vashishtha et al., 2025). These measures mitigate some errors but do not provide considerable improvement on benchmarks like CORR2CAUSE (Jin et al., 2024).

Moving beyond linear chains, Tree-of-Thought (ToT) prompting lets the model explore multiple reasoning paths rather than commit to one (Yao et al., 2023). Each intermediate step becomes a node in a tree, with the model proposing continuations or backtracking if a path seems unpromising (Besta et al., 2025). By introducing a form of search and planning, ToT can uncover solutions that a greedy chain might miss—potentially beneficial for causal discovery (Babu Shrestha et al., 2025; Le et al., 2025; Jiralerspong et al., 2024). However, ToT is computationally intensive, since many parallel chains are generated and evaluated, multiplying the number of LLM queries (Yao et al., 2023). Extensions such as graph-of-thoughts have been proposed (Besta et al., 2024), but lie outside the scope of this work.

In summary, advanced prompting techniques like CoT and ToT can boost causal reasoning but also reveal the limits of prompting alone. Recent studies have investigated self-evaluation or feedback loops for LLMs (Ren et al., 2023; Huang et al., 2023). Yet substantial research questions how truthful these self-evaluations are—some find that LLM evaluators suffer from self-bias (Xu et al., 2024) and favor their own outputs (Panickssery et al., 2024) which questions their contribution in causal discovery domain (Gendron et al., 2024).

2.4 Reasoning-Specialist LLMs

As the limitation of general-purpose LLMs on reasoning tasks became evident (Jin et al., 2024), a new class of model emerged, often called *reasoning specialists*. These models are fine-tuned with reinforcement learning (RL) to optimize for step-by-step solving, internal consistency, and logical correctness on complex reasoning tasks (OpenAI, 2024). The large-scale RL phase teaches the model to "think" productively using its own internal chain of thought, yielding substantial improvements in logical coherence and analytical depth that address many pitfalls of general LLMs (Xu et al., 2025). Whereas a conventional model such as *GPT-4* (OpenAI et al., 2024) may produce an answer in a single forward pass, a reasoning specialist iterates internally over multiple steps—sometimes taking seconds or even minutes to arrive at a response. By focusing on structured reasoning tasks, these specialists may sacrifice some open-ended conversational fluency but excel at formal logic (IBM, 2025). In the context of causal discovery, such multi-step inference is arguably advantageous—we want the model to behave more like a rigorous statistician than a free-form writer.

DeepSeek-R1, a reasoning specialist model, attained great attention for matching the performance of much larger LLMs on mathematics and coding benchmark, despite its relatively low training and inference cost (DeepSeek-AI, 2025). Its success stems from a training process that teaches the model to generate and leverage explicit "thinking tokens". During inference, *DeepSeek-R1* allocates extra computation to produce an intermediate rationale, which it then uses to derive the final answer (Marjanović et al., 2025; Zhang et al., 2025). A key advantage over hidden-chain models is that *DeepSeek-R1* exposes these "thinking traces" in its output, increasing transparency for end users (Jiang et al., 2025). Similarly, OpenAI's *o3-mini* branded as "small reasoning model" is engineered for strong STEM reasoning performance with low latency and cost comparable to much bigger architectures (OpenAI, 2025). Rather than a temperature parameter, *o3-mini* offers three-level "reasoning effort" control (low, medium, high) that effectively tunes how deeply it internally reasons. Higher effort levels allocate more internal steps, iterations, and tokens to a problem, while lower settings prioritize speed (Ballon et al., 2025). This design highlights the latency-reasoning trade-off: *o3-mini* autonomously ex-

plores solution paths, backtracks when necessary, and tries alternatives all within its inference process (OpenAI, 2025).

In conclusion, this review has traced the evolution from classical causal discovery methods to LLM-driven and reasoning-specialist models, underscoring the need for a structured, modular solution. The subsequent chapters detail and evaluate our proposed framework, demonstrating how causal reasoning in pieces can advance the state of the art in data-driven causal inference.

3 Preliminaries

This section is dedicated to discuss the CORR2CAUSE benchmark (Jin et al., 2024), which has been designed for evaluating causal discovery capabilities of generative models (Bagheri et al., 2024; Liu et al., 2024; Wan et al., 2025). In what follows, we detail the theoretical foundations of causal discovery assessed in this benchmark dataset, construction methodology, and statistical properties of the dataset. Furthermore, we critically examine the experimental evaluation performed in original work, highlighting limitations associated with fine-tuning and the reliance on Chain-of-Though (CoT) (Wei et al., 2023) prompting to our research work. The discussion lays the necessary background and justification for the research questions and methodological innovations presented in subsequent sections.

3.1 Theoretical Foundations

The CORR2CAUSE benchmark is grounded in causal discovery using Directed Graphical Causal Models (DGCMs). In this modeling, causal relationships between N variables are presented by a directed acyclic graph (DAG), where nodes denote variables and there is an edge from X_i to X_j if X_i is the direct cause of X_j (e.g., $X_i \rightarrow X_j$). Key concepts include:

Causal Relationships: The graphical definitions such as parent-child, ancestor-descendant, and special structures like confounders, colliders, and mediators (see Appendix A) are fundamental to the dataset's construction. In this benchmark, these relationships form the basis of the hypotheses, which are expressed as natural language questions. The primary objective is to evaluate whether language models can accurately infer these causal relationships, thereby demonstrating their capability to reason about causal structures.

D-Separation (Pearl, 1988): D-separation is used to determine conditional independence between variables, by reading off the underlying causal graph (see Appendix A). A clear understanding of d-separation is crucial, as it defines which variables are conditionally independent, thereby enabling mapping from the underlying causal structure to the observational independencies.

Markov Property and Markov Equivalence: The Markov Property states that each variable is conditionally independent of its non-descendants given its parents, allowing the joint probability distribution to be factorized according to the structure of the DAG. Moreover, different DAGs may induce the same set of joint distributions, resulting in the formation of Markov Equivalence Classes (MECs). For a language model to accurately infer the causal hypotheses in this benchmark, it must not only identify direct causal relationships but also recover MECs. This involves identifying all possible DAGs within an equivalence class to ensure that the in-

ferred causal relationships are consistent with the given conditional independencies among the variables.

3.2 Construction and Task Formulation

The CORR2CAUSE benchmark (Jin et al., 2024), is built through a systematic pipeline. In brief, for a set of N variables, all possible DAGs are generated by allowing edges $X_i \to X_j$ only when $i < j$, ensuring acyclicity, with redundant graphs removed via isomorphism checks. d-separation sets are then extracted to capture conditional independence relationships, and the unique DAGs are clustered into Markov Equivalence Classes (MECs) under the faithfulness assumption. For each pair of variables, six standardized causal hypotheses (e.g., "Is-Parent", "Is-Child", etc.) are generated and assigned a binary validity label v based on whether the hypothesized relation holds across all graphs in the corresponding MEC; these conditional independencies and hypotheses are subsequently verbalized into natural language.

Within this benchmark, the core task is to learn a function

$$f : (s, h) \mapsto v,$$

where s describes the conditional independencies among the variables, h posits a causal relation between two specific variables X_i and X_j, and $v \in \{0, 1\}$ indicates the validity of the proposed relation.

3.3 Evaluation and Methodological Perspective

Thanks to systematic construction pipeline, the CORR2CAUSE benchmark provides a robust framework for evaluating large language models in a controlled settings for causal discovery.

However, the original experimental methodology reveals critical limitations of most language models in inferring causal relationships. Although extensive fine-tuning yields high in- distribution performance (e.g., *RoBERTa-Large MNLI* achieving an F1 score of 94.74%), performance degrades markedly under adversarial perturbations—dropping to 55.45% with paraphrasing and from 85.52% to 46.96% for OpenAI's *GPT-3* under variable refactorization, indicating that models tend to overfit to dataset-specific patterns rather than having causal reasoning. While the benchmark paper also explores in-context learning via Chain-of-Thoughts (CoT), we aim to improve upon these techniques by integrating advanced reasoning models that do not rely solely on extensive fine-tuning.

3.4 Exploratory Data Analysis

Exploratory Data Analysis (EDA) is an fundamental first step in any empirical investigation. It provides a visual and statistical overview of a dataset characteristics, helping to spot patterns, anomalies, and basic distributions before any formal modeling (Hoaglin et al., 2000). This preliminary step inform later choices and guards against unreliable conclusions.

In the context of evaluating causal discovery with LLMs on the CORR2CAUSE benchmark, a comprehensive EDA provides critical insights. The dataset comprises natural-language premises

Table 1: Descriptive statistics for the test split of the CORR2CAUSE benchmark.

Statistic	Count	Percent (%)
Total examples (hypotheses)	1178	100.0
Not Supported (label = 0)	986	83.7
Supported (label = 1)	192	16.3
Graph size (num_variables)		
2 variables	6	0.5
3 variables	50	4.2
4 variables	74	6.3
5 variables	515	43.7
6 variables	533	45.2
Templates (categories)		
child	194	16.5
parent	194	16.5
has_collider	193	16.4
has_confounder	194	16.5
non-child descendant	194	16.5
non-parent ancestor	195	16.6

outlining independence statements, associated graph sizes (number of variables), and one of six structural-relationship templates, yielding a heterogeneous structure. Understanding these characteristics is essential, as they can significantly influence the empirical experiments outcome, and how LLMs process the information and perform causal reasoning tasks.

Table 1 presents descriptive statistics for the test split of the CORR2CAUSE benchmark. If the hypothesis is true in the underlying ground-truth graph, the label is 1 (`Supported`); if false, it is 0 (`Not Supported`). The dataset is heavily skewed toward the `Not Supported` class ($\sim 84\%$ of examples) versus the `Supported` class ($\sim 16\%$). In the presence of such a pronounced imbalance, overall accuracy can be misleading; precision and recall (and their harmonic mean, F1-score) become far more informative metrics for assessing model performance on the minority class (Jin et al., 2024). Graphs with 5–6 variables comprise nearly 90% of instances. This is expected: as the number of variables grows, the number of possible template-specific graphs increases super-exponentially, whereas with only two variables each template yields exactly one distinct graph. Finally, each of the six structural templates appears in approximately 16-17% of examples—that guarantees every causal query is evaluated with equal coverage.

Figures 2 and 3 provide complementary insights into the characteristics of the CORR2CAUSE causal reasoning dataset. Figure 2 illustrates the distribution of the six structural relationship templates across supported (label = 1) and unsupported (label = 0) hypotheses, revealing notable variations in template prevalence. As shown in Table 1, all templates are dominated by unsupported examples, reflecting with the overall $\sim 84\%$ skew toward false hypotheses. However, the `has_collider` and `parent` templates exhibit the highest support rates among the six patterns, whereas the `child` template has zero supported instances in the test split.

Figure 3 presents the Pearson correlation matrix among three numeric features: premise length (word count), graph size (number of nodes), and hypothesis label (1 = supported, 0 = not

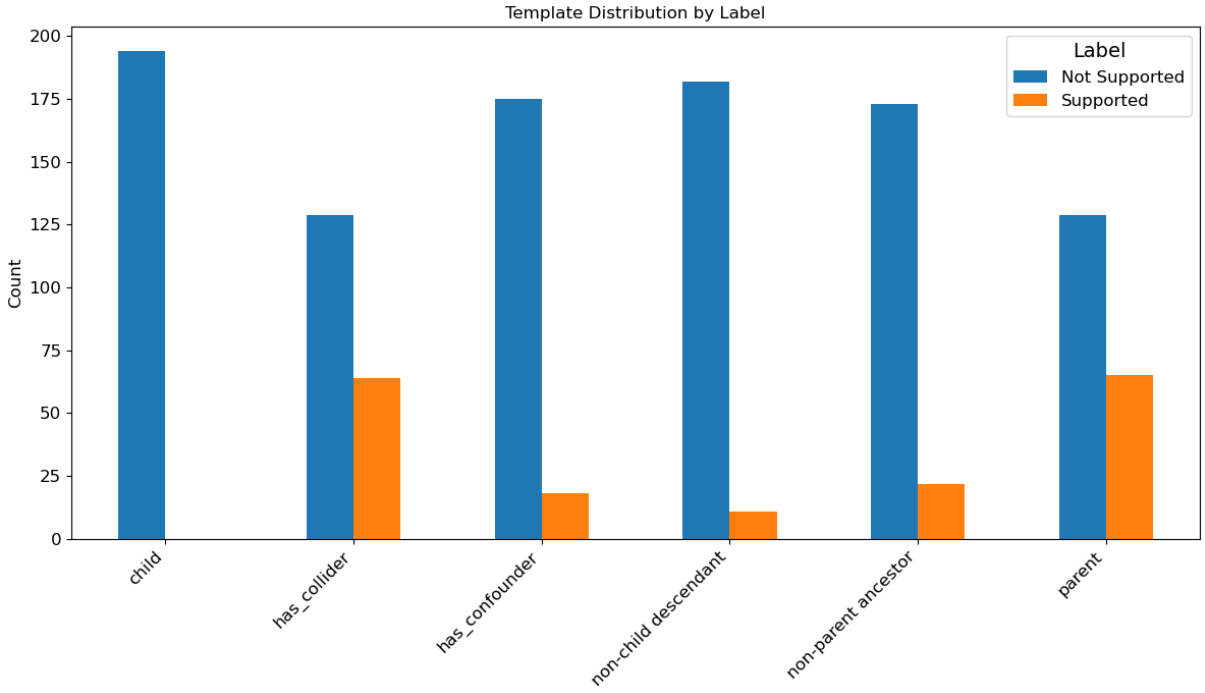


Figure 2: Frequency distribution of different causal reasoning templates in the CORR2CAUSE test set, stratified by hypothesis support. Blue bars show the count of "not supported" hypotheses (label = 0), and orange bars show "supported" hypotheses (label = 1).

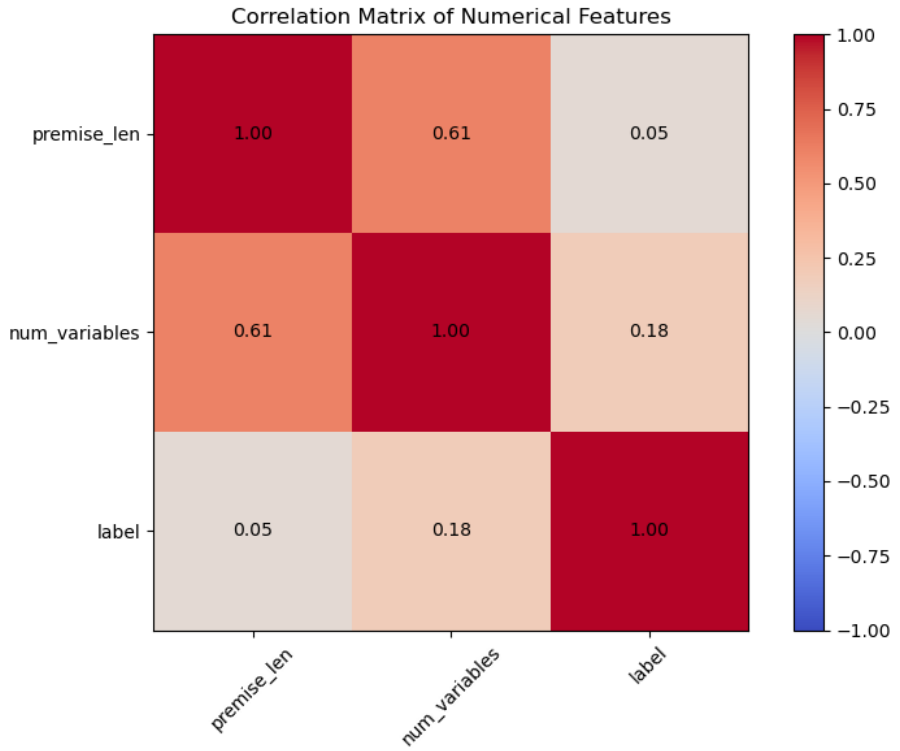


Figure 3: Pairwise Pearson correlation matrix of key numerical features in the CORR2CAUSE test set: premise length (word count), causal graph size (number of variables), and hypothesis support status.

supported). We observe a moderate positive correlation ($r = 0.61$) between premise length and graph size, reflecting that larger graphs naturally yield longer textual descriptions. In contrast, graph size shows only a weak positive correlation with hypothesis validity ($r = 0.18$), and premise length is essentially uncorrelated with the label ($r = 0.05$), indicating that longer inputs do not systematically make hypotheses more or less likely to hold. These near-zero correlations indicate that each feature provides largely independent signals, reducing the risk that model predictions are trivially driven, for instance, by defaulting to `not supported` on longer inputs.

4 Research Objectives and Problem Statement

Large language models (LLMs) have demonstrated impressive performance across a wide variety of reasoning tasks. Nonetheless, recent studies highlight significant challenges when these models are confronted with causal discovery tasks—where often rely on superficial correlation patterns instead of genuine causal inference (Jin et al., 2024; Zečević et al., 2023). Meanwhile, a new class of *reasoning-specialist* LLMs (e.g. *DeepSeek-R1*, *OpenAI's o3-mini*) explicitly trained via reinforcement learning to perform multi-step inference, has emerged. A systematic assessment of their causal reasoning capabilities is still missing.

4.1 Summary and Motivation

The literature reviewed in Sections 2.1, 2.2, 2.3, and 2.4 reveals both the potential and limitations of applying large language models to causal discovery. Traditional causal discovery methods, such as the Peter-Clark (PC) algorithm (Section 2.1), provide a sound theoretical foundation but lack integration of high-level reasoning and external knowledge. LLMs step into this gap by serving as domain-knowledge agents that can interpret correlational evidence and suggest causal relations (Section 2.2) (Wan et al., 2025). However, benchmarks like CORR2CAUSE clearly demonstrate that off-the-shelf LLMs fare not robust causal reasoners when faced with purely data-driven problems (Jin et al., 2024). Advanced prompting techniques like Chain-of-Thought (CoT) and Tree-of-Thought (ToT) (Section 2.3) improve interpretability and accuracy, yet they fall short of overcoming inherent logical and consistency limitations. Conversely, reasoning-specialist models such as *DeepSeek-R1* and *o3-mini* (Section 2.4) show that targeted training for step-by-step inference can elevate reasoning capabilities from an emergent property to an optimized skill.

Building on these insights, this thesis proposes a novel modular in-context learning pipeline for causal discovery. The main contribution is to quantify the native reasoning performance of specialist models (*DeepSeek-R1*, *o3-mini*) on causal tasks (Section 6.3), then decompose the overall discovery process into smaller, manageable reasoning modules (Section 6.4). This "in pieces" strategy reflects our approach of orchestrating multiple smaller modules to construct robust, effective end-to-end causal discovery process with the LLMs. We hypothesize that this framework will substantially improve performance on challenging benchmarks such as CORR2CAUSE.

4.2 Study Aims

Building on the CORR2CAUSE benchmark, our study is motivated by recent breakthroughs in large language models and their emergent embedded reasoning capabilities. Notably, open-source model such as *DeepSeek-R1* (DeepSeek-AI, 2025) and proprietary OpenAI's *o3-mini* (OpenAI, 2025) exemplify these advances. *DeepSeek-R1*, which leverages reinforcement learning to induce transparent reasoning behaviors without relying on extensive supervised fine-tuning, and OpenAI's *o3-mini*, designed for cost-effective reasoning with flexible inference settings. These models provide a promising approach for achieving robust causal inference. In this context, we aim to consider:

In-Context Learning: While extensive fine-tuning of non-reasoning models on the CORR2 CAUSE benchmark yields high in-distribution performance, these models fail to generalize to perturbations in dataset (such as node relabeling). We seek to explore whether state-of-the-art reasoning LLMs can effectively perform causal discovery tasks using in-context learning, thereby reducing the reliance on heavy fine-tuning.

Prompt Strategies: Given that reasoning-specialist LLMs (e.g., *DeepSeek-R1*) exhibit emergent, RL-driven internal CoT processes, we investigate novel prompt formulations to enhance their base reasoning capabilities. Our goal is to guide these models more effectively in inferring causal relationships.

Modular Causal Discovery Framework: Inspired by the Tree of Thoughts (Yao et al., 2023) and Chain-of-Thoughts (Wei et al., 2023) approaches, we propose a streamlined causal discovery pipeline that decomposes the problem into smaller, manageable sub-tasks with intermediate reasoning stages. By guiding the model through detailed per-stage prompts, we aim to foster deeper, more in-depth reasoning and improve overall performance in causal discovery.

Reasoning Processes: We analyze the reasoning traces produced by reasoning model on identical CORR2CAUSE samples and contrast them with the answers of conventional models, pinpointing the inference steps and patterns that give reasoning models its edge. We further examine whether cases misclassified by the conventional model correspond to longer, more elaborate reasoning chains—to explore how reasoning length relates to accuracy.

By clearly defining and addressing these research objectives, our thesis systematically advances the understanding and capabilities of LLM-based causal discovery methodologies.

5 Conventional LLMs Methodologies

In this section, we conduct preliminary experiments with conventional LLMs, specifically *LLaMA3.3-70B* (Touvron et al., 2023) and the *GPT*-series from OpenAI (Yenduri et al., 2023) on the CORR2CAUSE benchmark (Jin et al., 2024). Our primary research questions is whether *conventional, general-purpose LLMs can accurately reconstruct undirected causal skeletons from conditional independence statements using standard in-context prompting strategies.* Addressing this question provides an essential baseline that clarifies the inherent limitation of widely adopted LLM architectures when they are tasked with causal discovery problems. Establishing this baseline is crucial for demonstrating the potential benefits of specialized reasoning models and structured prompting frameworks introduced in subsequent parts of this work.

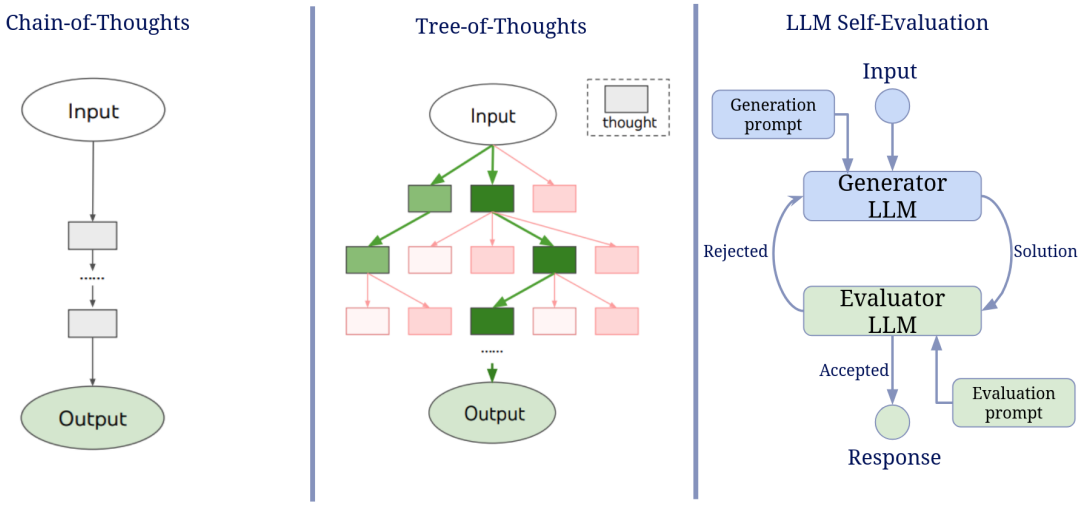


Figure 4: Overview of three in-context prompting paradigms evaluated with general LLMs. Starting from the left: (a) Chain-of-Thought (CoT)—one continuous reasoning trace per query; (b) Tree-of-Thought (ToT)—parallel exploration and selection of multiple reasoning branches; (c) Self-Evaluation Feedback Loop—iterative generation and critique by a secondary evaluator prompt.

To this end, in Section 5.1 we apply three widely used in-context prompting strategies, each requiring no fine-tuning and leveraging the models' latent reasoning capabilities. Subsequently, in Section 5.2, we empirically evaluate each prompting strategy specifically targeting the undirected skeleton discovery stage of CORR2CAUSE.

We quantify performance using the `exact_match` ratio, defined as the proportion of cases in which the predicted skeleton exactly matches the ground-truth skeleton, irrespective of edge orientation. We selected the `exact_match` ratio because it directly assess the strict correctness of graph reconstruction, providing a binary and definitive criterion. This metric is particularly suitable because partial matches or minor deviation can significantly compromise causal correctness in the subsequent stages of PC algorithm.

5.1 In-Context Prompting

Figure 4 provides an overview of the three in-context prompting paradigms we applied to conventional LLMs. In leftmost panel (a), the Chain-of-Thought (Wei et al., 2023) setup begins with a comprehensive system prompt that guides the model through four fixed steps: (i) read the data and enumerate all marginal and conditional independence statements; (ii) initialize the full correlation graph; (iii) systematically test each edge against all independence statements, documenting its full reasoning and decision trace; and (iv) output the final undirected skeleton in a rigid, human-readable format. To ground the model's behavior, we include a small set of few-shot examples that illustrates exactly how each step should look, ensuring the model unfolds its internal reasoning rather than leaping to a final answer (complete prompt listings for Chain-of-Thought, Tree-of-Thought, and the self-evaluation loop are provided in Appendix C).

Table 2: Exact_match ratio for undirected skeleton discovery stage on the CORR2CAUSE test split. Results are reported for Chain-of-Thought (CoT) and Tree-of-Thought (ToT) prompting across common LLMs at different graph sizes (temperature = 0.1).

Paradigm	**Model**	**Exact Match Ratio**		
		4 variables	5 variables	6 variables
Chain-of-Thought	gpt-4o	**0.95**	**0.82**	**0.54**
	gpt-4o-mini	0.92	0.50	0.10
	LLaMA3.3-70B	0.88	0.64	0.22
Tree-of-Thought	LLaMA3.3-70B	0.87	0.60	0.21

The middle panel (b) illustrates our Tree-of-Thought (Yao et al., 2023) implementation, in which the same four undirected skeleton discovery driven steps are executed as a branching search. At each step, the model applies a concise prompt template (e.g., "Apply Independencies: remove edges based on each statement and justify each removal") along with few-shot examples to spawn multiple candidate "thoughts." We then perform a depth-first traversal, pruning low-scoring branches and ultimately selecting the path whose final skeleton best satisfies all independence constraints. This explicit exploration of alternative reasoning trajectories helps recover correct graphs when a single linear chain might introduce errors.

In the rightmost panel (c), we consider a lightweight Self-Evaluation feedback loop (Ren et al., 2023; Huang et al., 2023). After generating an undirected skeleton with the CoT prompt, we feed it into a secondary evaluation prompt, asking the model to identify any mismatched edges and score its edge removal decisions on a 0–10 scale against our strict criteria. If the self-score falls below our threshold (≥ 8), we discard the skeleton and request a regeneration. When the model flags specific misclassified edges, we automatically correct those edges and accept the revised skeleton if it then meets the threshold. This single iteration feedback captures and rectifies many errors at minimal overhead, without the computational explosion of a full Tree-of-Thought search.

5.2 Empirical Analysis

Table 2 reports the `exact_match` ratio for the undirected skeleton discovery stage, computed in our programmatic evaluation sandbox against ground-truth skeletons. Performance degrades sharply as graph size increases: for CoT on *gpt-4o-mini*, the ratio falls from 0.92 at four variables to 0.10 at six variables. Upscaling the model size to *gpt-4o* yields substantial gains reaching 0.54 at six variables versus 0.10 for the mini version. We observe no significant difference between self-hosted *LlaMA3.3-70B* and the OpenAI's proprietary models under identical prompting. All results use the optimal temperature setting (0.1) found during tuning.

Contrary to expectations, ToT did not substantially outperform CoT on our self hosted *LLaMA3.3-70B*. Although both paradigms achieve similar scores at four variables (0.88 vs. 0.87), the marginal gains at larger graph sizes are insignificant. This outcome likely stems from the fact that Tree-of-Thought performance depends on implementation specific choices: branching heuristics, search depth, and pruning strategy that vary between applications. Although we followed the original ToT guidelines in our own implementation, further optimization

of these parameters might unlock better results. We also attempted to integrate third-party ToT libraries, but found they could not be readily adapted to our evaluation framework, underscoring the need for task-specific customization to realize the full potential of ToT.

A single-iteration of Self-Evaluation feedback loop (omitted from Table 2 due to the absence of meaningful output) tasked the model with critiquing and correcting its own edges in one pass. It failed to produce usable outputs, indicating that a single round of self-critique is insufficient for uncovering errors in complex graph reconstruction—and may even suggest that the model cannot reliably detect its own mistakes.

6 Modular In-Context Pipeline Framework

We build our methodology on the mathematical foundations of causal discovery, with the Peter-Clark (PC) algorithm (Spirtes et al., 2001) as its core. We pursue two complementary strategies on the CORR2CAUSE benchmark (Jin et al., 2024). First, a *baseline in-context learning* approach prompts models directly to judge causal hypotheses from natural-language conditional independence statements (Section 6.3). Second, we introduce a *modular in-context causal discovery pipeline*, which decomposes the PC algorithm into separate reasoning stages (Section 6.4). This pipeline both enhances the interpretability and robustness of state-of-the-art models and yields a fully in-context framework for causal discovery that requires no additional fine-tuning. Third, Section 6.5 documents the key engineering challenges—prompt iteration, schema enforcement, token budgeting, and parameter tuning, as well as the iterative solutions that ultimately enabled the high-performing pipeline reported in Section 7.

6.1 Peter-Clark Algorithm

The PC algorithm is a well-known constraint-based causal discovery algorithm, using conditional independence testing to recover a causal graph from purely observational data. It proceeds in two main phases: first, it constructs an undirected skeleton by removing any edge between a pair of variables whenever a conditional independence test succeeds; second, it partially orients the remaining edges. Crucially, the correctness of these steps rests on three assumptions, which together guarantee that the set of conditional independence observed in the data reflects the true underlying causal structure. We now briefly review these assumptions before guiding an LLM through each phase of the algorithm:

Directed Acyclic Graphs (DAGs): It is assumed that the causal relationships among the variables $\mathcal{X} = \{X_1, \ldots, X_N\}$ is represented by a DAG $G = (X, E)$, where each directed edge $X_i \to X_j$ denotes a direct causal effect and no directed cycles exist.

Causal Markov Condition: Under the Causal Markov assumption (Scheines, 2005), each variable X_i in a DAG is independent of its non-descendants given its direct parents:

$$X_i \perp\!\!\!\perp \mathrm{NonDesc}(X_i) \;\big|\; \mathrm{Pa}(X_i).$$

Equivalently, the joint distribution factorizes as $P(X) = \prod_i P(X_i \mid \mathrm{Pa}(X_i))$. For instance, in the chain $A \to B \to C$, one obtains $A \perp\!\!\!\perp C \mid B$. For a distribution generated by a structural causal model.

Faithfulness: Under the faithfulness assumption, the observed distribution exhibits exactly the conditional independence relations implied by the DAG via d-separation (see Appendix A for a formal definition) (Uhler et al., 2013). Equivalently, no conditional independencies exist beyond those entailed by the causal the Markov Property.

6.2 Stages of PC Algorithm

The PC algorithm can be decomposed into four sequential stages. This decomposition underlies our modular pipeline framework, issuing each stage as a dedicated prompt and can alternatively be merged into a single, comprehensive baseline. By studying each stage separately, we can systematically review and analyze model responses for that stage. The full PC procedure can be decomposed as follows:

1. **Skeleton Estimation:** Construct the skeleton of the true underlying graph by checking each pair (X_i, X_j) for finding a conditional independence $X_i \perp\!\!\!\perp X_j | S$ for some subset $S \subseteq \mathcal{X} \backslash \{X_i, X_j\}$, and removing edge X_i–X_j whenever conditional independence is detected.

2. **V-structure Identification:** From the skeleton, identify all triples (X_i, X_k, X_j) where X_i and X_j are both adjacent to X_k but not to each other. Orient these as $X_i \rightarrow X_k \leftarrow X_j$ whenever X_k does not appear in any separating set S for (X_i, X_j).

3. **Edge Orientation via Meek's Rules (Meek, 2013):** Having identified the undirected skeleton and all colliders, Meek's rules are applied to orient further edges. Each rule matches a specific subgraph in the current Partially Directed Acyclic Graph (PDAG) and orients one further, subject to the constraints of acyclicity and collider preservation. Repeated application of these rules yields a maximally oriented PDAG, also known as the Completed Partially Directed Acyclic Graph (CPDAG), which represents the Markov Equivalence Class (MEC) of DAGs consistent with the observed conditional independencies.

4. **Hypothesis Evaluation within the Equivalence Class:** Given the resulting CPDAG, determine whether the target causal hypothesis holds in every valid DAG of its Markov Equivalence Class.

These four steps form the backbone of both our *baseline in-context learning* (where all are embedded in one prompt) and our *modular pipeline* (where each step is a separate prompt). In the next sections, we describe how each approach instantiates these tasks.

6.3 Single-Prompt Baseline Assessment

To establish a zero-shot performance baseline and probe the native causal-reasoning capabilities of modern reasoning large language models, we employ a single comprehensive prompt that encapsulates all four stages of the PC algorithm described in Section 6.2 within one in-context request. This approach requires minimal prompt engineering, serving both as an accessible native method and as a baseline for a detailed in-context causal discovery pipeline

Table 3: Single-prompt abstract template for PC algorithm based causal hypothesis evaluation in reasoning models: in-context guidance via persona framing and key principles.

Preamble	Frame the model as a causal-discovery scientist with PC-algorithm expertise.
Task description	Evaluate a causal hypothesis using an end-to-end, in-context PC execution.
Key principles	Enumerate the algorithmic steps in skeleton extraction, collider identification, Meek-rule orientation, and hypothesis validation.
I/O specification	Provide a premise (conditional independence statements) and a hypothesis block, and elicit a binary validity judgment.

in Section 6.4. By comparing results from this unified prompt against conventional or fine-tuned CORR2CAUSE models, we can quantify gains attributable to emergent reasoning models without any external fine-tuning.

As detailed in Table 3, our single-prompt template comprises four coordinated elements: a persona-based preamble, a concise task description, key principles, and a unified I/O specification. The persona preamble situates the model in a domain-specific causal-discovery role, an approach shown to improve reasoning consistency and contextual grounding (Tseng et al., 2024; Tsai et al., 2024). The task description and key principles explicitly restate the PC algorithm's procedural steps, skeleton extraction, v-structure identification, applying Meek's rules, and hypothesis validation to steer the model's internal thinking process. Finally, the I/O specification standardizes a uniform binary output, enabling systematic, fully automated evaluation. This streamlined assessment exploits modern reasoning models' native strengths with minimal engineering and provides a clear reference point for demonstrating the gains of our modular pipeline (see Section 6.4). For full reproducibility, the complete prompt template is provided in Appendix D.

6.4 Modular In-Context Causal Discovery Pipeline

Building upon our zero-shot baseline, we now introduce a four-stage in-context pipeline that decomposes the PC algorithm into separate prompts. This modular framework amplifies the model's reasoning effort at each stage and produces intermediate artifacts for fine-grained error analysis, yielding substantial gains over the monolithic baseline on the CORR2CAUSE benchmark.

Although our single-prompt approach captures some native reasoning capacity of modern LLMs, it merges four distinct algorithmic operations into one request, making failures difficult to diagnose and limiting the model's focused attention on each stage. Inspired by Tree-of-Thoughts and Chain-of-Thoughts methodologies (Yao et al., 2023; Wei et al., 2023), we instead decompose the task into explicit, stage-wise prompts each targeting a single PC stage, and show that this yields both clearer diagnostics and substantial performance gains.

The overall workflow is depicted in Figure 5. Each vertical panel comprises an LLM prompt followed by a parser module. Arrows indicate that each stage consumes the previous parser's output as structured input for the next prompt. By design, no stage requires any external fine-

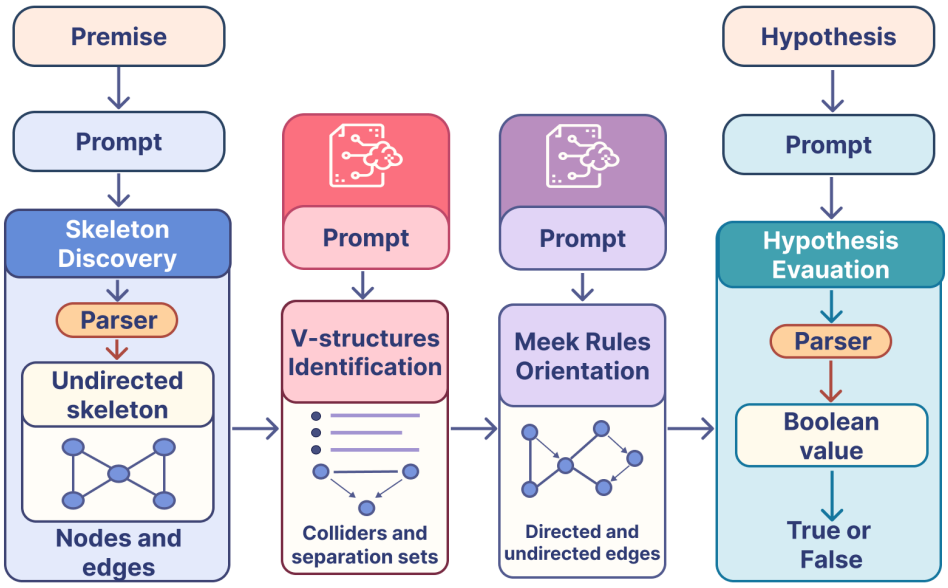


Figure 5: Stage-wise in-context pipeline for causal discovery. Each vertical panel represents a separate prompt and parser corresponding to a stage of the decomposed PC algorithm, with each stage's output serving as the next stage's input. Arrows indicate the flow of data between prompts and parsing modules.

tuning, only in-context examples and carefully crafted instructions. Moreover, we standardize across all four prompts the persona framing, input-schema template, instruction wording, and output-schema enforcement. Parser modules, implemented in Python, convert raw LLM outputs into canonical Python objects, lists of v-structure triples and edges for onward prompting. Full prompt text appears in Appendix E. For a complete overview of the software architecture—interfaces, concrete stages, LLM back-ends, and the pipeline orchestration, see Appendix G.

These design choices also affect prompt length and token budget. Although shorter, focused prompts fit more comfortably within model context windows, four sequential calls incur additional latency. In our experiments, the performance gains far outweigh the modest increase in token usage. Our implementation additionally includes automatic re-prompting on schema violations to ensure robustness during empirical evaluation. In sum, this modular in-context framework unlocks the full potential of reasoning-specialist LLMs and offers a blueprint for causal discovery.

6.5 Design Iterations and Engineering Challenges

Both the baseline studies and the modular in-context pipeline framework required extensive methodological refinement and engineering work. In this subsection, we summarize the principal challenges we faced, the iterative prompt-engineering cycles we conducted, and the practical design choices such as temperature tuning, schema enforcement, and token-budget management—that ultimately shaped the final system.

Iterative Prompt Development: Crafting effective prompts for the baseline experiments, and later, for the four-stage pipeline required an intensive, iterative process that involved hundreds of incremental adjustments and retries. We first developed a set of *key principles* for our baseline template (see Appendix D), drawing on a thorough review of prompt-engineering literature and model documentation. The same disciplined workflow was then applied to the stage-specific prompts used in the pipeline experiments (see Appendix E). Each prompt iteration was systematically logged and evaluated, yielding progressively improved outputs. This iterative approach was crucial to creating prompts that reliably elicited coherent and accurate reasoning from the models.

Temperature Parameter Tuning: The temperature setting of *DeepSeek-R1* was rigorously tuned to balance creativity and consistency. We conducted a structured parameter sweep over temperature values $\{1, 0.6, 0.1\}$, observing empirically that higher temperatures introduced substantial variability and hallucinations, whereas the lowest settings (0.1) produced stable, precise, and reproducible outputs. We therefore adopted a conservative temperature of 0.1 to ensure high fidelity and robustness of our final pipeline stages (see Table 4 for full results).

I/O Robustness: Ensuring the validity and consistency of model-generated outputs posed a significant challenge. Early experiments frequently produced outputs that deviated from the expected format, complicating downstream parsing and evaluation. To address this, we implemented strict JSON schema validators and integrated prompt-retry mechanisms that activate upon any schema violation. This approach greatly improved the pipeline's reliability and enabled seamless integration across stages.

Evaluation Sandbox: To iterate efficiently and refine each stage of the modular pipeline independently, we developed a dedicated evaluation sandbox environment. The sandbox environment allowed isolated testing of each stage, enabling rapid iteration without error propagation across components. By refining individual stages through isolated trials, we ensured that every subcomponent met the highest standards before integration (see the stage-wise performance in Table 5).

Token Budgeting: Careful resource management was essential in both deployment modes. On the GPU cluster, we tuned batch sizes, context-window limits, and memory settings so that the self-hosted models fit on dual *NVIDIA H100* GPUs without triggering out-of-memory errors. For the API-accessed models, every prompt tracked token usage and employed early-exit guards to prevent runaway costs. Comparative profiling showed that OpenAI's *o3-mini* achieved the best accuracy-per-token ratio, whereas DeepSeek-R1 required roughly two to three times more tokens for comparable performance (see Figure 6).

In summary, the challenges we encountered and the iterative methodological enhancements we applied underscore the non-trivial nature of our prompt engineering and pipeline development. These careful considerations and adjustments were essential to achieving the robust, reproducible, and high-performing modular in-context pipeline presented in this thesis.

7 Empirical Evaluation and Analysis

In this section, we present a threefold assessment of our causal-discovery methods. First, in Section 7.1, we compare standard reasoning model baselines against the original CORR2CAUSE

Table 4: Comparative performance on the CORR2CAUSE benchmark, reporting F1 (primary metric), precision, recall, and accuracy for published benchmark results, zero-shot reasoning-model baselines, and the modular pipeline framework. Within each block, the best value in each column is underlined; the overall best metrics across all methods are both bold and underlined.

Method	F1	Precision	Recall	Accuracy
Benchmark results				
Random (uniform)	20.38	15.11	31.29	62.78
BART MNLI	33.38	31.59	35.38	78.50
Alpaca-7B	27.37	15.93	97.37	21.33
GPT-4	29.08	20.92	47.66	64.60
Baseline results				
DeepSeek-R1-70B (temp=1)	44.93	30.39	86.11	67.24
DeepSeek-R1-70B (temp=0.6)	47.02	32.54	84.75	70.09
DeepSeek-R1-70B (temp=0.1)	48.56	33.76	86.49	70.80
DeepSeek-R1 API (temp=0.1)	64.57	53.33	81.82	86.02
OpenAI o3-mini	66.28	70.19	62.78	90.10
Pipeline framework results				
DeepSeek-R1-70B (temp=0.1)	65.33	55.88	78.62	87.28
DeepSeek-R1 API (temp=0.1)	79.83	73.40	**87.65**	93.27
OpenAI o3-mini	**83.83**	**90.91**	77.78	**95.32**

benchmark (Jin et al., 2024), and demonstrate how the robust causal-discovery pipeline builds upon and surpasses both these baselines and the initial benchmark results. Second, in Section 7.2, we conduct an inter-model analysis of different reasoning architectures, focusing on the stages at which each model is most likely to introduces mistakes. Finally, in Section 7.3, we offer a combined quantitative and qualitative examination of why conventional (non-reasoning) models fail on critical independence assertions and how the reasoning models address these failures, along with a discussion of the associated complexity overhead.

All empirical experiments described in this section were conducted using the computational environment and inference settings detailed in Appendix B.

7.1 Comparative Benchmarking

We evaluate on the full test split of the CORR2CAUSE benchmark, reporting F1 (primary metric), precision, recall, and accuracy for published baselines, our zero-shot reasoning models, and the four-stage pipeline, in Table 4. Among the published results, *BART MNLI* achieves the highest F1 (33.38), reflecting the limitations of conventional language models. In contrast, zero-shot reasoning models deliver dramatic native improvements. The self-hosted *DeepSeek-R1* (70B parameters) attains an F1 of 48.56, while the much larger *DeepSeek-R1 API* (685B parameters) and OpenAI's *o3-mini* (size undisclosed) reach F1 of 64.57 (see Apendix F for bootstrap standard deviation and 95 % confidence interval) and 66.28, respectively. These gains show both the power of emergent reasoning models and the impact of model scale when

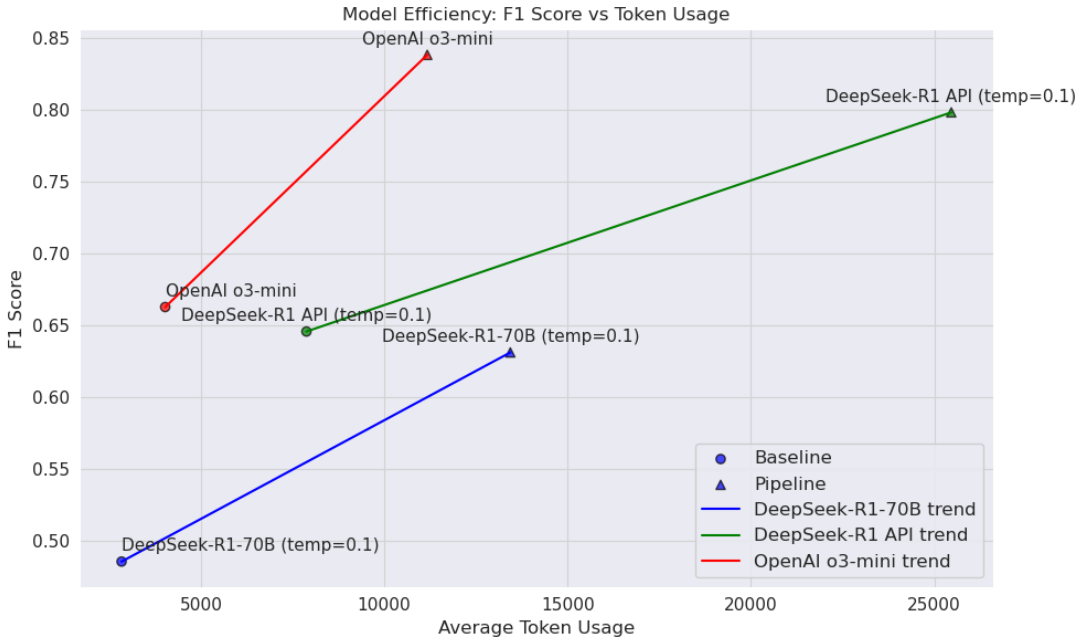


Figure 6: Efficiency trade-off on the CORR2CAUSE benchmark. Scatter points denote F1 scores against average token usage for both baseline (circles) and pipeline (triangles) settings. Solid lines connect each model's baseline and pipeline points as trend lines, highlighting how increased token investment in the pipeline correlates with F1 improvements.

applied without any additional fine-tuning.

Proposed modular pipeline further amplifies performance, *DeepSeek-R1 API*'s F1 jumps to 79.83, and the OpenAI's *o3-mini* pipeline achieves 83.83. Notably, these improvements are accompanied by more balanced precision and recall despite the dataset's approximately 85% "false" label prevalence. Recall increases by roughly 10–15 percentage points over the zero-shot baselines, accompanied by even larger gains in precision, indicating that our stage-wise prompts help the model avoid a trivial "always-false" strategy. Together, these results validate that (i) reasoning-specialist LLMs substantially outperform prior models in a pure zero-shot setting and ii a carefully structured, in-context pipeline can unlock further gains, achieved solely through prompt design and stage decomposition, without external supervision or fine-tuning.

7.2 Inter-Model Comparison

Figure 6 plots each model's baseline and full-pipeline F1 scores against its average token usage. Immediately evident is the linear uplift that our modular pipeline yields over the same monolithic prompt—every triangle sits above its corresponding circle, demonstrating that decomposing the PC algorithm into stage-wise subprompts delivers consistent gains without any additional fine-tuning. Under the identical single-prompt setup, the *DeepSeek-R1 API* (685B parameters) outperforms its 70B counterpart by approximately 15 $F1$ points, underscoring our hypothesis that *more intrinsic reasoning capacity* leads to better causal discovery performance.

OpenAI's *o3-mini* not only achieves the highest end-to-end F1 but also the most favorable trade-off between token cost and accuracy—rising from roughly $4K$ to $11K$ tokens for a

Table 5: Stage-wise F1 scores for each pipeline-tested reasoning model.

Model	Undirected Skeleton Discovery	V-structures Identification	Meek's Rules Orientation	Hypothesis Evaluation
DeepSeek-R1-70B	0.83	0.76	0.70	0.58
DeepSeek-R1 API	1.00	0.99	0.90	0.80
OpenAI o3-mini	1.00	0.99	0.95	0.84

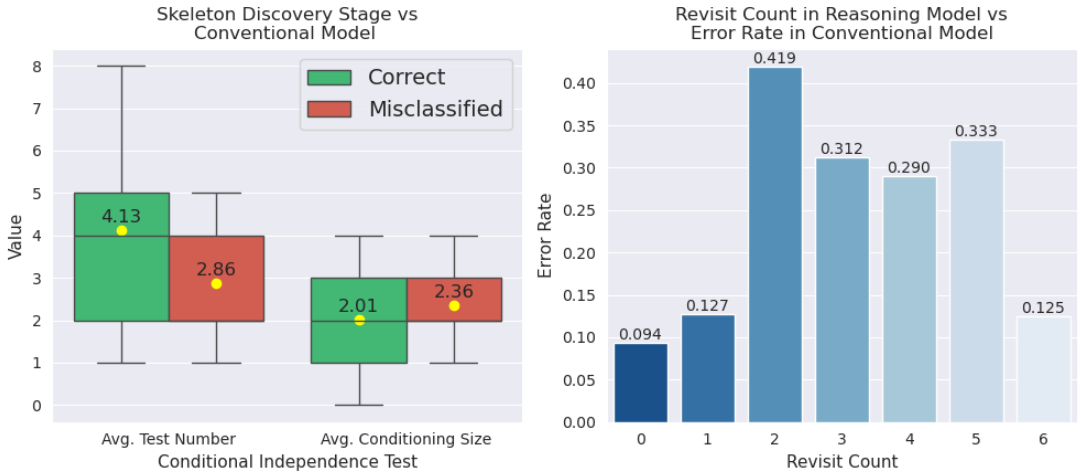


Figure 7: (Left) Boxplot of conditional independence sets metrics—number of tests and maximum conditioning size reported for the conventional model. Edges are labeled "Correctly classified" if the skeleton matches the ground truth, and "Misclassified" if it includes incorrect edges or leaves out true ones. Yellow dots indicate mean values. (Right) Bars show, for each revisit count k, the fraction of edges that the common model misclassified among those the reasoning model revisited k times.

$+0.17$ points gain in F1 , versus *DeepSeek-R1 API*'s $8K \rightarrow 25K$ tokens for a comparable improvement. Thus, while larger models can invest more computation, the *o3-mini* attains superior performance per token when orchestrated through our pipeline.

Table 5 summarizes each model's F1 at the four pipeline stages. All three models achieve near-perfect skeleton discovery and v-structure identification steps, but diverge later under Meek's rules and hypothesis evaluation phases, with the *o3-mini* retaining the strongest performance. Taken together, these comparisons show (i) our modular pipeline uniformly boosts every model's F1 via focused, stage-wise reasoning; (ii) larger LLMs begin already perform better on the initial stages; and (iii) OpenAI *o3-mini* achieves the best overall trade-off of token cost and final accuracy.

7.3 Qualitative and Quantitative Failure Analysis

We use *LLaMA3.3-70B* as the conventional model and *DeepSeek-R1-70B* (distilled from *LLaMA3.3-70B*) as the reasoning model. We selected approximately 100 first stage samples where the common model made at least one error; the reasoning model attained perfect accuracy on these. Behaviorally, the reasoning model performs iterative self-checks ("Wait...",

"Hold on..."), explicitly revisits each edge during reasoning trace micro-steps (mean of 66 micro-steps, maximum of 135 micro-steps), and refines decisions in real time. While specific patterns vary, this "thinking aloud" and iterative refinement underlie its qualitative advantage.

Figure 7 (left) compares the average number of independence tests of pair of variables that common models classify correctly versus those it misclassifies. Misclassified edges have fewer tests (mean 2.86 vs 4.13). It is important to note that the presence of even a single conditional independence between a given pair of variables is sufficient to remove the edge connecting them in the skeleton. The common models often require multiple conditional independence assertions involving the same pair of variables in the premise to remove the edge between them from the skeleton. We also plot the average size of the conditioning set in a conditional independence test (i.e., the size of Z in a test of the form $X \perp\!\!\!\perp Y \mid Z$) for both correctly and misclassified instances. As observed, the average size of the conditioning set tends to be larger in the misclassified class.

The right panel shows the average error of the conventional model on instances where the reasoning model revisits k times in its reasoning trace. This result shows frequent re-evaluations as a key mechanism by which the reasoning pipeline corrects the common model's errors.

8 Conclusions

In this thesis, we performed a comprehensive reassessment of the CORR2CAUSE benchmark, beginning with an empirical investigation of conventional large language models under three in-context prompting paradigms—Chain-of-Thought, Tree-of-Thought and self-evaluation feedback loops. These initial experiments revealed significant performance declines as graph complexity grows, underscoring the limits of the common models architectures. Based on this observation, we considered two state-of-the-art reasoning-specialist families—OpenAI's *o3-mini* and *DeepSeek-R1* and demonstrated that embedding the Peter Clark algorithm within a single unified prompt yields a new zero-shot baseline that already surpass prior results.

Leveraging those native gains, we then developed a modular in-context pipeline that decomposes the PC algorithm into four distinct stages, generates and validates intermediate artifacts at each step, and allows reasoning traces to compound iteratively across prompts. This structured approach yields up to a three-fold improvement in F1 over the initial CORR2CAUSE baseline without any fine-tuning, thanks to (i) decomposition of the PC algorithm into separate stages, (ii) generation of rich intermediate artifacts for early error detection, and (iii) amplification of the model's native iterative enrichment of its reasoning traces.

These benefits come with trade-offs. The multi-stage pipeline requires multiple sequential requests, increasing both token consumption and end-to-end latency. Future work will involve stage-wise exploration to balance computational cost and introduce optimizations into the pipeline. Moreover, our evaluation has thus far been limited to a single benchmark; extending it to other synthetic structural causal models and real-world corpora will be essential to establish robustness and generalizability, thereby unlocking the full potential of this approach.

References

Babu Shrestha, R., Malberg, S., and Groh, G. (2025). From causal parrots to causal prophets? towards sound causal reasoning with large language models. In Hämäläinen, M., Öhman, E., Bizzoni, Y., Miyagawa, S., and Alnajjar, K., editors, *Proceedings of the 5th International Conference on Natural Language Processing for Digital Humanities*, pages 319–333, Albuquerque, USA. Association for Computational Linguistics.

Bagheri, A., Alinejad, M., Bello, K., and Akhondi-Asl, A. (2024). C^2P: Featuring large language models with causal reasoning.

Ballon, M., Algaba, A., and Ginis, V. (2025). The relationship between reasoning and performance in large language models – o3 (mini) thinks harder, not longer.

Ban, T., Chen, L., Wang, X., and Chen, H. (2023). From query tools to causal architects: Harnessing large language models for advanced causal discovery from data.

Besta, M., Blach, N., Kubicek, A., Gerstenberger, R., Podstawski, M., Gianinazzi, L., Gajda, J., Lehmann, T., Niewiadomski, H., Nyczyk, P., and Hoefler, T. (2024). Graph of thoughts: Solving elaborate problems with large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16):17682–17690.

Besta, M., Memedi, F., Zhang, Z., Gerstenberger, R., Piao, G., Blach, N., Nyczyk, P., Copik, M., Kwaśniewski, G., Müller, J., Gianinazzi, L., Kubicek, A., Niewiadomski, H., O'Mahony, A., Mutlu, O., and Hoefler, T. (2025). Demystifying chains, trees, and graphs of thoughts.

Bondarenko, A., Wolska, M., Heindorf, S., Blübaum, L., Ngonga Ngomo, A.-C., Stein, B., Braslavski, P., Hagen, M., and Potthast, M. (2022). CausalQA: A benchmark for causal question answering. In Calzolari, N., Huang, C.-R., Kim, H., Pustejovsky, J., Wanner, L., Choi, K.-S., Ryu, P.-M., Chen, H.-H., Donatelli, L., Ji, H., Kurohashi, S., Paggio, P., Xue, N., Kim, S., Hahm, Y., He, Z., Lee, T. K., Santus, E., Bond, F., and Na, S.-H., editors, *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3296–3308, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners.

Bühlmann, P., Peters, J., and Ernest, J. (2014). Cam: Causal additive models, high-dimensional order search and penalized regression. *The Annals of Statistics*, 42(6).

Chen, L., Ban, T., Wang, X., Lyu, D., and Chen, H. (2023). Mitigating prior errors in causal structure learning: Towards llm driven prior knowledge.

Chickering, D. M. (2002). Learning equivalence classes of bayesian-network structures. *J. Mach. Learn. Res.*, 2:445–498.

DeepSeek-AI (2025). Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning.

Geiger, D. and Pearl, J. (1990). On the logic of causal models* *this work was partially supported by the national science foundation grants #iri-8610155, "graphoids: A computer representation for dependencies and relevance in automated reasoning," and #iri-8821444, "probabilistic networks for automated reasoning.". In SHACHTER, R. D., LEVITT, T. S., KANAL, L. N., and LEMMER, J. F., editors, *Uncertainty in Artificial Intelligence*, volume 9 of *Machine Intelligence and Pattern Recognition*, pages 3–14. North-Holland.

Gendron, G., Rožanec, J. M., Witbrock, M., and Dobbie, G. (2024). Counterfactual causal inference in natural language with large language models.

Glymour, C., Zhang, K., and Spirtes, P. (2019). Review of causal discovery methods based on graphical models. *Frontiers in Genetics*, Volume 10 - 2019.

Harris, N. and Drton, M. (2013). Pc algorithm for nonparanormal graphical models. *Journal of Machine Learning Research*, 14(105):3365–3383.

Hoaglin, D. C., Mosteller, F., and (Editor), J. W. T. (2000). *Understanding Robust and Exploratory Data Analysis*. Wiley-Interscience, 1 edition.

Huang, B., Zhang, K., Lin, Y., Schölkopf, B., and Glymour, C. (2018). Generalized score functions for causal discovery. *KDD : proceedings. International Conference on Knowledge Discovery and Data Mining*, 2018:1551–1560.

Huang, S., Mamidanna, S., Jangam, S., Zhou, Y., and Gilpin, L. H. (2023). Can large language models explain themselves? a study of llm-generated self-explanations.

IBM (2025). Deepseek's reasoning ai shows power of small models, efficiently trained. `https://www.ibm.com/think/news/deepseek-r1-ai#:~:text=Reasoning%20models%20became%20the%20hot,a%20%E2%80%9Cchain%20of%20thought%E2%80%9D%20manner`. Accessed: May 24, 2025.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., and Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Jiang, Q., Gao, Z., and Karniadakis, G. E. (2025). Deepseek vs. chatgpt vs. claude: A comparative study for scientific computing and scientific machine learning tasks. *Theoretical and Applied Mechanics Letters*, 15(3):100583.

Jin, Z., Chen, Y., Leeb, F., Gresele, L., Kamal, O., LYU, Z., Blin, K., Gonzalez Adauto, F., Kleiman-Weiner, M., Sachan, M., and Schölkopf, B. (2023). Cladder: Assessing causal reasoning in language models. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems*, volume 36, pages 31038–31065. Curran Associates, Inc.

Jin, Z., Feder, A., and Zhang, K. (2022). CausalNLP tutorial: An introduction to causality for natural language processing. In El-Beltagy, S. R. and Qiu, X., editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pages 17–22, Abu Dubai, UAE. Association for Computational Linguistics.

Jin, Z., Liu, J., Lyu, Z., Poff, S., Sachan, M., Mihalcea, R., Diab, M., and Schölkopf, B. (2024). Can large language models infer causation from correlation?

Jiralerspong, T., Chen, X., More, Y., Shah, V., and Bengio, Y. (2024). Efficient causal graph discovery using large language models.

Joshi, N., Saparov, A., Wang, Y., and He, H. (2024). Llms are prone to fallacies in causal inference.

Kalisch, M. and Buehlmann, P. (2005). Estimating high-dimensional directed acyclic graphs with the pc-algorithm.

Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. Adaptive Computation and Machine Learning series. MIT Press.

Kıcıman, E., Ness, R., Sharma, A., and Tan, C. (2024). Causal reasoning and large language models: Opening a new frontier for causality.

Lauritzen, S. L. (1996). *Graphical models*, volume 17. Clarendon Press.

Le, H. D., Xia, X., and Chen, Z. (2025). Multi-agent causal discovery using large language models.

Liu, X., Wu, Z., Wu, X., Lu, P., Chang, K.-W., and Feng, Y. (2024). Are llms capable of data-based statistical and causal reasoning? benchmarking advanced quantitative reasoning with data.

Long, S., Schuster, T., and Piché, A. (2024). Can large language models build causal graphs?

Malinsky, D. and Danks, D. (2018). Causal discovery algorithms: A practical guide. *Philosophy Compass*, 13(1):e12470. e12470 10.1111/phc3.12470.

Marjanović, S. V., Patel, A., Adlakha, V., Aghajohari, M., BehnamGhader, P., Bhatia, M., Khandelwal, A., Kraft, A., Krojer, B., Lù, X. H., Meade, N., Shin, D., Kazemnejad, A., Kamath, G., Mosbach, M., Stańczak, K., and Reddy, S. (2025). Deepseek-r1 thoughtology: Let's think about llm reasoning.

Meek, C. (2013). Causal inference and causal explanation with background knowledge.

Meek, C. (2018). Complete Orientation Rules for Patterns.

Nogueira, A. R., Pugnana, A., Ruggieri, S., Pedreschi, D., and Gama, J. (2022). Methods and tools for causal discovery and causal inference. *WIREs Data Mining and Knowledge Discovery*, 12(2):e1449.

OpenAI (2024). Learning to reason with llms. `https://openai.com/index/learning-to-reason-with-llms/`. Accessed: May 24, 2025.

OpenAI (2025). Openai o3-mini: Pushing the frontier of cost-effective reasoning. `https://openai.com/research/openai-o3-mini`. Accessed: April 5, 2025.

OpenAI, Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., Avila, R., Babuschkin, I., Balaji, S., Balcom, V., Baltescu, P., Bao, H., Bavarian, M., Belgum, J., Bello, I., Berdine, J., Bernadett-Shapiro, G., Berner, C., Bogdonoff, L., Boiko, O., Boyd, M., Brakman, A.-L., Brockman, G., Brooks,

T., Brundage, M., Button, K., Cai, T., Campbell, R., Cann, A., Carey, B., Carlson, C., Carmichael, R., Chan, B., Chang, C., Chantzis, F., Chen, D., Chen, S., Chen, R., Chen, J., Chen, M., Chess, B., Cho, C., Chu, C., Chung, H. W., Cummings, D., Currier, J., Dai, Y., Decareaux, C., Degry, T., Deutsch, N., Deville, D., Dhar, A., Dohan, D., Dowling, S., Dunning, S., Ecoffet, A., Eleti, A., Eloundou, T., Farhi, D., Fedus, L., Felix, N., Fishman, S. P., Forte, J., Fulford, I., Gao, L., Georges, E., Gibson, C., Goel, V., Gogineni, T., Goh, G., Gontijo-Lopes, R., Gordon, J., Grafstein, M., Gray, S., Greene, R., Gross, J., Gu, S. S., Guo, Y., Hallacy, C., Han, J., Harris, J., He, Y., Heaton, M., Heidecke, J., Hesse, C., Hickey, A., Hickey, W., Hoeschele, P., Houghton, B., Hsu, K., Hu, S., Hu, X., Huizinga, J., Jain, S., Jain, S., Jang, J., Jiang, A., Jiang, R., Jin, H., Jin, D., Jomoto, S., Jonn, B., Jun, H., Kaftan, T., Łukasz Kaiser, Kamali, A., Kanitscheider, I., Keskar, N. S., Khan, T., Kilpatrick, L., Kim, J. W., Kim, C., Kim, Y., Kirchner, J. H., Kiros, J., Knight, M., Kokotajlo, D., Łukasz Kondraciuk, Kondrich, A., Konstantinidis, A., Kosic, K., Krueger, G., Kuo, V., Lampe, M., Lan, I., Lee, T., Leike, J., Leung, J., Levy, D., Li, C. M., Lim, R., Lin, M., Lin, S., Litwin, M., Lopez, T., Lowe, R., Lue, P., Makanju, A., Malfacini, K., Manning, S., Markov, T., Markovski, Y., Martin, B., Mayer, K., Mayne, A., McGrew, B., McKinney, S. M., McLeavey, C., McMillan, P., McNeil, J., Medina, D., Mehta, A., Menick, J., Metz, L., Mishchenko, A., Mishkin, P., Monaco, V., Morikawa, E., Mossing, D., Mu, T., Murati, M., Murk, O., Mély, D., Nair, A., Nakano, R., Nayak, R., Neelakantan, A., Ngo, R., Noh, H., Ouyang, L., O'Keefe, C., Pachocki, J., Paino, A., Palermo, J., Pantuliano, A., Parascandolo, G., Parish, J., Parparita, E., Passos, A., Pavlov, M., Peng, A., Perelman, A., de Avila Belbute Peres, F., Petrov, M., de Oliveira Pinto, H. P., Michael, Pokorny, Pokrass, M., Pong, V. H., Powell, T., Power, A., Power, B., Proehl, E., Puri, R., Radford, A., Rae, J., Ramesh, A., Raymond, C., Real, F., Rimbach, K., Ross, C., Rotsted, B., Roussez, H., Ryder, N., Saltarelli, M., Sanders, T., Santurkar, S., Sastry, G., Schmidt, H., Schnurr, D., Schulman, J., Selsam, D., Sheppard, K., Sherbakov, T., Shieh, J., Shoker, S., Shyam, P., Sidor, S., Sigler, E., Simens, M., Sitkin, J., Slama, K., Sohl, I., Sokolowsky, B., Song, Y., Staudacher, N., Such, F. P., Summers, N., Sutskever, I., Tang, J., Tezak, N., Thompson, M. B., Tillet, P., Tootoonchian, A., Tseng, E., Tuggle, P., Turley, N., Tworek, J., Uribe, J. F. C., Vallone, A., Vijayvergiya, A., Voss, C., Wainwright, C., Wang, J. J., Wang, A., Wang, B., Ward, J., Wei, J., Weinmann, C., Welihinda, A., Welinder, P., Weng, J., Weng, L., Wiethoff, M., Willner, D., Winter, C., Wolrich, S., Wong, H., Workman, L., Wu, S., Wu, J., Wu, M., Xiao, K., Xu, T., Yoo, S., Yu, K., Yuan, Q., Zaremba, W., Zellers, R., Zhang, C., Zhang, M., Zhao, S., Zheng, T., Zhuang, J., Zhuk, W., and Zoph, B. (2024). Gpt-4 technical report.

Panickssery, A., Bowman, S. R., and Feng, S. (2024). Llm evaluators recognize and favor their own generations.

Pearl, J. (1988). The morgan kaufmann series in representation and reasoning. In *Probabilistic Reasoning in Intelligent Systems*, page i. Morgan Kaufmann, San Francisco (CA).

Pearl, J. (2009). *Causality: Models, Reasoning and Inference*. Cambridge University Press, USA, 2nd edition.

Pearl, J. (2010). An introduction to causal inference. *The International Journal of Biostatistics*, 6(2).

Peters, J., Janzing, D., and Schlkopf, B. (2017). *Elements of Causal Inference: Foundations and Learning Algorithms*. The MIT Press.

Ramsey, J., Glymour, M., Sanchez-Romero, R., and Glymour, C. (2017). A million variables and more: the fast greedy equivalence search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images. *International Journal of Data Science and Analytics*, 3.

Ren, J., Zhao, Y., Vu, T., Liu, P. J., and Lakshminarayanan, B. (2023). Self-evaluation improves selective generation in large language models.

Rohrer, J. (2018). Thinking clearly about correlations and causation: Graphical causal models for observational data. *Advances in Methods and Practices in Psychological Science*, 1:251524591774562.

Russo, F. and Toni, F. (2025). Shapley-pc: Constraint-based causal structure learning with a shapley inspired framework.

Scheines, R. (2005). An introduction to causal inference. Technical report, Carnegie Mellon University.

Spirtes, P., Glymour, C., Scheines, R., and Heckerman, D. (2001). *Causation, Prediction, and Search*. Adaptive Computation and Machine Learning. The MIT Press, Cambridge, MA, 2 edition. Special Collection: CogNet.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. (2023). Llama: Open and efficient foundation language models.

Triantafillou, S. and Tsamardinos, I. (2016). Score based vs constraint based causal learning in the presence of confounders.

Tsai, A. Y., Kraft, A., Jin, L., Cai, C., Hosseini, A., Xu, T., Zhang, Z., Hong, L., Chi, E. H., and Yi, X. (2024). Leveraging llm reasoning enhances personalized recommender systems.

Tsamardinos, I., Brown, L., and Aliferis, C. (2006). The max-min hill-climbing bayesian network structure learning algorithm. *Machine Learning*, 65:31–78.

Tseng, Y.-M., Huang, Y.-C., Hsiao, T.-Y., Chen, W.-L., Huang, C.-W., Meng, Y., and Chen, Y.-N. (2024). Two tales of persona in llms: A survey of role-playing and personalization.

Uhler, C., Raskutti, G., Bühlmann, P., and Yu, B. (2013). Geometry of the faithfulness assumption in causal inference. *The Annals of Statistics*, 41(2):436 – 463.

Vashishtha, A., Reddy, A. G., Kumar, A., Bachu, S., Balasubramanian, V. N., and Sharma, A. (2025). Causal order: The key to leveraging imperfect experts in causal inference.

Vowels, M. J., Camgoz, N. C., and Bowden, R. (2021). D'ya like dags? a survey on structure learning and causal discovery.

Wan, G., Lu, Y., Wu, Y., Hu, M., and Li, S. (2025). Large language models for causal discovery: Current landscape and future directions.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. (2023). Self-consistency improves chain of thought reasoning in language models.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., and Zhou, D. (2023). Chain-of-thought prompting elicits reasoning in large language models.

Willig, M., Zečević, M., Dhami, D. S., and Kersting, K. (2022). Can foundation models talk causality?

Woodward, J. F. (2003). *Making Things Happen: A Theory of Causal Explanation*. Oxford University Press, New York.

Xu, F., Hao, Q., Zong, Z., Wang, J., Zhang, Y., Wang, J., Lan, X., Gong, J., Ouyang, T., Meng, F., Shao, C., Yan, Y., Yang, Q., Song, Y., Ren, S., Hu, X., Li, Y., Feng, J., Gao, C., and Li, Y. (2025). Towards large reasoning models: A survey of reinforced reasoning with large language models.

Xu, W., Zhu, G., Zhao, X., Pan, L., Li, L., and Wang, W. Y. (2024). Pride and prejudice: Llm amplifies self-bias in self-refinement.

Yang, L., Shirvaikar, V., Clivio, O., and Falck, F. (2024). A critical review of causal reasoning benchmarks for large language models.

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., and Narasimhan, K. (2023). Tree of thoughts: Deliberate problem solving with large language models.

Yenduri, G., M, R., G, C. S., Y, S., Srivastava, G., Maddikunta, P. K. R., G, D. R., Jhaveri, R. H., B, P., Wang, W., Vasilakos, A. V., and Gadekallu, T. R. (2023). Generative pre-trained transformer: A comprehensive review on enabling technologies, potential applications, emerging challenges, and future directions.

Zanga, A. and Stella, F. (2023). A survey on causal discovery: Theory and practice.

Zečević, M., Willig, M., Dhami, D. S., and Kersting, K. (2023). Causal parrots: Large language models may talk causality but are not causal.

Zhang, C., Deng, Y., Lin, X., Wang, B., Ng, D., Ye, H., Li, X., Xiao, Y., Mo, Z., Zhang, Q., and Bing, L. (2025). 100 days after deepseek-r1: A survey on replication studies and more directions for reasoning language models.

Zhang, Y., Zhang, Y., Gan, Y., Yao, L., and Wang, C. (2024a). Causal graph discovery with retrieval-augmented generation based large language models.

Zhang, Z., Zhang, A., Li, M., and Smola, A. (2022). Automatic chain of thought prompting in large language models.

Zhang, Z., Zhang, A., Li, M., Zhao, H., Karypis, G., and Smola, A. (2024b). Multimodal chain-of-thought reasoning in language models.

Zheng, X., Aragam, B., Ravikumar, P., and Xing, E. P. (2018). Dags with no tears: Continuous optimization for structure learning.

A Formal Definitions of Causal Concepts

In this appendix, we explain the core causal concepts used in both the CORR2CAUSE benchmark and the field of causal discovery. CORR2CAUSE turns each question into a binary decision task: it ask whether a certain pattern appears in a causal graph, based on four fundamental relationships that are described in Figure 8. For example, one question might read: *'Hypothesis: There exists at least one collider (i.e., common effect) of C and D.'*.

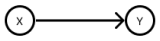
Conditional independence in Peter-Clark is determined by the appliance of d-separation criterion. Let G be a DAG and let X, Y be two nodes, with Z an arbitrary set of nodes. A path π between X and Y is *blocked by* Z if at least one of the following holds for some consecutive triple $A-B-C$ on π:

- **Chain or fork:** There exists B such that $A \to B \to C$ or $A \leftarrow B \to C$, and $B \in Z$.
- **Collider:** There exists B such that $A \to B \leftarrow C$, and neither B nor any descendant of B lies in Z.

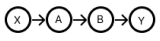
Then X and Y are *d-separated* by Z, written

$$X \perp\!\!\!\perp_d Y \mid Z,$$

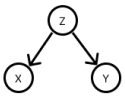
if every path between them is blocked by Z. In the PC algorithm, under the causal faithfulness assumption, testing $X \perp\!\!\!\perp Y \mid Z$ implies removal of the edge $X-Y$ from the skeleton.



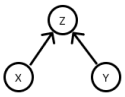
Parent–child: X is a *parent* of Y if there is a directed edge $X \to Y$ in the DAG.



Ancestor–descendant: X is an *ancestor* of Y if there exists a directed path $X \to \cdots \to Y$.



Confounder (common cause): Z is a *confounder* for X and Y if Z is a common parent of both.



Collider (common effect): Nodes X and Y form a *collider* at Z if $X \to Z \leftarrow Y$ and X, Y are not adjacent.

Figure 8: Graphical and formal definitions of the four fundamental causal relationships underpinning the CORR2CAUSE benchmark. These primitives form the basis of each natural-language hypothesis evaluated by our models.

B Empirical Evaluation Environment

The empirical evaluation was conducted using two access modes. Self-hosted experiments on GPU clusters used the 70B parameter models for both baseline and methodology development, while larger-scale runs leveraged third-party APIs.

GPU Cluster Settings for Self-Hosted Models

LLMs: The *LlaMa3.3-70B-Instruct*[4] version served as our conventional baseline. The *DeepSeek-R1-Distill-Llama-70B*[5]—a 70B parameter version of open-source *DeepSeek-R1* distilled from the *LlaMa3.3-70B* was our self-hosted reasoning model.

Compute node: $2\times$ NVIDIA H100-94 GB GPUs, 32 vCPUs, 360 GB RAM.

Inference settings: We perform inference in batches of four examples to maximize GPU utilization. Temperature parameter is listed in Table 4.

Runtime: Running the zero-shot baseline over the full $1,162$-sample test split requires about 47 GPU-hours per model. The four-stage pipeline takes roughly 220 GPU-hours on the same split. We ran three independent baseline experiments and one pipeline experiment, for a combined total of about 360 GPU-hours. We do not include additional research runs in this tally.

API Hosted Models

LLMs: We accessed two large reasoning LLMs via paid APIs: 685B version of *DeepSeek-R1*[6] and OpenAI's *o3-mini*[7].

Inference Settings: All *DeepSeek-R1* calls used the temperature values reported in Table 4. OpenAI's *o3-mini* does not expose a temperature parameter, instead it provides a `reasoning_effort` control, which we left at its default.

Token Usage: We quantify runtime by total tokens consumed. For the zero-shot baseline, *DeepSeek-R1* used ˜8 k tokens per sample and *o3-mini* ˜4 k tokens. Under the four-stage pipeline, *o3-mini* consumed ˜11 k tokens per sample and *DeepSeek-R1* over 25 k tokens. Overall, *o3-mini* experiments totaled ˜18 M tokens (input $+$ output), while *DeepSeek-R1* totaled ˜53 M tokens.

C Conventional LLMs Prompt Templates

This appendix lists the prompts used in the conventional-LLM experiments. Each subsection presents the exact `system` and `user` messages sent to the model, as described in Section 5.

[4] `https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct`
[5] `https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Llama-70B`
[6] `https://api-docs.deepseek.com/`
[7] `https://platform.openai.com/docs/models/o3-mini`

Chain-of-Thought

Listing 1: System prompt for the Chain-of-Thought template used to evaluate conventional LLMs on the CORR2CAUSE benchmark.

```
You are an expert in causal inference and data analysis, proficient in
applying the PC (Peter-Clark) algorithm.
Follow these steps in the provided order to respond accurately:

Step 1: Read the Data
- Identify extracted nodes and their correlations.
- Extract and list ALL marginal and conditional independence statements.

Step 2: Initialize the Graph
- Create edges between all correlated node pairs.
- List connections for each node.

Step 3: Systematic Independence Testing
- For EACH edge in the initial graph:
  a. Test against EACH independence statement INDIVIDUALLY
  b. Document your reasoning for each test
  c. Decide whether to keep or remove the edge
- Show your work using EXACTLY this format:
  * Edge X -- Y:
    - Testing against statement 1: [exact statement] $\rightarrow$ [reasoning]
$\rightarrow$ [consistent/inconsistent]
    - Testing against statement 2: [exact statement] $\rightarrow$ [reasoning]
$\rightarrow$ [consistent/inconsistent]
    - [Continue for ALL independence statements]
    - Decision: [keep/remove] edge X -- Y

Step 4: Compile the Causal Undirected Skeleton
- Construct the final graph structure
- List each node with its connected nodes.
- Ensure all applied independencies are reflected accurately.

CRITICAL REMINDER ABOUT d-SEPARATION:
- An edge X -- Y should be REMOVED if and only if there exists ANY
independence statement indicating that X and Y are independent (either
marginally or conditional on some set of variables).
- The absence of an independence statement means X and Y are dependent and the
edge should be KEPT.
- Do not assume independencies that are not explicitly stated in the premise

IMPORTANT: The output format of Step 4 must match EXACTLY the following
example:
Step 4: Compile the Causal Undirected Skeleton
In this graph:
  - Node A is connected to nodes B, C.
  - Node B is connected to nodes A, C.
  - Node C is connected to nodes A, B.
```

Listing 2: Three-shot user prompt for the Chain-of-Thought template used to evaluate conventional LLMs on the Corr2Cause benchmark.

```
Example 1:

Task Description: Given the provided premise, apply the PC algorithm following
the described step-by-step instructions to compute the causal undirected
skeleton.
Premise: Suppose there is a closed system of 4 variables, A, B, C and D. All
the statistical relations among these 4 variables are as follows: A correlates
with C. A correlates with D. B correlates with C. B correlates with D. C
correlates with D. However, A is independent of B. A and D are independent
given B and C. A and D are independent given C. B and D are independent given
A and C. B and D are independent given C.
Answer:
Step 1: Read the Data

- Extracted variables: A, B, C, D
- Correlations: (A, C), (A, D), (B, C), (B, D), (C, D)
- Marginal Independencies: (A, B)
- Conditional Independencies:
  - (A, D) are independent given B, C
  - (A, D) are independent given C
  - (B, D) are independent given A, C
  - (B, D) are independent given C

Step 2: Initialize the Graph

- Created edges between all correlated variable pairs.
- Initial edges: {(A, C), (A, D), (B, C), (B, D), (C, D)}

Step 3: Apply Marginal Independencies

- No edges removed in this step.

Step 4: Apply Conditional Independencies

- Because A and D are independent given B, C, there is no edge between A and D.
- Because B and D are independent given A, C, there is no edge between B and D.

Step 5: Compile the Causal Undirected Skeleton
In this graph:
  - Node A is connected to nodes C.
  - Node B is connected to nodes C.
  - Node C is connected to nodes D.
  - Node D has no connections.

Example 2:

Task Description: Given the provided premise, apply the PC algorithm following
the described step-by-step instructions to compute the causal undirected
```

skeleton.
Premise: Suppose there is a closed system of 4 variables, A, B, C and D. All the statistical relations among these 4 variables are as follows: A correlates with C. A correlates with D. B correlates with C. B correlates with D. C correlates with D. However, A is independent of B. A and D are independent given B and C. A and D are independent given C. B and D are independent given A and C. B and D are independent given C.
Answer:
Step 1: Read the Data

- Extracted variables: A, B, C, D
- Correlations: (A, C), (A, D), (B, C), (B, D), (C, D)
- Marginal Independencies: (A, B)
- Conditional Independencies:
  - (A, D) are independent given B, C
  - (A, D) are independent given C
  - (B, D) are independent given A, C
  - (B, D) are independent given C

Step 2: Initialize the Graph

- Created edges between all correlated variable pairs.
- Initial edges: {(A, C), (A, D), (B, C), (B, D), (C, D)}

Step 3: Apply Marginal Independencies

- No edges removed in this step.

Step 4: Apply Conditional Independencies

- Because A and D are independent given B, C, there is no edge between A and D.
- Because B and D are independent given A, C, there is no edge between B and D.

Step 5: Compile the Causal Undirected Skeleton
In this graph:
  - Node A is connected to nodes C.
  - Node B is connected to nodes C.
  - Node C is connected to nodes D.
  - Node D has no connections.

Example 3:

Task Description: Given the provided premise, apply the PC algorithm following the described step-by-step instructions to compute the causal undirected skeleton.
Premise: Suppose there is a closed system of 4 variables, A, B, C and D. All the statistical relations among these 4 variables are as follows: A correlates with B. A correlates with C. A correlates with D. B correlates with C. B correlates with D. C correlates with D. However, B and C are independent given A. B and C are independent given A and D. B and D are independent given A. B

and D are independent given A and C. C and D are independent given A. C and D are independent given A and B.
Answer:
Step 1: Read the Data

- Extracted variables: A, B, C, D
- Correlations: (A, B), (A, C), (A, D), (B, C), (B, D), (C, D)
- Marginal Independencies:
- Conditional Independencies:
  - (B, C) are independent given A
  - (B, C) are independent given A, D
  - (B, D) are independent given A
  - (B, D) are independent given A, C
  - (C, D) are independent given A
  - (C, D) are independent given A, B

Step 2: Initialize the Graph

- Created edges between all correlated variable pairs.
- Initial edges: {(A, B), (A, C), (A, D), (B, C), (B, D), (C, D)}

Step 3: Apply Marginal Independencies

- No edges removed in this step.

Step 4: Apply Conditional Independencies

- Because B and C are independent given A, there is no edge between B and C.
- Because B and D are independent given A, there is no edge between B and D.
- Because C and D are independent given A, there is no edge between C and D.

Step 5: Compile the Causal Undirected Skeleton
In this graph:
  - Node A is connected to nodes B, D, C.
  - Node B has no connections.
  - Node C has no connections.
  - Node D has no connections.

Question:

Task Description: Given the provided premise, apply the PC algorithm following the described step-by-step instructions to compute the causal undirected skeleton.
Premise: Suppose there is a closed system of 4 variables, A, B, C and D. All the statistical relations among these 4 variables are as follows: A correlates with C. A correlates with D. B correlates with C. C correlates with D. However, A is independent of B. A and B are independent given D. B is independent of D. B and D are independent given A. B and D are independent given A and C. C and D are independent given A. C and D are independent given A and B.

```
Answer:
```

Tree-of-Thought

Listing 3: Step instructions for the four intermediate thought stages in the Tree-of-Thought template used to evaluate conventional LLMs on the Corr2Cause benchmark (shown in YAML format).

```
steps_instructions:
  1: |
    Read the Data
    - Identify extracted nodes and their correlations.
    - Note all independencies.
  2: |
    Initialize the Graph
    - Create edges between all correlated node pairs.
    - List connections for each node.
  3: |
    Apply Independencies
    - Remove edges based on independencies.
    - Specify which independencies led to each removal.
  4: |
    Compile the Causal Undirected Skeleton
    - List every variable (node) identified in the premise.
    - For each node, list all other nodes it is connected to based on the
correlations.
    - Exclude any edge that should be removed due to a declared independency.
    - Ensure the final graph reflects both:
        - All existing correlations as edges, and
        - The absence of edges where independencies apply.
    - Format the output in a clear, consistent manner, for example:
        In this graph:
        - Node A is connected to nodes B, C.
        - Node B is connected to node A.
        - Node C is connected to node A.
        - Node D has no connections.
    Compile the Causal Undirected Skeleton
    - List each node with its connected nodes.
    - Ensure all applied independencies are reflected accurately.
```

Listing 4: Step examples for the four intermediate thought stages in the Tree-of-Thought template used to evaluate conventional LLMs on the Corr2Cause benchmark (shown in YAML format).

```
steps_examples:
  1: |
    Example for Step 1 - Read the Data:
    - Extracted variables: A, B, C, D
    - Correlations: (A, B), (A, C), (A, D), (B, C), (B, D), (C, D)
    - Independencies:
      - (B, D) are independent given A
```

```
      - (B, D) are independent given A, C
      - (C, D) are independent given A
      - (C, D) are independent given A, B
  2: |
    Example for Step 2 - Initialize the Graph:
    - Created edges between all correlated variable pairs.
    - Initial edges: {(A, B), (A, C), (A, D), (B, C), (B, D), (C, D)}
  3: |
    Example for Step 3 - Apply Independencies:
    - Because B and D are independent given A, there is no edge between B and
D.
    - Because C and D are independent given A, there is no edge between C and
D.
  4: |
    Example for Step 4 - Compile the Causal Undirected Skeleton:
    In this graph:
      - Node A is connected to nodes B, D, C.
      - Node B is connected to nodes C.
      - Node C has no connections.
      - Node D has no connections.
```

Listing 5: System prompt for the Tree-of-Thought template used to evaluate conventional LLMs on the CORR2CAUSE benchmark.

```
You are an expert in causal inference and data analysis, proficient in
applying the PC (Peter-Clark) algorithm.
Your overall task is to extract the causal undirected skeleton from a given
premise by following a multi-step process.

When generating candidate thought steps, focus ONLY on the current step as
indicated by the instructions and examples provided.
Do not include information or steps beyond the one requested.
```

Listing 6: Generation prompt for the Tree-of-Thought template used to evaluate conventional LLMs on the CORR2CAUSE benchmark.

```
Current state:
{current_state}

Next step instruction (focus exclusively on this step):
{next_step_instruction}

Step example (for guidance, only for this step):
{next_step_example}

IMPORTANT:
- Your response MUST ONLY address the above step.
- DO NOT generate any content or steps beyond what is requested for this step.
```

Listing 7: Evaluation prompt for the Tree-of-Thought template used to evaluate conventional LLMs on the CORR2CAUSE benchmark.

```
Given the following chain-of-thought:
```

```
{previous_state}

And the following candidate step:
{recent_step}

Please evaluate the candidate step with respect to these constraints:
{instruction_constraints}

Consider the following criteria:
- Does the candidate step correctly implement the change(s) specified?
- Are all nodes and edges (or removals) consistent with the overall
constraints?
- Is the candidate step detailed enough to reflect the intended modifications?

Rate on a scale from 0 (poor) to 10 (excellent) how well the candidate step
satisfies the constraints.

IMPORTANT:
- Your response MUST be exactly one integer number between 0 and 10.
- Do NOT include any extra text, explanation, or additional characters.

For example, if you believe the state is excellent, simply respond with:
10
```

Self-Evaluation Feedback

Listing 8: Self-Evaluation prompt for the LLMs feedback loop template used to evaluate conventional LLMs on the CORR2CAUSE benchmark.

```
You are an expert in causal inference and graph theory with deep expertise in
the PC (Peter-Clark) algorithm. You are provided with a set of human-language
independence statements (the "Premise") and a causal undirected skeleton
computed by the PC algorithm. Your task is to verify that the graph's edges
are consistent with the independence and correlation information in the
Premise.

Key points:
- The PC algorithm starts with a fully connected graph and iteratively removes
edges when conditional independence is found.
- Focus on applying the independence statements correctly.
- Your output should concisely identify any inconsistencies and suggest the
necessary modifications.

You are an expert in causal inference analyzing an undirected causal graph.

Premise: Suppose there is a closed system of 5 variables, A, B, C, D and E.
All the statistical relations among these 5 variables are as follows: A
correlates with C. A correlates with D. A correlates with E. B correlates with
C. B correlates with D. B correlates with E. C correlates with D. C correlates
with E. D correlates with E. However, A is independent of B. A and D are
independent given B and C. A and D are independent given C. B and D are
```

```
independent given A and C. B and D are independent given A, C and E. B and D
are independent given C. B and E are independent given A and C. B and E are
independent given A, C and D.

Current edges in the graph:
- B -- E
- B -- C
- A -- D
- D -- E
- C -- D
- A -- E
- C -- E
- A -- C

For each edge, you must test its consistency against EVERY independence
statement. Follow this process systematically:

1. First, extract ALL independence statements from the premise, listing them
one by one.
2. For EACH edge in the graph:
   a. Test the edge against EACH independence statement
   b. Document your reasoning for each test
   c. Conclude whether each edge should remain or be removed
3. Check if any MISSING edges are implied by correlations but not contradicted
by independence statements

Structure your analysis as follows:

## Independence Statements
1. [Statement 1]
2. [Statement 2]
...

## Edge Analysis

Edge: X -- Y
- Test against statement 1: [reasoning] $\rightarrow$ [consistent/inconsistent]
- Test against statement 2: [reasoning] $\rightarrow$ [consistent/inconsistent]
- Decision: [keep/remove] edge X -- Y

[Repeat for each edge]

## Missing Edges Check
[Check for any correlations that imply edges should be added]

## Final Decision
{
    "edges_to_add": [["A", "B"], ...],
    "edges_to_remove": [["C", "D"], ...]
}
```

D Single-Prompt Baseline Template

We establish a zero-shot baseline by issuing the single-prompt template summarized in Table 3. The complete prompt text is given in Listing 9.

Listing 9: Single-prompt template for baseline evaluation on the Corr2Cause benchmark.

```
You are a scientist specializing in causal discovery algorithms, particularly
the Peter-Clark (PC) algorithm. You expertly apply correlation statements to
initialize graphs and use independence assertions to compute causal undirected
skeletons. You have the skill to leverage these skeletons and separation sets
to identify v-structures and then translate them into a maximally oriented
graph by applying Meek rules. You can also evaluate hypotheses about specific
causal relationships between variables, determining whether the provided
correlation and independence statements support those hypotheses.

You are provided with the following inputs:
- **Premise:** A set of statements containing observed correlations along with
both marginal and conditional independence relationships.
- **Hypothesis:** A statement that posits a particular causal relationship
between one or more variables. This claim could assert a direct causal link
(for example, 'E directly causes A') or an indirect causal relationship (for
example, 'B causes something else which causes A'). The hypothesis specifies
what causal connection is being proposed and is what you should evaluate based
on the given statistical relations and independence statements.

**Task:** Use the PC (Peter-Clark) algorithm to assess whether the provided
causal claim in the hypothesis is supported by the relationships and
independence statements in the premise.

**Key Principles:**
- Extract the causal undirected skeleton by interpreting the correlations and
conditional independencies.
- Identify v-structures (colliders) that arise from the undirected skeleton
and separation sets.
- Apply Meek rules to further orient remaining undirected edges.
- Evaluate the hypothesis based on the processed causal graph.

Please reason step by step, and after completing your analysis, provide your
final answer only for the hypothesis as a boolean value (either true or false)
in this exact JSON format:

'''json
{{
  "hypothesis_answer": true/false
}}
'''

**Inputs:**
Premise: {premise}
```

```
Hypothesis: {hypothesis}
```

E Stage-Wise Pipeline Prompt Templates

In this appendix, we provide the exact in-context prompts used at each stage of our modular PC–algorithm pipeline described in Section 6.4. These prompts were refined through a careful, iterative prompt engineering process: we drafted multiple variants and validated each in a sandbox using programmatic JSON-schema checks to ensure correct, schema-compliant outputs. Every prompt is accompanied by a dedicated parser module in our codebase to extract structured results.

Stage 1: Undirected Skeleton Extraction

Listing 10: Undirected skeleton discovery stage prompt template.

```
You are a scientist specializing in causal discovery algorithms, particularly
the Peter-Clark (PC) algorithm. Your expertise lies in using correlation
statements and independence assertions to initialize graphs and compute causal
undirected skeletons. In this stage, focus exclusively on analyzing the
provided correlation and independence information to accurately construct the
underlying undirected graph that represents the relationships among the
variables.

**Task:** Based on the given Premise, apply the PC (Peter-Clark) algorithm to
identify a causal undirected skeleton from the correlations and independence
statements.

**Key Principles:**
- Initially, assume all correlated variables are connected
- An edge X--Y should be removed if ANY independence statement shows X and Y
are independent (marginally or conditionally)
- Keep all edges unless contradicted by an explicit independence statement
- The final skeleton should reflect all independence relationships in the data

Your analysis should be thorough but focused on removing edges based on
independence statements.

**Required Output Format:**
  After completing your analysis, provide your final answer in this exact JSON
format:

‘‘‘json
{{
  "nodes": ["list", "of", "nodes"],
  "edges": [
    ["Node1", "Node2"],
    ["Node2", "Node3"]
```

```
  ]
}}
```

Inputs:
Premise: {premise}

Stage 2: V-Structure Identification

Listing 11: V-structure identification stage prompt template.

```
You are an expert in causal discovery, specializing in the Peter-Clark (PC)
algorithm. With a deep understanding of undirected skeletons and separation
sets, your task in this stage is to identify v-structures. Analyze the given
undirected skeleton along with the corresponding separation sets to pinpoint
the colliders that indicate potential directional causal relationships among
the variables.

**Task:** You are given a causal skeleton (an undirected graph produced by the
PC algorithm) and a Premise that contains independence statements. Your job is
to perform two distinct steps:

1. **Extraction of Independence Statements:**
  - **Parse the Premise:** Identify and extract all independence statements.
  - **Representation:** For each statement that indicates that a pair of
variables is independent (optionally given a conditioning set), represent it
as an entry in a dictionary. Use a key that is a sorted pair of variables
(e.g., "A,C") and set its value as a list of the conditioning variables (if
any).
2. **Identification of V-Structures (Colliders):**
  - **Candidate Identification:** Systematically consider every triple of
nodes (X,Z,Y) from the skeleton where:
    - There are edges X--Z and Y--Z in the skeleton.
    - There is no direct edge between X and Y (i.e., they are non-adjacent).
  - **Verification via Separation Test:**
    For each candidate triple, check the separation (independence) information:
      - **Valid V-Structure:** Include [X,Z,Y] only if for the pair (X,Y) (as
found in the independence statements) the corresponding separation set does
not contain Z.
     - **Systematic Check:** Ensure that every candidate triple is evaluated
using the criteria of non-adjacency, common neighbor, and the separation test
to avoid false positives (including a Z that appears in the separation set)
and false negatives (omitting any valid candidate).

**Required Output Format:**
After completing your analysis, provide your final answer in this exact JSON
format:
```json
{{
  "separation_sets": {{
    "A,C": ["B"],
```
```

```
    "A,D": [...],
    ...
  }},
  "v_structures": [
    ["X", "Z", "Y"],
    ["X2", "Z2", "Y2"],
    ...
  ]
}}
```

```
**Inputs:**
Premise: {premise}
Casual skeleton:
```json
{{
  "nodes": {nodes},
  "edges": {edges}
}}
```
```

Stage 3: Meek's Rules Orientation

Listing 12: Meek's rules orientation stage prompt template.

```
You are an expert in causal inference with deep knowledge of the PC
algorithm's Meek orientation rules. Your task is to convert a given undirected
causal skeleton into a partially directed acyclic graph (PDAG) by orienting as
many edges as possible while following these rules:

**Rules and Constraints:**
- **Cycle Avoidance:** Do not create any directed cycles.
- **V-Structure Preservation:** Maintain all given v-structures (i.e.,
collider configurations).
- **Independence Consistency:** Ensure all orientations comply with the
provided marginal and conditional independence relationships (see "Premise").
- **Edge Pool Restriction:** Only orient edges that exist in the provided
causal skeleton.
- **Conservative Orientation:** Only assign a direction to an edge if it is
uniquely compelled by a v-structure or through propagation via Meek rules. If
the orientation is ambiguous (i.e., both directions are equally supported),
leave the edge undirected.
- **Final PDAG Requirement:** The final PDAG must include exactly the edges
provided in the causal skeleton, with no additional orientations beyond what
is compelled by the evidence.

**Decision Flow:**
1. Identify and orient all edges that form the given v-structures.
2. Apply Meek rules to propagate any forced orientations.
3. For each remaining edge, determine if its orientation is uniquely dictated
by the rules. If not, leave it undirected.
```

```
**Example:**
If the edge A-B does not appear in any v-structure and both A -> B and B -> A
would satisfy the constraints, do not orient it; keep it as A-B.

**Required Output Format:**
After completing your analysis, provide your final answer in this exact JSON
format:
‘‘‘json
{{
  "final_graph": {{
    "directed_edges": [
      {{
        "from": "Node1",
        "to": "Node2"
      }},
      {{
        "from": "Node2",
        "to": "Node3"
      }}
    ],
    "undirected_edges": [
      ["Node3", "Node4"],
      ["Node5", "Node6"]
    ]
  }}
}}
 ‘‘‘

**Inputs:**
Premise: {premise}
(Contains the relevant marginal and conditional independence information.)

V-Structures: {v_structures}
(A list of collider patterns that must be preserved.)

Casual skeleton:
‘‘‘json
{{
  "nodes": {nodes},
  "edges": {edges}
}}
‘‘‘
```

Stage 4: Hypothesis Evaluation

Listing 13: Hypothesis evaluation stage prompt template .

```
You are a specialist in causal discovery algorithms, particularly the
Peter-Clark (PC) algorithm, with a proven ability to evaluate complex causal
relationships. In this final stage, your task is to assess a specific
```

hypothesis regarding causal relationships between variables. Based on the constructed and oriented graph-derived from correlation statements, independence assertions, v-structures, and the application of Meek rules-determine whether the evidence supports or contradicts the proposed causal hypothesis.

Task: Evaluate whether the given causal hypothesis is supported by the provided causal graph.

Context:
You are given a causal graph that represents a Markov equivalence class (a CPDAG) derived from the above premises. This graph includes:
- **Directed edges:** These are relationships that are unambiguously oriented.
- **Undirected edges:** These represent ambiguous relationships where multiple orientations (across equivalent DAGs) are possible.

IMPORTANT: When evaluating the hypothesis, consider only those completions (i.e., fully oriented DAGs) that respect the above premises, including the separation sets. Do not allow any edge orientations that would violate these conditional independence statements (for example, avoid orienting the ambiguous edges as A -> B and C -> B simultaneously if that contradicts A _| C | B).

Your task is to determine if the specified hypothesis holds in every valid DAG within this equivalence class. The hypothesis is considered supported (true) only if it is true in all valid completions; if it holds in some but not all, then the answer should be false.

Required Output Format:
Provide your conclusion as a JSON object with a single boolean field:

```json
{{
  "hypothesis_answer": true/false
}}
```

Inputs:
Premise: {premise}
Provided Graph:
```json
{{
  "nodes": {nodes},
  "directed_edges": {directed_edges},
  "undirected_edges": {undirected_edges}
}}
```
Hypothesis: {hypothesis}

F Standard Deviation and Confidence Interval Report

For the single-prompt baseline experiment (Section 6.3), we report the bootstrap mean F1 for the DeepSeek-R1 API. Using five bootstrap samples ($R = 5$, $B = 1000$), we obtain a mean F1 of 0.6527 (standard deviation 0.0102) with a 95% confidence interval of $[0.6461, 0.6616]$. We believe that these results are representative of the remaining experiments in Table 4; however, due to the high cost of additional runs and budget constraints, we only performed the full bootstrap analysis on the baseline experiment.

G Implementation Overview: Architecture Diagram

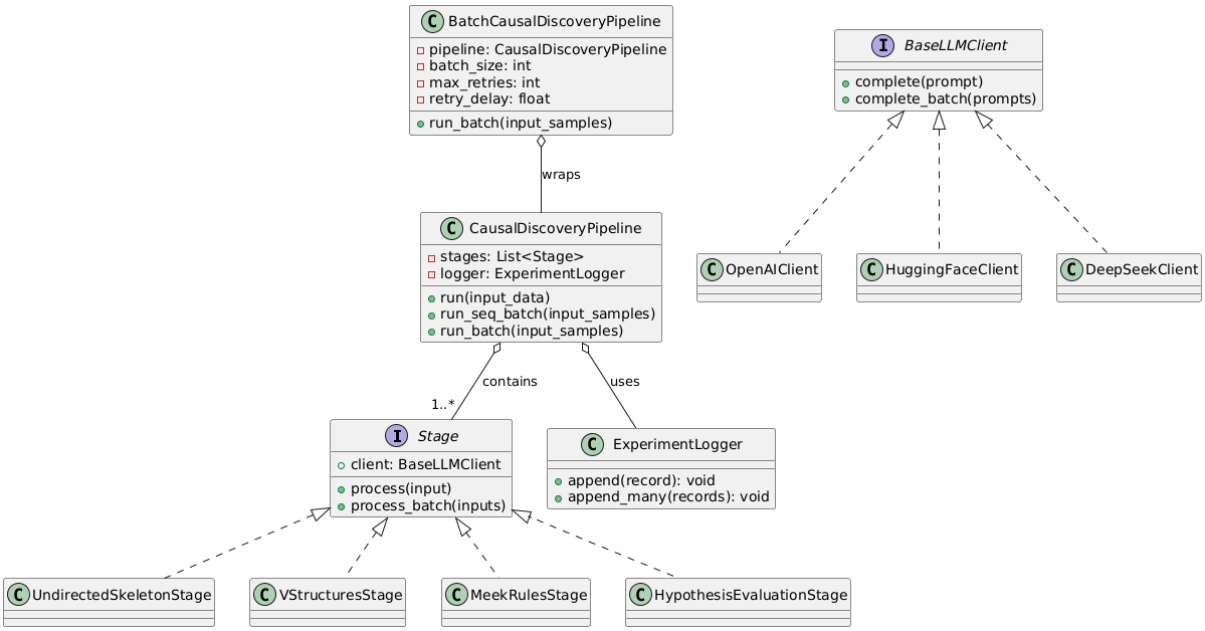


Figure 9: UML class diagram of the Python causal-discovery pipeline implementation. The diagram illustrates the two core interfaces—`Stage` and `BaseLLMClient`, their concrete subclasses for each pipeline stage and LLM backend, the orchestrating `CausalDiscoveryPipeline` (with its batched wrapper), and the `ExperimentLogger`, which records all inputs, outputs, and token usage.

Figure 9 shows a UML class diagram of our Python implementation of the five-stage causal-discovery pipeline.

Stage interface (bottom left): The Stage interface declares the methods `process` and `process_batch` and holds a reference to a `BaseLLMClient`. Each concrete subclass (UndirectedSkeletonStage, VStructuresStage, MeekRulesStage, HypothesisEvaluationStage) provides its own `prompt_template` and implements stage-specific JSON extraction logic.

LLM clients (right): The `BaseLLMClient` interface has three concrete implementations: OpenAIClient, HuggingFaceClient, and DeepSeekClient. New backend can be integrated by supplying a client that implements the required `complete` and `complete_batch` methods.

Pipeline orchestration (top left): `CausalDiscoveryPipeline` orchestrates a sequence of `Stage` instances and records every result through an `ExperimentLogger`. BatchCausalDiscoveryPipeline wraps the core pipeline with batching, retry logic, and delay handling. The `ExperimentLogger` writes each sample's outputs and metadata (e.g., token usage) to a CSV file for downstream analysis.

This design offers three key benefits: (i) extensibility—new stages can be added by subclassing `Stage`; (ii) modularity—LLM backend can be swapped by providing new `BaseLLMClient` implementations; (iii) reproducibility—every sample's token usage and intermediate artifacts are logged in a uniform CSV file.