Causal DAG Summarization (Technical Report)

Anna Zeng
annazeng@mit.edu
CSAIL, MIT

Michael Cafarella
michjc@csail.mit.edu
CSAIL, MIT

Batya Kenig
batyak@technion.ac.il
Technion

Markos Markakis
markakis@mit.edu
CSAIL, MIT

Brit Youngmann
brity@technion.ac.il
Technion

Babak Salimi
bsalimi@ucsd.edu
University of California, San Diego

ABSTRACT

Causal inference aids researchers in discovering cause-and-effect relationships, leading to scientific insights. Accurate causal estimation requires identifying confounding variables to avoid false discoveries. Pearl's causal model uses causal DAGs to identify confounding variables, but incorrect DAGs can lead to unreliable causal conclusions. However, for high dimensional data, the causal DAGs are often complex beyond human verifiability. Graph summarization is a logical next step, but current methods for general-purpose graph summarization are inadequate for causal DAG summarization. This paper addresses these challenges by proposing a causal graph summarization objective that balances graph simplification for better understanding while retaining essential causal information for reliable inference. We develop an efficient greedy algorithm and show that summary causal DAGs can be directly used for inference and are more robust to misspecification of assumptions, enhancing robustness for causal inference. Experimenting with six real-life datasets, we compared our algorithm to three existing solutions, showing its effectiveness in handling high-dimensional data and its ability to generate summary DAGs that ensure both reliable causal inference and robustness against misspecifications.

1 INTRODUCTION

Causal inference is central to informed decision-making in economics, sociology, medicine, and in helping analysts unravel complex cause-effect relationships [30, 43, 104]. It has become increasingly critical in machine learning, where it supports algorithmic fairness [89], data debiasing [116, 116, 117], explainable AI [11, 28, 63, 64], and enhanced robustness [53, 90, 101]. Causal inference has also become a major theme in recent data management research [13, 58, 60, 83, 88], integrating causality into data management tasks such as finding input responsibilities toward query answers [58–60], explaining for query results [84, 87, 110, 111], data discovery [27, 112], data cleaning [78, 89], hypothetical reasoning [26], and large system diagnostics [7, 29, 36, 55].

Drawing causal conclusions from data fundamentally hinges on access to background knowledge and assumptions, as data alone cannot establish causality [73, 86]. A principled way to encode such background knowledge is through Causal Directed Acyclic Graphs (DAGs) [73]. These graphs explicitly represent assumed causal relationships, enabling systematic reasoning about interventions. Causal DAGs can be used together with graphical criteria such as the backdoor criterion, or in general, Pearl's *do*-calculus [73] to determine whether the effect of interventions can be answered using data and available background knowledge. If so, they help identify the right set of confounding variables to control for, ensuring sound causal inference given the background knowledge.

However, the soundness and robustness of causal inference hinges on the availability of high-quality causal DAGs, which are often not readily available. These DAGs are typically constructed using domain knowledge [16, 56, 105] or through causal discovery methods [20, 35, 94, 107, 118]. This elicitation process is costly, error-prone [68], and time-consuming. Causal discovery methods, while useful, are fundamentally restrictive as they identify a class of DAGs compatible with observed data rather than a singular, definitive model [35]. Moreover, existing discovery methods often do not perform well on real-world data and require significant human intervention for verification [21, 40, 97]. The problem is even worse for high-dimensional data, increasing the need for efficient methods to simplify and verify causal models while retaining essential information [70]. We illustrate this with an example:

Example 1. Consider the application of performance diagnosis for a cloud-based data warehouse service. Specifically, consider a dataset collected from the monitoring views in Amazon Redshift Serverless [8], including performance metrics and query-extracted features, such as the number of unique tables and columns referenced in the executed query. This dataset offers opportunities to answer crucial causal queries for optimizing performance. For example, understanding the impact of caching on latency (i.e., `Result Cache Hit` on `Elapsed Time`) can help tune caching mechanisms. Similarly, analyzing the effect of join complexity on the query planner's performance (i.e., `Num Joins` on `Plan Time`) can optimize query execution strategies. However, the necessary causal DAG to answer such questions is not readily available, and getting it right is non-trivial.

Figure 1 shows an example causal DAG covering variables from just one monitoring view [6] and a few query features, chosen for illustration. This is just a small part of the overall high dimensional dataset. Edges in causal DAGs represent potential cause-effect relationships. In our example, for instance, the edge from `Num Columns` to `Exec. Time` suggests that the number of columns referenced in a query may influence the query's execution time.

To answer the above causal queries, `Query Template` is a critical confounder that must be adjusted for because it influences both the performance metrics (e.g., `Elapsed Time`, `Plan Time`) and the analyzed mechanisms (e.g., `Result Cache Hit`, `Num Joins`). Failing to adjust for this variable can lead to biased estimations and incorrect conclusions. Hence, any possible misspecification in the causal DAG that would fail to identify this variable as a confounder would result in incorrect effect estimations. Such sensitivity to graph errors makes domain expert verification essential for each existing or missing edge. This task can be overwhelming, even in this small example with only 12 nodes, as it involves inspecting 66 potential edges, one per pair of nodes. In the full dataset, the number of variables would be much higher, further complicating the task. □

Graph summarization is a logical next step, as it reduces the number of nodes and edges, making it easier for users to verify and inspect causal DAGs in high-dimensional datasets. Graph summarization has been extensively studied, with state-of-the-art methods designed to efficiently generate concise representations aimed at minimizing reconstruction errors [47, 109], or facilitating accurate query answering [52, 95]. However, we argue that while general-purpose methods are adept at managing massive graphs, they are inadequate for summarizing causal graphs, a task that demands the preservation of causal information crucial for reliable inference.

In this paper, we propose a graph summarization technique tailored for causal inference. It simplifies high-dimensional causal DAGs into manageable forms without compromising essential causal information, thereby improving interpretability. Our approach introduces a causal DAG summarization objective, which balances simplifying the graph for enhanced comprehensibility and retaining essential causal information. Using our technique, one can summarize an initial causal DAG (constructed using partial domain knowledge or causal discovery) for simpler verification and elicitation. Additionally, the summary causal DAG can be directly used for causal inference and is more robust to misspecification of assumptions. Our approach thereby improves *interpretability*, *verifiability*, and *robustness* in causal inference, facilitating the adoption of these techniques in practice. We illustrate this with an example:

Example 2. Consider Fig. 2a, which shows the summary graph generated by SSumM [47] for the causal DAG of Fig. 1. SSumM is a top-performing general-purpose graph summarization method that effectively balances conciseness and reconstruction accuracy. However, the generated graph can no longer be interpreted as a causal DAG, since it exhibits cycles and self-loops. For example, computing the causal effect of `Num Joins` on `Plan time` is impossible due to the bidirectional edge between their cluster nodes. Other methods (e.g., [109]) exhibit similar weaknesses, making them unsuitable for summarizing causal DAGs. An in-depth comparison with another graph summarization method [102] is provided in Section 8. We show that although this method can be adapted to generate summary DAGs compatible with causal inference principles, it does not optimally preserve critical causal information, reducing the accuracy of the inference over the summary DAG.

In contrast, Fig.2b shows the 5-node summary DAG generated by our approach, which preserves critical causal information, offering a more interpretable summary that can be directly used for inference. This summary DAG makes it easier to verify the soundness of assumptions it encodes. Furthermore, this summary DAG is inherently more robust to misspecification, because our summarization process creates a summary DAG compatible with a *set* of possible initial DAGs. Hence, even if the original causal DAG missed an edge, our summarization algorithm can still create the necessary connections and maintain causal integrity. Using the summary DAG for inference intuitively leads to a more conservative set of confounders: it may lead to adjusting for redundant attributes, but they will only be ones that do not hurt the analysis. □

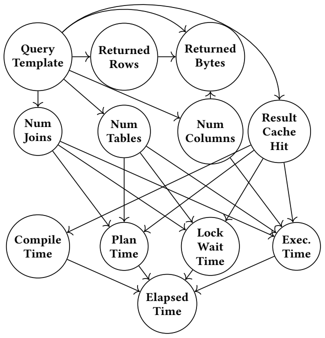
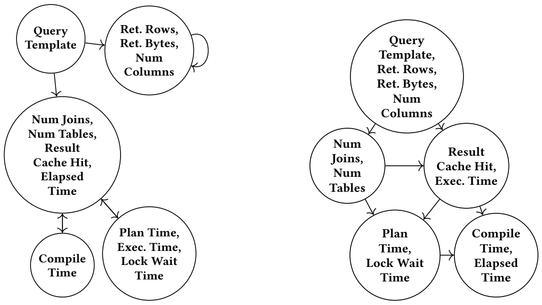


Figure 1: Example causal DAG


Query Template
Ret. Rows, Ret. Bytes, Num Columns
Num Joins, Num Tables, Result Cache Hit, Elapsed Time
Compile Time
Plan Time, Exec. Time, Lock Wait Time
Query Template, Ret. Rows, Ret. Bytes, Num Columns
Num Joins, Num Tables
Result Cache Hit, Exec. Time
Plan Time, Lock Wait Time
Compile Time, Elapsed Time

(a) Problematic Summary Graph **(b) Our Summary DAG**

Figure 2: 5-node summary graphs for the DAG in Fig. 1.

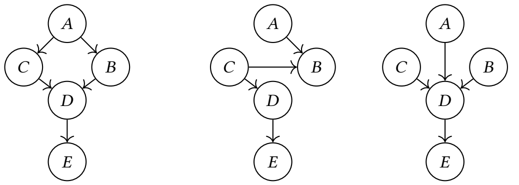
Our main contributions are summarized as follows.

Causal DAG Summarization. We introduce the problem of summarizing causal DAGs in a way that preserves their utility for reliable causal inference (Section 3). This necessitates preserving the causal information encoded in the input DAG. Causal DAGs encode information through missing edges, which imply Conditional Independence (CI) constraints. We therefore formalize causal DAG summarization as finding a summary DAG that preserves CI statements to the greatest extent possible, while meeting a node number constraint. We prove that this problem is NP-hard.

Summary Causal DAGs. We introduce the concept of *summary causal DAGs*, derived by grouping nodes within the original DAG via *node contractions*. Despite inherently leading to information loss, node contraction enables summary DAGs to compactly encapsulate potential causal DAGs from which the summary DAG could have originated. We show that contracting nodes is akin to adding edges to the input causal DAG. Based on this connection, we develop a sound and complete algorithm for identifying all CIs encoded by a summary DAG. This connection is crucial for utilizing summary causal DAGs for causal inference. (Section 4).

The CaGreS Algorithm. We devise an efficient heuristic greedy algorithm called CaGreS. A key feature of CaGreS is its approach to choosing which node pair to contract. This process is informed by the connection between node contraction and the addition of edges to the input DAG, prioritizing node pairs that add the fewest edges upon contraction. Additionally, CaGreS incorporates several optimizations, including caching mechanisms, making it a practical tool for generating summary causal DAGs (Section 5).

Causal Inference over Summary Causal DAGs We show that summary causal DAGs can be directly utilized for causal inference.



(a) $\mathcal{G}_1$ **(b)** $\mathcal{G}_2$ **(c)** $\mathcal{G}_3$

Figure 3: Three causal DAGs over the same set of nodes.

We establish that Pearl’s *do-calculus* framework [73], which provides a set of sound and complete rules for reasoning about the effects of interventions using causal DAGs, remains sound and complete for summary DAGs. By examining the connection between node contractions and the addition of edges, we offer clear insights into how these modifications affect the soundness and completeness of do-calculus within the framework of summary DAGs (Section 6).

Experimental Evaluation We demonstrate via a case study how summary DAGs offer robustness against errors in the input causal DAG (Section 7). We further conduct an extensive experimental study over six datasets demonstrating the effectiveness of CaGreS compared to three existing solutions and two possible variations of CaGreS. Our results demonstrate the robust quality of CaGreS. The results further show the efficiency of CaGreS in handling high-dimensional datasets and its ability to generate summary DAGs that ensure reliable inference (Section 8).

Related work is discussed in Section 9 and we conclude in Section 10. All proofs are provided in our technical report [3].

2 BACKGROUND

We consider a single-relation database over a schema $\mathbb{A}$. We use upper case letters to denote a variable from $\mathbb{A}$ and bold symbols for sets of variables. The broad goal of causal inference is to estimate the effect of an *exposure variable* $T \in \mathbb{A}$ on an *outcome variable* $O \in \mathbb{A}$. We use Pearl’s model for causal inference on observational data [73].

To get an unbiased estimate for the causal effect of the exposure T on the outcome O, one must mitigate the effect of *confounding variables*, i.e., variables that can affect the exposure assignment and outcome [73]. For instance, when estimating how query execution time affects the elapsed time, one would avoid a source of *confounding bias* by considering the number of columns and tables. Pearl’s model provides ways to account for confounding variables to get an unbiased causal estimate using *causal DAGs* [73]. Causal DAGs provide a simple way of representing causal relationships within a set of variables. A causal DAG $\mathcal{G}$ for the variables in $\mathbb{A}$ is a specific type of a Bayesian network and is formally defined as follows:

Causal DAG. A Bayesian network is a DAG $\mathcal{G}$ in which nodes represent random variables and edges express direct dependence between the variables. Each node X_i is associated with the conditional distribution $\mathbb{P}(X_i | \pi(X_i))$, where $\pi(X_i)$ is the set of parents of X_i in $\mathcal{G}$. The joint distribution over all variables $\mathbb{P}(X_1, \ldots, X_n)$, is given by the product of all conditional distributions. That is,

$$\mathbb{P}(X_1, \ldots, X_n) = \prod_{i=1}^{n} \mathbb{P}(X_i | \pi(X_i)) \tag{1}$$

A causal DAG is a Bayesian network where edges signify direct causal influence rather than statistical dependence. We say that X is a potential cause of Y if there is a directed path from X to Y.

Example 3. Fig. 3 shows three example causal DAGs. In $\mathcal{G}_1$, A is a potential cause of B, whereas in $\mathcal{G}_3$ it is not. □

***d*-Separation.** d-separation is a criterion in a causal DAG that determines whether two sets of nodes are conditionally independent, given a third set, by checking whether all paths between the sets are “blocked” based on specific structural rules. If two sets of nodes are d-separated, by definition it means that all paths connecting them are blocked by other nodes. Formally, a *trail* $t = (X_1, \ldots, X_n)$ is a sequence of nodes s.t. there is a a distinct edge between X_i and X_{i+1} for every i. That is, $(X_i \rightarrow X_{i+1}) \in \mathsf{E}(\mathcal{G})$ or $(X_i \leftarrow X_{i+1}) \in \mathsf{E}(\mathcal{G})$ for every i. A node X_i is said to be *head-to-head* with respect to t if $(X_{i-1} \rightarrow X_i) \in \mathsf{E}(\mathcal{G})$ and $(X_i \leftarrow X_{i+1}) \in \mathsf{E}(\mathcal{G})$. A trail $t = (X_1, \ldots, X_n)$ is *active* given $\mathbf{Z} \subseteq \mathcal{X}$ if (1) every X_i that is a head-to-head node with respect to t either belongs to $\mathbf{Z}$ or has a descendant in $\mathbf{Z}$, and (2) every X_i that is not a head-to-head node w.r.t. t does not belong to $\mathbf{Z}$. If a trail t is not active given $\mathbf{Z}$, then it is *blocked* given $\mathbf{Z}$ [73]. In a DAG, two sets of nodes $\mathbf{X}$ and $\mathbf{Y}$ are d-separated by a third set of nodes $\mathbf{Z}$ if all trails connecting $\mathbf{X}$ and $\mathbf{Y}$ are blocked by $\mathbf{Z}$.

Conditional Independence. Causal DAGs encode a set of Conditional Independence statements (CIs) that can be read off the graph using d-separation [73]. These statements describe the absence of an active trail between two sets of variables when conditioning on other variables. If two sets of nodes $\mathbf{X}$ and $\mathbf{Y}$ are d-separated by $\mathbf{Z}$, then $\mathbf{X}$ and $\mathbf{Y}$ are conditionally independent given $\mathbf{Z}$.

Example 4. Examples of CIs encoded in the causal DAG depicted in Fig. 3(a) include: $(B \perp\!\!\!\perp_d C \mid A)$, and $(D \perp\!\!\!\perp_d A \mid BC)$. □

CIs & Missing Edges. In causal DAGs, the information encoded by missing edges implies the set of CIs the DAG represents. Namely, removing edges can undermine the causal model as it implies CIs that do not necessarily hold in the distribution. On the other hand, existing edges indicate *potential* causal dependence. This implies that adding edges to a causal DAG, provided acyclicity is maintained, does not necessarily compromise validity [73].

The Recursive Basis. The *Recursive Basis* (RB) [33] for a causal DAG comprises a set of at most n CIs, signifying that each node is conditionally independent of its non-descendants nodes given its parents. This succinct set of CIs holds significance, as it can be used for constructing the causal DAG, and all other CIs encoded in the causal DAG can be deduced from it (see full details in [3]).

Formally, given a causal DAG $\mathcal{G}$, let $\langle X_1, \ldots, X_n \rangle$ denote a complete topological order over $\mathsf{V}(\mathcal{G})$. Equation 1 implicitly encodes a set of n CIs, called the RB for $\mathcal{G}$, defined as follows:

$$\Sigma_{\text{RB}}(\mathcal{G}) \overset{\text{def}}{=} \{(X_i \perp\!\!\!\perp X_1 \ldots X_{i-1} \setminus \pi(X_i) \mid \pi(X_i)) : i \in [n]\} \tag{2}$$

It has been shown [32, 33, 106] that both the semi-graphoid axioms [3] and d-separation are sound and complete for inferring CIs from the RB, which matches the CIs encoded by the causal DAG.

Example 5. Consider the causal DAG $\mathcal{G}_1$ in Fig. 3(a). In the nodes’ topological order, A precedes B and C, which in turn, precedes D. The last node is E. The RB of $\mathcal{G}_1$ is given in Table 1. Given the topological order over the nodes and the RB, $\mathcal{G}_1$ can be fully constructed. Further, any CI statement encoded in $\mathcal{G}_1$ can be implied from this RB by using the semi-graphoid axioms. □

ATE& do-Calculus. The *d*o-operator, a fundamental concept in causal inference, is used to denote interventions on variables in a causal model. It represents the intervention on a variable to observe the resulting change in an outcome variable while holding the external factors constant. In computing the *Average Treatment Effect* (ATE) [73], a popular measure of causal estimate, the *do*-operator is applied to represent the treatment assignment for treatment and control groups. The ATE quantifies the average causal effect of a treatment T on an outcome variable O in a population:

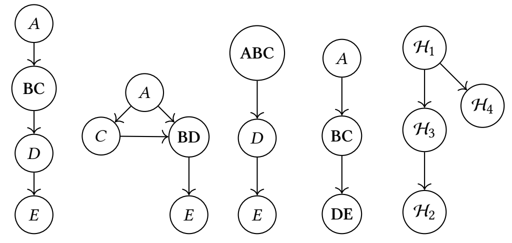
$$ATE(T, O) = \mathbb{E}[O \mid do(T = 1)] - \mathbb{E}[O \mid do(T = 0)] \qquad (3)$$

To compute the causal effect of T on O, it is crucial to identify and adjust for confounders. The backdoor criterion [73] provides a sufficient condition by identifying a set of variables $\mathbf{Z}$ that blocks all backdoor paths between T and O, enabling confounder adjustment within the causal DAG framework. However, it is part of the *do*-calculus system, an axiomatic framework designed for reasoning about interventions and their effects within causal models. The *do*-calculus comprises three rules that facilitate the substitution of probability expressions containing the *do*-operator with standard conditional probabilities [73]. It provides a systematic method for deriving causal relationships from observational data. Due to its soundness and completeness, the framework offers a broad toolkit for causal inference. Since these concepts are not directly used in this paper, we omit a detailed review.

3 PROBLEM FORMULATION

Our goal is to distill a causal DAG[1] into an interpretable summary by grouping nodes while preserving its utility for causal inference. Thus, the summary DAG should meet the following criteria:
Size Constraint: The summary DAG should be concise to reduce the cognitive load on analysts [14]. We therefore impose a size constraint to enhance the summary DAG's intelligibility, ensuring the core complexity of the original DAG is maintained.
Preserving Causal Information: The summary DAG must maintain the causal dependencies present in the original DAG: If variable A has a directed causal path to B in the original DAG, this relationship should be faithfully preserved in the summary DAG. The summary DAG should also preserve the CIs represented in the original DAG. If variables A and B are conditionally independent, this lack of dependence should be reflected in the summary DAG. Lastly, the summary DAG should not introduce any spurious conditional independencies that the original causal DAG does not imply.

Our objective is to preserve the utility of the summary causal DAG for causal inference. As mentioned, in causal DAGs, the information encoded by missing edges implies the set of CIs the DAG represents. Therefore, removing edges can undermine the causal model as it implies CIs that do not necessarily hold in the original DAG. On the other hand, existing edges indicate *potential* causal dependence. This implies that adding edges to a causal DAG, provided acyclicity is maintained, does not necessarily compromise validity. We, therefore, rigorously enforce conditions on the summary DAG to ensure that the directionality is faithfully preserved and assert that the summary DAG should preserve, to the greatest extent possible, a subset of the independence assumptions encoded in the original DAG. We show that, with these considerations, the summary causal DAG remains a viable tool for causal inference.



(a) $\mathcal{H}_1$ **(b)** $\mathcal{H}_2$ **(c)** $\mathcal{H}_3$ **(d)** $\mathcal{H}_4$ **(e) Order**

Figure 4: Summary causal DAGs for $\mathcal{G}_1$ and the partial order among them.

We first formalize the concept of a summary causal DAG, then rigorously formalize the problem of causal DAG summarization.

3.1 Summary Causal DAGs

A *summary graph* is obtained by applying *node contraction* operations [75]. The resulting graph retains the essential connectivity information of the original graph with a reduced number of nodes.

Given a graph $\mathcal{G}$, the contraction of a pair of nodes $U, V \in \mathsf{V}(\mathcal{G})$ is the operation that produces a graph $\mathcal{H}$ in which the two nodes U and V are replaced with a single node $\mathbf{C}=\{U, V\} \in \mathsf{V}(\mathcal{H})$, where $\mathbf{C}$ is now neighbors with nodes that U and V were originally adjacent to (edge directionality is preserved). If U and V were connected by an edge, the edge is removed upon contraction.

Definition 1 (Summary-DAG)**.** A summary DAG of a DAG $\mathcal{G}$ is a pair $(\mathcal{H}, f)$, where $\mathcal{H}$ is a DAG with nodes $\mathsf{V}(\mathcal{H})$, edges $\mathsf{E}(\mathcal{H})$, and $f : \mathsf{V}(\mathcal{G}) \to \mathsf{V}(\mathcal{H})$ is a function that partitions the nodes $\mathsf{V}(\mathcal{G})$ among the nodes $\mathsf{V}(\mathcal{H})$, such that: If $(U, V) \in \mathsf{E}(\mathcal{G})$, then $f(U)=f(V)$ or $(f(U), f(V)) \in \mathsf{E}(\mathcal{H})$. We define the inverse $f^{-1}: \mathsf{V}(\mathcal{H}) \to 2^{\mathsf{V}(\mathcal{G})}$ as follows: $f^{-1}(X) \overset{\text{def}}{=} \{V \in \mathsf{V}(\mathcal{G}) : f(V)=X\}$

To simplify the notations, we omit f whenever possible.

Example 6. Consider Fig. 3(a) which depicts a DAG $\mathcal{G}_1$. After contracting B and C, the resulting summary DAG $\mathcal{H}_1$ is displayed in Fig. 4(a). In $\mathcal{H}_1$, the nodes B and C have been contracted into the node $\mathbf{BC}$. Namely, $f(B) = f(C) = \mathbf{BC}$, and $f^{-1}(\mathbf{BC}) = \{B, C\}$. □

A causal DAG $\mathcal{G}$ is said to be *compatible* with a summary DAG $\mathcal{H}$, if, there exists a function f that partitions the nodes $\mathsf{V}(\mathcal{G})$ among the nodes $\mathsf{V}(\mathcal{H})$, such that: If $(U, V) \in \mathsf{E}(\mathcal{G})$, then $f(U)=f(V)$ or $(f(U), f(V)) \in \mathsf{E}(\mathcal{H})$. Namely, $\mathcal{H}$ is a summary DAG of $\mathcal{G}$.

Definition 2 (Compatibility)**.** Let $(\mathcal{H}, f)$ be a summary DAG. A DAG $\mathcal{G}$ is *compatible* with $\mathcal{H}$ if $\mathcal{H}$ is a summary DAG for $\mathcal{G}$. We use $\{\mathcal{G}_i\}_{\mathcal{H}}$ to denote the set of all causal DAGs compatible with $\mathcal{H}$.

We also use the term compatibility to describe the relationship between two causal DAGs sharing the same set of nodes, where the edges of one are fully contained in the set of edges of another graph. Let $\mathcal{G}$ be a causal DAG and let $\mathcal{G}'$ be a causal DAG where $\mathsf{V}(\mathcal{G})=\mathsf{V}(\mathcal{G}')$. We say that $\mathcal{G}'$ is a *supergraph* of $\mathcal{G}$ if $\mathsf{E}(\mathcal{G}) \subseteq \mathsf{E}(\mathcal{G}')$. In this case, we also say that $\mathcal{G}$ is *compatible* with $\mathcal{G}'$.

[1] For a discussion of other causal graph formats like mixed graphs, see [3]

Table 1: The recursive bases of the summary DAGs in Figure 4

Graph	Recursive Basis
$\mathcal{G}_1$	$(C \perp\!\!\!\perp B\|A), (D \perp\!\!\!\perp A\|BC), (E \perp\!\!\!\perp ABC\|D)$
$\mathcal{H}_1$	$(D \perp\!\!\!\perp A\|BC), (E \perp\!\!\!\perp ABC\|D)$
$\mathcal{H}_2$	$(E \perp\!\!\!\perp AC\|BD)$
$\mathcal{H}_3$	$(E \perp\!\!\!\perp ABC\|D)$
$\mathcal{H}_4$	$(DE \perp\!\!\!\perp A\|BC)$

Example 7. Consider again Fig. 3. Both $\mathcal{G}_1$ and $\mathcal{G}_2$ are compatible with the summary DAG $\mathcal{H}_1$ shown in Fig. 4(a) (achieved by contracting B and C). However, $\mathcal{G}_3$ is not compatible with $\mathcal{H}_1$, since the edge between D and A is not preserved. □

We aim to find acyclic summary graphs. Thus, we prove a simple lemma characterizing node contractions that preserve acyclicity.

Lemma 3.1. *Let $\mathcal{G}$ be a DAG, and let $V, U \in \mathsf{V}(\mathcal{G})$. Let $\mathcal{H}_{VU}$ denote the summary graph that results from $\mathcal{G}$ by contracting V and U. Then $\mathcal{H}_{VU}$ contains a directed cycle if and only if $\mathcal{G}$ contains a directed path P from V to U (or U to V), where $|P|\geq 2$.*

A *summary causal DAG* is a specific type of summary graph obtained through node contraction operations over a causal DAG $\mathcal{G}$ and ensures acyclicity.

As mentioned, the RB of a causal DAG comprises a set of at most n CIs (where $n{=}|\mathsf{V}(\mathcal{G})|$), signifying that each node is conditionally independent of its preceding nodes[2] given its parents. This succinct set of CIs holds significance, because it enables the derivation of all other CIs represented in the causal DAG.

The RB of a summary causal DAG is defined in a manner akin to the RB of a causal DAG, as denoted by Eq. (2). The only difference is that in a summary causal DAG, a node may represent a subset of nodes of the original DAG.

Example 8. Figure 4 displays four summary graphs for the causal DAG in Figure 3(a). Table 1 shows the RBs of these summary causal DAGs. In $\mathcal{H}_4$ there are only three nodes and therefore the RB includes a single CI statement. □

3.2 The Causal DAG Summarization Problem

As mentioned, we aim to reduce an input causal DAG by contracting its nodes, retaining maximal causal information. We covered the two criteria of our problem before proceeding with formalizing it.

Size Constraint A size constraint is a key motivating constraint for graph summarization work and may be imposed on the number of nodes, storage space, minimum description length, etc. [51]. We focus on a node-based size constraint as limited-size graphs are generally more accessible for inspection [14, 39]. Additionally, setting and adjusting a limit on the number of nodes is shown to be relatively straightforward for analysts [42, 102]. We also observe that other summarization problems share similar hyperparameters, such as many clustering algorithms or k-nearest neighbors [45, 49].

Causal Information Preservation As mentioned, if two variables have a directed path between them in the original DAG, then this relationship should be faithfully preserved in the summary DAG. Indeed, this follows from the definition of a summary DAG (Def. 1).

Given two summary DAGs derived from the same causal DAG $\mathcal{G}$ (i.e., $\mathcal{G}$ is compatible with both summary DAGs), both adhering to the size constraint, we prefer the one that preserves, to a larger degree, the set of CIs represented in $\mathcal{G}$. To this end, we devise a measure to compare summary DAGs based on their RBs. When comparing two summary DAGs $\mathcal{H}_1$ and $\mathcal{H}_2$, we assert that $\mathcal{H}_1$ is *superior* to $\mathcal{H}_2$ if the RB of $\mathcal{H}_2$ is implied by the RB of $\mathcal{H}_1$. Namely, all the CIs encoded by $\mathcal{H}_2$ can also be deduced from $\mathcal{H}_1$. We are searching for a maximal summary causal DAG, namely, that its RB is not implied by any other valid summary DAG.

As mentioned, a summary DAG should not introduce any spurious CIs that the original causal DAG does not imply. However, it may overlook some CIs that are present in the original DAG. We refer to this property as an I-Map. Formally, Let $\Omega \overset{\text{def}}{=} \{X_1, \ldots, X_n\}$ be a set of jointly distributed random variables with distribution $\mathbb{P}$ (i.e., nodes of the original DAG). Formally,

Definition 3 (I-Map)**.** A DAG $\mathcal{G}$ is an *I-Map* for $\mathbb{P}$ if for every disjoint sets $\mathbf{X}, \mathbf{Y}$, and $\mathbf{Z}$ it holds that $(\mathbf{X} \perp\!\!\!\perp_d \mathbf{Y} \mid \mathbf{Z})_{\mathcal{G}}$ only if $(\mathbf{X} \perp\!\!\!\perp_{\mathbb{P}} \mathbf{Y} \mid \mathbf{Z})$.

Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be two DAGs that are I-Maps for $\mathbb{P}$. We say that $\mathcal{G}_2$ is superior to $\mathcal{G}_1$, in notation $\mathcal{G}_2{\succ}\mathcal{G}_1$, if for every $\sigma{\in}\Sigma_{\text{RB}}(\mathcal{G}_1)$, it holds that $\Sigma_{\text{RB}}(\mathcal{G}_2) \implies \sigma$. Note that the relation $\succ$ does not necessarily form a complete order. We say that $\mathcal{G}$ is *maximal for* $\mathbb{P}$ if $\mathcal{G}$ is an I-Map for $\mathbb{P}$, and there does not exist any $\mathcal{G}'{\in}\mathcal{G}(\mathbb{P})$ such that $\mathcal{G}'{\succ}\mathcal{G}$. Our goal is to find a summary DAG that is an I-Map for $\mathbb{P}$ and maximal for $\mathbb{P}$, given a constraint on the number of nodes.

Example 9. Consider the causal DAG $\mathcal{G}_1$ in Fig. 3(a). Fig. 4(a) presents a 4-size summary DAG $\mathcal{H}_1$ for $\mathcal{G}_1$. The RBs of both DAGs are shown in Table 1. Clearly, $\Sigma_{\text{RB}}(\mathcal{H}_1){\subset}\Sigma_{\text{RB}}(\mathcal{G}_1)$, and hence $\mathcal{H}_1$ is an I-Map for $\mathbb{P}$. Fig. 4(b) presents $\mathcal{H}_2$, another 4-size summary DAG for $\mathcal{G}_1$, where $\Sigma_{\text{RB}}(\mathcal{H}_2){=}\{(E \perp\!\!\!\perp AC|BD)\}$. From the semi-graphoid axioms, it holds that $(E \perp\!\!\!\perp ABC|D) \implies (E \perp\!\!\!\perp AC|BD)$. Thus, $\mathcal{H}_1{\succ}\mathcal{H}_2$. Hence, $\mathcal{H}_1$ is a superior summary DAG. Similarly, Figures 4(c) and 4(d) illustrate $\mathcal{H}_3$ and $\mathcal{H}_4$, 3-size summary DAGs for $\mathcal{G}_1$. Their RBs are given in Table 1. The partial order among all summary DAGs is presented in Fig. 4(e). Despite $\mathcal{H}_3$ having only three nodes, it surpasses $\mathcal{H}_2$. However, $\mathcal{H}_3$ and $\mathcal{H}_4$ are incomparable, i.e., neither $\Sigma_{\text{RB}}(\mathcal{H}_3) \implies \Sigma_{\text{RB}}(\mathcal{H}_4)$ nor $\Sigma_{\text{RB}}(\mathcal{H}_4) \implies \Sigma_{\text{RB}}(\mathcal{H}_3)$. □

We define the causal DAG summarization problem as follows:

Problem 1 (Causal DAG Summarization)**.** Given a causal DAG $\mathcal{G}$ defined over a joint distribution $\mathbb{P}$, and a bound k, find a summary causal DAG $\mathcal{H}$ s.t. (i) the number of nodes in $\mathcal{H}$ is $\leq k$; (ii) $\mathcal{G}$ is compatible with $\mathcal{H}$, is an I-Map for $\mathbb{P}$ and is maximal for $\mathbb{P}$.

Example 10. Consider again the causal DAG in Fig. 1. We set k=5. Fig. 2b depicts an optimal summary causal DAG. Namely, the RB of any other summary causal DAG with 5 or fewer nodes is not superior to RB of this 5-node summary causal DAG. □

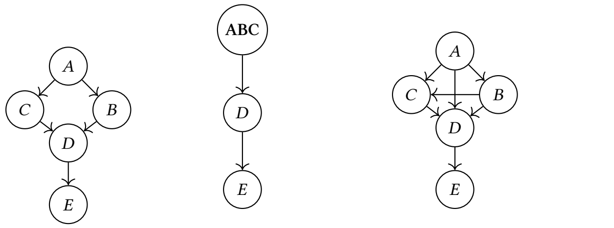
We show that the causal DAG summarization problem is *NP*-hard via a reduction from the k-Max-Cut problem [38]. More details on the proof are given at the end of Section 4.1.

Theorem 3.2. *Causal DAG summarization is an NP-hard problem.*

4 NODE-CONTRACTION AS EDGE ADDITION

Next, we establish the connection between node contractions and the addition of edges to the input causal DAG. This connection will be used to read off, from a given summary causal DAG, all

[2] According to a given full topological order of the nodes



(a) Causal DAG **(b)** Summary DAG **(c)** Canonical causal DAG

Figure 5: A causal DAG, its summary DAG, and the corresponding canonical causal DAG

the CIs it encodes. It also serves as a pivotal factor in guiding our algorithm for selecting promising node pairs to merge. Additionally, in Section 6, we will leverage this connection to demonstrate how causal inference can be directly conducted over summary DAGs.

We note that the canonical causal DAG is not an objective of our problem; rather, it serves as a tool to formally define causal inference over summary DAGs and to guide our algorithm in identifying node contractions that minimize information loss.

4.1 The Canonical Causal DAG

Given a summary causal DAG $\mathcal{H}$, we define its corresponding canonical causal DAG, denoted as $\mathcal{G}_{\mathcal{H}}$. In this causal DAG, cluster nodes are decomposed into distinct nodes connected by edges. We show that the RB of the canonical causal DAG is *equivalent* to that of $\mathcal{H}$. We first define the notion of equivalence for sets of CIs.

Definition 4 (CI Sets Equivalence)**.** Let $\mathbf{S}$ and $\mathbf{T}$ denote two sets of CIs over the variable-set $\{X_1, \ldots, X_n\}$. We say that $\mathbf{S} \Longrightarrow \mathbf{T}$ if $\mathbf{S} \Longrightarrow \sigma$ for every CI $\sigma \in \mathbf{T}$. We say that $\mathbf{S}$ and $\mathbf{T}$ are *equivalent*, in notation $\mathbf{S} \equiv \mathbf{T}$, if $\mathbf{S} \Longrightarrow \mathbf{T}$ and $\mathbf{T} \Longrightarrow \mathbf{S}$.

Next, we formally define the notion of the *canonical causal DAG* for a given summary DAG.

Definition 5 (Canonical Causal DAG)**.** Let $(\mathcal{H}, f)$ be a summary DAG for a causal DAG $\mathcal{G}$. Let $\langle X_1, \ldots, X_n \rangle$ denote a complete topological order over $\mathsf{V}(\mathcal{G})$. We define the canonical causal DAG associated with $(\mathcal{H}, f)$, denoted $\mathcal{G}_{\mathcal{H}}$ as follows: $\mathsf{V}(\mathcal{G}_{\mathcal{H}}) = \mathsf{V}(\mathcal{G})$, and

$$\begin{array}{lr} (X_i, X_j) \in \mathsf{E}(\mathcal{G}_{\mathcal{H}}) \text{ if and only if} & (X_i, X_j) \in \mathsf{E}(\mathcal{G}) \\ \text{or} & (f(X_i), f(X_j)) \in \mathsf{E}(\mathcal{H}) \\ \text{or} & f(X_i) = f(X_j) \text{ and } i < j \end{array}$$

We observe that, by definition, $\mathcal{G}_{\mathcal{H}}$ is compatible with the summary DAG $(\mathcal{H}, f)$.

Example 11. Consider Figures 5(a) and 5(b) that depict an input causal DAG, and its 3-node summary. Fig. 5(c) depicts the corresponding canonical causal DAG. In the topological order A precedes B which in turn precedes C. All nodes within the node **ABC** are connected by edges in the canonical causal DAG, according to the topological order. Since **ABC** is the parent of D in the summary DAG, in the canonical causal DAG all A, B and C are parents of D. Note that Fig. 5(c) contains two more edges than Fig. 5(a), which represents conditional independence relationships which are not captured in the 3-node summary graph Figure 5(b). □

We show that the RB of the canonical causal DAG $\mathcal{G}_{\mathcal{H}}$ is equivalent to that of the summary DAG $\mathcal{H}$ obtained by node contractions to a causal DAG $\mathcal{G}$. In other words, node contractions can be conceptualized as the addition of edges to the input causal DAG.

Theorem 4.1. *Let $\mathcal{H}$ be a summary causal DAG, and $\mathcal{G}_{\mathcal{H}}$ is its corresponding canonical causal DAG. We have: $\Sigma_{RB}(\mathcal{H}) \equiv \Sigma_{RB}(\mathcal{G}_{\mathcal{H}})$.*

Continuing with Example 11, the RB of $\mathcal{G}_{\mathcal{H}_3}$ is $(E \perp\!\!\!\perp ABC|D)$, which is identical to that of $\mathcal{H}_3$ (see Table 1).

Proof Intuition for Theorem 3.2. On our proof we rely on the connection between a summary DAG and it canonical causal DAG. Specifically, Theorem 3.2 establishes that finding a summary DAG $(\mathcal{H}, f)$ whose canonical causal DAG $\mathcal{G}_{\mathcal{H}}$ results in the smallest number of added edges $|\mathsf{E}(\mathcal{G}_{\mathcal{H}})| - |\mathsf{E}(G)|$ is NP-Hard. Specifically, our proof shows that finding a summary DAG $(\mathcal{H}, f)$ where $|\mathsf{V}(\mathcal{H})| = k$ and $|\mathsf{E}(\mathcal{G}_{\mathcal{H}})| - |\mathsf{E}(G)| \le \tau$ for some threshold $\tau > 0$ is an NP-complete problem. In fact, we prove the stronger claim that finding a summary DAG $(\mathcal{H}, f)$ where $|\mathsf{V}(\mathcal{H})| = k$ and

$$\left|\{(X_i, X_j) \in \mathsf{E}(\mathcal{G}_{\mathcal{H}}) \backslash \mathsf{E}(G) : f(X_i) = f(X_j)\}\right| \le \tau \tag{4}$$

is NP-hard. If $|\mathsf{E}(\mathcal{G}_{\mathcal{H}})| - |\mathsf{E}(G)| \le \tau$, then (4) must hold as well. We establish that finding a summary DAG where (4) holds is NP-Hard, and hence finding a summary DAG where $|\mathsf{E}(\mathcal{G}_{\mathcal{H}})| - |\mathsf{E}(G)| \le \tau$ is NP-Hard as well. The full proof is provided in [3].

4.2 s-Separation

We introduce the notion of s-*separation*, an extension of d-separation, tailored to identify CIs encoded by a summary DAG. Intuitively, a summary DAG represents a collection of causal DAGs that are compatible with it, meaning that it could have been obtained from any of those DAGs (similar to *possible worlds* in probabilistic database [23]). Each of these DAGs encodes a different set of CIs. The set of CIs encoded by a summary DAG is defined as the intersection of CIs that holds in all compatible DAGs. In this way, we can ensure we restrict ourselves only to CIs that are certainly present in a particular context and can be reliably used for inference. s-separation extends d-separation, which allows the identification of valid CIs within a *summary* causal DAG, as opposed to regular causal DAGs. We also introduce a sound and complete s-separation algorithm that leverages the standard d-separation algorithm.

The validity of a CI statement, as derived from summary DAG $\mathcal{H}$, is given by the following definition:

Definition 6 (Validity of a CI in a summary DAG)**.** A CI statement is deemed *valid* in a summary causal DAG $\mathcal{H}$ if and only if it is implied by all causal DAGs within $\{\mathcal{G}_i\}_{\mathcal{H}}$.

s-separation captures all certain CIs that hold across all DAGs in $\{\mathcal{G}_i\}_{\mathcal{H}}$. We propose the following criterion for s-separation to encapsulate this notion of validity.

Definition 7 (s-separation)**.** Given a summary DAG $(\mathcal{H}, f)$ and disjoint subsets $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \subseteq \mathsf{V}(\mathcal{H})$, we say that $\mathbf{X}$ and $\mathbf{Y}$ are s-separated in $\mathcal{H}$ by $\mathbf{Z}$, denoted by $(\mathbf{X} \perp\!\!\!\perp_S \mathbf{Y} \mid \mathbf{Z})_{\mathcal{H}}$, iff $f^{-1}(\mathbf{X})$ and $f^{-1}(\mathbf{Y})$ are d-separated by $f^{-1}(\mathbf{Z})$ in every causal DAG within $\{\mathcal{G}_i\}_{\mathcal{H}}$.

We say that $\mathbf{X}$ and $\mathbf{Y}$ are s-connected in $(\mathcal{H}, f)$ by $\mathbf{Z}$, if there exists a causal DAG $\mathcal{G} \in \{\mathcal{G}_i\}_{\mathcal{H}}$, such that $f^{-1}(\mathbf{X})$ and $f^{-1}(\mathbf{Y})$ are d-connected in $\mathcal{G}$ by $f^{-1}(\mathbf{Z})$.

4.2.1 ***s*-separation Algorithm.** Given a summary causal DAG $\mathcal{H}$, we aim to derive the set of CIs it encodes. A naive approach would be to employ d-separation algorithms [73]. However, $\mathcal{H}$ can potentially encompass more CIs than those discerned through d-separation alone, as demonstrated in the following example.

Example 12. Referring back to Fig. 3, $(B \perp\!\!\!\perp_d E \mid D)$, $(C \perp\!\!\!\perp_d E \mid B, D)$, and $(B, C \perp\!\!\!\perp_d E \mid D)$ all hold in $\mathcal{G}_1$ and $\mathcal{G}_2$. Likewise, $(\mathbf{BC} \perp\!\!\!\perp_d E \mid D)$ holds in $\mathcal{H}_1$ (Fig. 4(a)). However, since $\mathcal{H}_1$ does not contain B or C as separate nodes, we cannot establish $(B \perp\!\!\!\perp_d E \mid D)$ or $(C \perp\!\!\!\perp_d E \mid B, D)$ from $\mathcal{H}_1$ using d-separation. □

To address this, a simple solution is to find the set of CIs shared across all DAGs compatible with $\mathcal{H}$. However, this approach is costly. We, therefore, present a simple algorithm for s-separation that leverages the connection between a summary DAG and its canonical causal DAG. This algorithm operates as follows: Given a summary DAG $\mathcal{H}$, establish a topological order for its nodes.[3] Using this order, construct the canonical causal DAG $\mathcal{G}_{\mathcal{H}}$. Next, apply d-separation over $\mathcal{G}_{\mathcal{H}}$ and return the resulting CI set. We demonstrate that this algorithm is sound and complete.

Theorem 4.2 (Soundness and Completeness of s-separation). *In a summary DAG $(\mathcal{H}, f)$, let $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \subseteq \mathbf{V}(\mathcal{H})$ be disjoint sets of nodes. If $\mathbf{X}$ and $\mathbf{Y}$ are d-separated by $\mathbf{Z}$ in $\mathcal{H}$, then in any causal DAG $\mathcal{G} \in \{\mathcal{G}_i\}_{\mathcal{H}}$, $f^{-1}(\mathbf{X})$ and $f^{-1}(\mathbf{Y})$ are d-separated by $f^{-1}(\mathbf{Z})$. That is:*

$$(\mathbf{X} \perp\!\!\!\perp_d \mathbf{Y} | \mathbf{Z})_{\mathcal{H}} \implies (f^{-1}(\mathbf{X}) \perp\!\!\!\perp_d f^{-1}(\mathbf{Y}) | f^{-1}(\mathbf{Z}))_{\mathcal{G}} \implies (\mathbf{X} \perp\!\!\!\perp_s \mathbf{Y} | \mathbf{Z})_{\mathcal{H}}$$

If $\mathbf{X}$ and $\mathbf{Y}$ are d-connected by $\mathbf{Z}$ in $\mathcal{H}$, then there exists a DAG $\mathcal{G} \in \{\mathcal{G}_i\}_{\mathcal{H}}$, s.t $f^{-1}(\mathbf{X})$ and $f^{-1}(\mathbf{Y})$ are d-connected by $f^{-1}(\mathbf{Z})$ in $\mathcal{G}$.

5 THE CAGRES ALGORITHM

As demonstrated in Theorem 3.2, the causal DAG summarization problem is NP-hard and therefore it is not trivial to devise an efficient algorithm with theoretical guarantees. We, therefore, introduce a heuristic algorithm, named CaGreS for the causal DAG summarization problem. Although lacking theoretical guarantees, CaGreS effectively meets the size constraint and produces summary causal DAGs that can be directly used for sound causal inference. A brute force approach explores all summary DAGs with up to k nodes. It finds the optimal summary DAG, but runs in exponential time due to the exponential number of potential graphs. CaGreS addresses this by estimating the merging effect on the canonical causal DAG rather than iterating over all possible summary DAGs.

Overview The CaGreS algorithm follows a previous line of work [34, 102], where a bottom-up greedy approach is used to identify promising node pairs for contraction. Its main contribution lies in *how* it estimates merge costs related to the causal interpretation of the graph: It counts the number of edges to be added in the canonical causal DAG for each node pair (a proxy for the RB's effect, as discussed in Section 4). In each iteration, the algorithm contracts the node pair resulting in the minimal number of additional edges. We also introduce optimizations for runtime efficiency, such as semantic constraint, fast low-cost merges, and caching mechanisms.

The CaGreS algorithm is given in Algorithm 1. Given a bound k and an input causal DAG, this algorithm iteratively seeks the next-best pair of nodes to be merged, until the size constraint is met (lines 4-15). The next-best pair of nodes to merge is the node pair whose contraction has the lowest cost (lines 10-12). The algorithm randomly breaks ties (lines 13-14). The GetCost procedure is shown in Algorithm 2. The cost of merging two (clusters of) nodes $\mathbf{U}$ and $\mathbf{V}$ is equal to the number of edges to be added in the corresponding canonical causal DAG: (1) edges to be added between the nodes within the combined cluster $\mathbf{U} \bigcup \mathbf{V}$ (lines 3-4), (2) new parents for the nodes in $\mathbf{U}$ or $\mathbf{V}$ post-merge (lines 6-11), and (3) new children for the nodes in $\mathbf{U}$ or $\mathbf{V}$ after the merge (lines 13-18).

We next propose three optimizations to improve runtime.

Algorithm 1: The CaGreS Algorithm

```
input  : A causal DAG $\mathcal{G}$ and a number $k$.
output : A summary causal DAG $\mathcal{H}$ with $k$ nodes.
1  $\mathcal{H} \leftarrow \mathcal{G}$
2  /* Merge node-pairs in which their cost is $\leq 1$ */
3  $\mathcal{H} \leftarrow$ LowCostMerges($\mathcal{H}$)
4  while $size(\mathcal{H}.nodes) > k$ do
5      $min\_cost \leftarrow \infty$
6      $(\boldsymbol{X}, \boldsymbol{Y}) \leftarrow Null$
7      for $(\boldsymbol{U}, \boldsymbol{V}) \in \mathcal{H}.nodes$ do
8          if IsValidPair $(\boldsymbol{U}, \boldsymbol{V}, \mathcal{H})$ then
9              cost$_{UV}$ $\leftarrow$ GetCost$(\boldsymbol{U}, \boldsymbol{V}, \mathcal{H})$
10             if $cost_{UV} < min\_cost$ then
11                 min_cost $\leftarrow$ cost$_{UV}$
12                 $(\boldsymbol{X}, \boldsymbol{Y}) \leftarrow (\boldsymbol{U}, \boldsymbol{V})$
13             if $cost_{UV} == min\_cost$ then
14                 Randomly decide if to replace $\boldsymbol{X}$ and $\boldsymbol{Y}$ with $\boldsymbol{U}$ and $\boldsymbol{V}$
15     $\mathcal{H}$.Merge$(\boldsymbol{X}, \boldsymbol{Y})$
16 return $\mathcal{H}$
```

Algorithm 2: The GetCost Procedure

```
input  : A summary causal DAG $\mathcal{H}$ and a pair of nodes $\boldsymbol{U}$ and $\boldsymbol{V}$.
output : The cost of contracting $\boldsymbol{U}$ and $\boldsymbol{V}$.
1  cost $\leftarrow 0$
2  /* New edges among the nodes in the cluster */
3  if $\mathcal{H}$.HasEdge$(\boldsymbol{U}, \boldsymbol{V})$ == $False$ then
4      cost $\leftarrow$ cost $+ size(\boldsymbol{U}) \cdot size(\boldsymbol{V})$
5  /* New parents */
6  parents$_U$ $\leftarrow$ $\mathcal{H}$.GetPredecessors$(\boldsymbol{U})$
7  parents$_V$ $\leftarrow$ $\mathcal{H}$.GetPredecessors$(\boldsymbol{V})$
8  parentsOnlyU $\leftarrow$ parents$_U$ \ parents$_V$
9  cost $\leftarrow$ cost + size(parentsOnlyU)$\cdot$size$(\boldsymbol{V})$
10 parentsOnlyV $\leftarrow$ parents$_V$ \ parents$_U$
11 cost $\leftarrow$ cost + size(parentsOnlyV)$\cdot$size$(\boldsymbol{U})$
12 /* New children */
13 children$_U$ $\leftarrow$ $\mathcal{H}$.GetSuccessors$(\boldsymbol{U})$
14 children$_V$ $\leftarrow$ $\mathcal{H}$.GetSuccessors$(\boldsymbol{V})$
15 childrenOnlyU $\leftarrow$ children$_U$ \ children$_V$
16 cost $\leftarrow$ cost + size(childrenOnlyU) $\cdot$ size$(\boldsymbol{V})$
17 childrenOnlyV $\leftarrow$ children$_V$ \ children$_U$
18 cost $\leftarrow$ cost + size(childrenOnlyV) $\cdot$ size$(\boldsymbol{U})$
19 return cost
```

[3] The order of nodes within a cluster is considered arbitrary, or it may be determined based on the topological order of the input causal DAG if such information is preserved.

Semantic Constraint: We can reduce the search space and ensure that only semantically related variables are merged, thereby supporting semantic coherence in the summary DAG. To achieve this, the user may specify which node pairs are allowed to be merged by providing a semantic similarity matrix and a threshold that indicates the maximum distance between two nodes within a cluster. The user can assess the semantic similarity using previous work on semantic similarity [37, 62] or large language models [5].

Assume a semantic similarity measure $sim(\cdot, \cdot)$ that assigns a value between 0 and 1 to a pair of variables. For a summary DAG $\mathcal{H}$ and a threshold τ, we say that $\mathcal{H}$ satisfies the semantic constraint if, for every cluster $\mathbf{C} \in \mathbb{V}(\mathcal{H})$, $sim(V_i, V_j) \geq \tau$ for every $V_i, V_j \in \mathbf{C}$. This condition is checked in line 8 of the CaGreS algorithm when validating whether a pair of nodes is suitable for contraction.

Caching: We use two caching mechanisms: one for storing invalid node pairs and another for cost scores. We demonstrated in Section 8.5 that these caching mechanisms are effective in reducing runtime.

We initialize the invalid pairs cache during a preprocessing phase. An invalid pair is a node pair with semantic similarity below the threshold or connected by a directed path of length above 2 (Lemma 3.1). During CaGreS's run, invalid pairs are cached, and each iteration checks the validity of node pairs before computing costs.

The cost of a node pair $\mathbf{U}, \mathbf{V}$ remains unchanged after merging another node pair $\mathbf{X}, \mathbf{Y}$ if neither $\mathbf{U}$ nor $\mathbf{V}$ are neighbors of $\mathbf{X}$ or $\mathbf{Y}$. Following the merge of $\mathbf{X}$ and $\mathbf{Y}$, we update the cost cache by removing the cost scores of all node pairs involving one of their neighbors. When calculating the cost for a node pair, we check if the score is in the cache. If not, we compute and add it, ensuring the cache reflects node pair mergers' impact on neighboring pairs.

Low Cost Merges: As a pre-processing step, we contract node pairs with low costs (line 4). This involves merging nodes that share identical children and parents, with a cost of at most 1. Additionally, we merge nodes linked along a non-branching path of nodes, each having at most one parent and one child, incurring a cost of 1. In Section 8.5, we experimentally show that this optimization benefits small or low-density causal DAGs.

Time Complexity A single cost computation with n=$|\mathbb{V}(\mathcal{G})|$ takes $O(n)$ due to the maximum number of neighbors a node can have. The algorithm undergoes $n-k$ iterations, evaluating all node pairs (O(n^2) such pairs) in the current summary DAG with no more than n neighbors. Thus, the overall time complexity is $O((n-k) \cdot n^3)$.

6 DO-CALCULUS IN SUMMARY CAUSAL DAGS

Next, we show that the rules of *do*-calculus are sound and complete in summary causal DAGs. This is vital to ensure that the summary causal DAGs are effective formats that support causal inference by enabling direct causal inference on the summary DAGs. Our proof relies on the equivalence between the RB of a summary DAG and its canonical causal DAG (Theorem 4.1). This result is not surprising because the canonical causal DAG is a supergraph of the input causal DAG. Pearl already observed in [73] that: *"The addition of arcs to a causal diagram can impede, but never assist, the identification of causal effects in nonparametric models. This is because such addition reduces the set of d-separation conditions carried by the diagram; hence, if causal effect derivation fails in the original diagram, it is bound to fail in the augmented diagram".*

Given a causal DAG $\mathcal{G}$, for a set of nodes $\mathbf{X} \subseteq \mathbb{V}(\mathcal{G})$, let $\mathcal{G}_{\overline{\mathbf{X}}}$ denote the graph that results from $\mathcal{G}$ by removing all incoming edges to nodes in $\mathbf{X}$, by $\mathcal{G}_{\underline{\mathbf{X}}}$ the graph that results from $\mathcal{G}$ by removing all outgoing edges from the nodes in $\mathbf{X}$. For a set of nodes $\mathbf{X} \subseteq \mathbb{V}(G) \backslash \mathbf{Z}$, we denote by $\mathcal{G}_{\overline{\mathbf{X}}\underline{\mathbf{Z}}}$ the graph that results from $\mathcal{G}$ by removing all incoming edges into $\mathbf{X}$ and all outgoing edges from $\mathbf{Z}$.

Theorem 6.1 (Soundness of Do-Calculus in summary causal DAGs). *Let $\mathcal{G}$ be a causal DAG encoding an interventional distribution $P(\cdot \mid do(\cdot))$, compatible with the summary causal DAG $(\mathcal{H}, f)$. For any disjoint subsets $\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \mathbf{W} \subseteq \mathbb{V}(\mathcal{H})$, the following rules hold:*

$$R_1 : (\mathrm{Y} \perp\!\!\!\perp \mathrm{Z} | \mathrm{X}, \mathbf{W})_{\mathcal{H}_{\overline{\mathrm{X}}}} \implies P(Y \mid do(\boldsymbol{X}), Z, \boldsymbol{W}) = P(Y \mid do(\boldsymbol{X}), \boldsymbol{W})$$

$$R_2 : (Y \perp\!\!\!\perp Z | \boldsymbol{X}, \boldsymbol{W})_{\mathcal{H}_{\overline{X}\underline{Z}}} \implies P(Y | do(\boldsymbol{X}), do(Z), \boldsymbol{W}) = P(Y | do(\boldsymbol{X}), Z, \boldsymbol{W})$$

$$R_3 : (Y \perp\!\!\!\perp Z | \boldsymbol{X}, \boldsymbol{W})_{\mathcal{H}_{\overline{XZ(\boldsymbol{W})}}} \implies P(Y | do(\boldsymbol{X}), do(Z), \boldsymbol{W}) = P(Y | do(\boldsymbol{X}), \boldsymbol{W})$$

where, $\boldsymbol{U} \overset{def}{=} (U)$ for every $U \in \mathbb{V}(\mathcal{H})$, and $Z(W)$ is the set of nodes in Z that are not ancestors of any node in W.

Theorem 6.2 (Completeness of Do-Calculus in summary causal DAGs). *Let $(\mathcal{H}, f)$ be a summary causal DAG for $\mathcal{G}$, and let $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{W}, \boldsymbol{Z} \subseteq \mathbb{V}(\mathcal{H})$ be disjoint sets of variables. If $\boldsymbol{Y}$ is d-connected to $\boldsymbol{Z}$ in $\mathcal{H}_{\overline{\mathbf{X}}}$ w.r.t. $\boldsymbol{X} \cup \boldsymbol{W}$, then there exists a causal DAG $\mathcal{G}'$ compatible with $\mathcal{H}$, such that $f(Y)$ is d-connected to $f(Z)$ in $\mathcal{G}'_{\overline{f(X)}}$ w.r.t. $f(\boldsymbol{X} \cup \boldsymbol{W})$.*

ATE Computation over Summary DAGs: To conclude this section, we explain how causal effects (ATE, see in Section 2) can be calculated directly over the summary DAG. If the treatment or outcome is part of a cluster node in the summary DAG $\mathcal{H}$, we proceed as follows: For a node pair U, V, we estimate the causal effect $ATE(U, V)$ over the corresponding canonical causal DAG $\mathcal{G}_{\mathcal{H}}$ (this is valid as demonstrated in Section 6). To reduce the adjustment set's size, we arrange for U to precede all nodes within its cluster node in the $\mathcal{G}_{\mathcal{H}}$. An alternative approach that yields an upper and lower bound involves constraining the causal effect across the summary DAG by considering all subsets in U's cluster in $\mathcal{H}$.

7 ROBUSTNESS AGAINST DAG QUALITY

We evaluate the effectiveness of summary DAGs in providing robustness against a flawed input causal DAG. In a case study, we demonstrate that the summary DAG facilitates the handling of errors in the input DAG more effectively than directly examining the causal DAG (which may be overwhelming to the user). This study emphasizes that causal DAG summarization helps address quality issues and increases robustness against misspecifications.

We revisit the Redshift causal DAG (Fig. 1). For each variable pair, we consulted GPT-4 [69] about the edge presence and direction, resulting in 55 detected edges. GPT-4 correctly identified 21 of the 23 original edges, inverted 1, and missed 1. It also generated 33 additional edges not in the original DAG. We will demonstrate how causal DAG summarization can reduce the impact of these errors.

Missing Edges: Starting from the Redshift DAG (Fig. 1), we remove the edge GPT-4 failed to detect (`Result Cache Hit` → `Lock Wait Time`), marked in red in Fig. 6. As evident in Fig. 7a, CaGreS produces the same summary DAG as in Fig. 2b. The information of which node in the cluster {`Results Cache Hit`, `Exec. Time`} has a directed edge to one of the nodes in the cluster {`Plan Time`, `Lock Time`} is lost upon summarization. Any causal

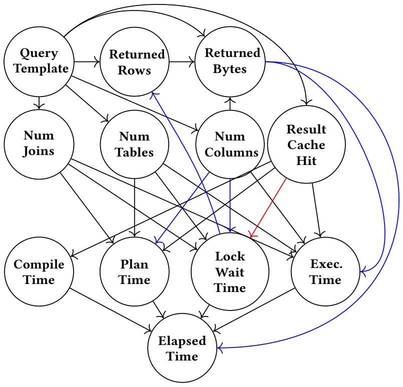
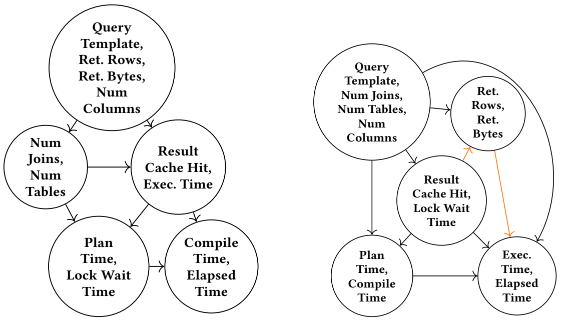


Figure 6: Modifications to the Redshift DAG (Fig. 1).


Query Template, Ret. Rows, Ret. Bytes, Num Columns
Num Joins, Num Tables
Result Cache Hit, Exec. Time
Plan Time, Lock Wait Time
Compile Time, Elapsed Time
Query Template, Num Joins, Num Tables, Num Columns
Ret. Rows, Ret. Bytes
Result Cache Hit, Lock Wait Time
Plan Time, Compile Time
Exec. Time, Elapsed Time

(a) Summary After Deletion **(b) Summary After Additions**

Figure 7: 5-node summary DAGs after DAG modifications.

estimation performed over the summary DAG considers all possible causal DAGs compatible with this summary DAG, including once where the edge is included. Thus, the impact of this error is reduced. **Extraneous Edges**: Starting again from the Redshift DAG, we add 5 random edges from the set of redundant edges produced by GPT-4, marked in blue in Fig. 6. These additional edges reduce the number of CIs entailed by the DAG, which can hurt causal inference accuracy. However, manually pruning extraneous edges would require having the user check each of the (now 28) edges in the DAG for correctness. If we instead summarize the DAG using CaGreS with $k = 5$, the user is faced with the simpler, 9-edge summary DAG shown in Fig. 7b. It is sufficient for the user to detect the 2 suspicious orange edges among these 9 to discover 3 of the 5 extraneous edges. The remaining 2 extraneous edges (from `Num Columns` to `Plan Time` and `Lock Wait Time`) are subsumed grouping `Num Columns` together with other, highly semantically similar, query-related features. As such, graph summarization effectively helps address extraneous edges by facilitating their detection.

8 EXPERIMENTAL EVALUATION

In this section, we empirically demonstrate the following claims: (**C1**) Our summary DAGs support reliable causal inference. (**C2**) Our objective evaluation method effectively determines superior summary causal DAGs. (**C3**) CaGreS outperforms other methods in causal DAG summarization and achieves efficient performance. (C4) Our proposed optimizations help improve the runtime of CaGreS without compromising quality.

Table 2: Datasets

Dataset	# Nodes (Variables)	# Edges	# Tuples
Redshift	12	23	9900
Flights	11	15	1M
Adult	13	48	32.5K
German	21	43	1000
Accidents	41	368	2.8M
Urls	60	310	1.7M

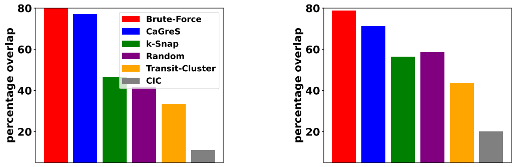
8.1 Experimental Setting

All algorithms are implemented in Python 3.7. Causal effect computation was performed using the DoWhy library [92]. The experiments were executed on a PC with a 4.8GHz CPU, and 16GB memory. Our code and datasets are available at [3].

Datasets. We examine six datasets, as shown in Table 2. Five of the datasets are publicly available, while the remaining one (Redshift) was collected by running a publicly available benchmark on publicly available cloud resources. We use the DAG in Fig. 1 for Redshift and build the input causal DAG using [112] for the remaining datasets. **Redshift**: A dataset collected by running queries from the TPC-DS benchmark [79] on Amazon Redshift Serverless [8]. We execute 100 queries from the query templates benchmark and retrieve the associated entries in the monitoring view [6]. We augment each entry with query-related features (e.g., num joins and tables). **Flights** [2]: a dataset describing domestic flight statistics in the US. We enriched it with attributes describing the weather, population, and properties of the airline carriers. **Adult** [1]: a dataset comprises demographic information of individuals including their education, age, and income. **German** [10]: a dataset contains details of bank account holders, including demographic and financial information. **Accidents** [65]: This dataset provides information on various factors that are pertinent to the severity of car accidents, including weather conditions and the presence of traffic signs. **Urls** [4]: a dataset containing descriptions of malicious and non-malicious URLs. It encompasses properties such as URL length, the number of digits, and the occurrence of sensitive words.

We also created **synthetic data** using the DoWhy package [92], enabling manipulation of node count, edge count, and data size.

Baselines. We examine the following baselines: **Brute-Force**: This algorithm implements an exhaustive search over all possible summary DAGs that satisfy the constraint yielding the optimal solution. **k-Snap** [102]: A general-purpose graph summarization algorithm that employs bottom-up node contractions (akin to CaGreS). The primary distinction lies in the objective function: k-Snap focuses on ensuring homogeneity among nodes within a cluster. We have enhanced k-Snap to address acyclicity. **Transit-Cluster** In [103], the authors proposed Transit Clusters as a specific type of summary causal DAG that maintains identifiability properties under certain conditions. They introduced an algorithm to identify all transit clusters for a graph. For a fair comparison, we consider the transit cluster that meets the constraints and has the maximal RB. CIC [67] The authors of [67] proposed a Clustering Information Criterion (CIC) that represents various complex interactions among variables in a causal DAG. Based on this criterion, they developed a greedy-based approach to learn clustered causal DAGs directly from the data. **Random**: As a sanity check, this algorithm generates a random summary DAG that adheres to the size constraint.



(a) Flights (b) German

Figure 8: Average percentage overlap with ground truth.

Metrics of evaluation. As mentioned, in some cases, summary DAGs are incomparable, meaning that their RBs are not strictly implied by one another. To nevertheless compare their quality, we quantify the number of additional edges in their corresponding canonical causal DAG — edges that are absent in the original DAG. A smaller number of such edges implies a more sparse summary DAG that encodes more CIs.

As a default configuration, we set the size constraint k to $\frac{n}{2}$, where n is the number of nodes in the input causal DAG. The runtime cutoff was set at 1 hour.

8.2 Usability Evaluation (C1)

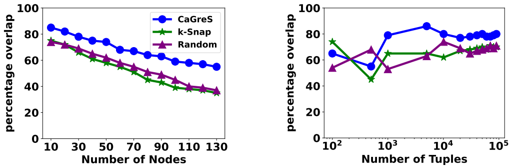
8.2.1 The utility of the summary causal DAGs for causal inference. We assess the utility of the summary causal DAGs for causal inference. To this end, we compare the causal effects estimated within the original DAG with those computed within the summary causal DAGs. Each causal effect estimation yields an interval (of 95% confidence). We compare the intervals derived from the input DAG (the ground truth) with those obtained by the baselines. Given that the adjustment sets in the summary DAGs may differ from those in the original DAG, we anticipate getting different intervals.

Average Percentage Overlap: We report the average percentage of overlap of the causal interval across all node pairs connected by a causal path in the input DAG. A higher percentage overlap indicates greater robustness in causal inference. The results for Flights and German are shown in Fig. 8 (similar trends were observed for the other datasets). CaGreS's average percentage overlap is close to that of Brute-Force, suggesting a high degree of similarity between the two summary DAGs. CaGreS surpasses all other competitors. This underscores the superior suitability of CaGreS for causal inference compared to the baselines.

In what comes next, we use synthetic data, allowing us to manage the number of nodes in the input DAG and database tuples. We omit from presentation the Brute-Force, Transit-Cluster, and CIC baselines as they exceeded our time limit cutoff.

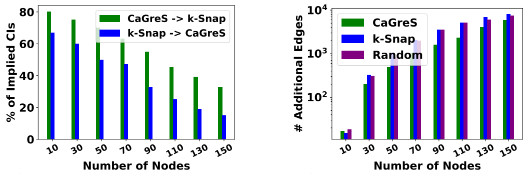
# of attributes: We examine how the number of nodes in the input causal DAG affects the performance. With a larger number of nodes, the task of finding the optimal summary DAG becomes harder. Here, the number of data tuples is fixed at 10K. The results are depicted in Fig. 9(a). For all baselines, with more data attributes, their alignment with the input causal DAG diminishes. Nevertheless, CaGreS consistently outperforms the competing methods.

# of tuples: We analyze the impact of data size on performance, fixing the input causal DAG at 30 nodes. Since causal effects are sensitive to sample size, we expect larger datasets to yield effects on summary DAGs closer to those on the input DAG. As shown in Fig. 9(b), small data sizes produce noisy results, while larger sizes stabilize them. *Again, CaGreS outperforms its competitors.*



(a) (b)

Figure 9: Average percentage overlap vs. data properties.



(a) Higher bars are better (b) Lower bars are better

Figure 10: Quality metrics vs. the number of nodes.

8.3 Quality Evaluation (C2)

When multiple summary DAGs achieve maximal RBs, we use three metrics to identify a superior summary DAG: (1) the percentage of CIs in one summary DAG's RB implied by another; (2) number of additional edges in the canonical causal DAG; and (3) the size of adjustment sets in causal estimation, with smaller sets enhancing accuracy. As we show, these metrics are highly correlated.

We generated random causal DAGs with various number of nodes (five DAGs for each node count), while keeping all other parameters fixed. We omit from presentation the Brute-Force, Transit-Cluster, and CIC baselines as they exceeded our time cutoff. The results are depicted in Fig. 10. Fig. 10(a) depicts the percentage of CIs in the RB of k-Snap that are implied by that of CaGreS and vice versa. Similar trends were observed for Random. A higher percentage of k-Snap's CIs are implied by CaGreS compared to the percentage of CaGreS's CIs that are implied by k-Snap. Hence, while no RB entirely implies the other, we can still conclude that the summary DAG of CaGreS is superior to that of k-Snap. Fig. 10(b) depicts the number of additional edges in the canonical causal DAG. CaGreS consistently yields summary DAGs with fewer edges. We also considered the average size of the adjustment sets in the computation of causal estimations (omitted from the presentation). We report that CaGreS outperforms the competitors, consistently yielding smaller adjustment sets. *Since these three metrics are closely interrelated, we deduce that it is appropriate to use the count of additional edges for comparing the quality of summary DAGs.*

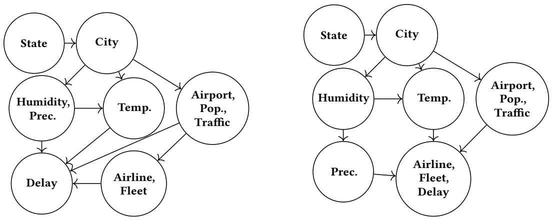
8.4 Effectiveness Evaluation (C3)

We assess CaGreS based on quality and runtime performance.

Case Study: Flights We present the pairwise percentage of the CIs in the RB implied by all baseline pairs. The results are shown in Table 3. The summary DAGs obtained by CaGreS and k-Snap are given in Fig. 11 (The optimal summary DAG by Brute-Force is omitted from presentation). Brute-Force yields the most effective summary DAG, as it implies the highest percentage of CIs of any other baseline. While 60% of the CIs of k-Snap are implied by the RB of CaGreS, only 16% of the CIs of CaGreS are implied by the RB of k-Snap. This superiority of CaGreS over k-Snap is further

Table 3: Pair-wise percentage of the RB's CIs implied.

	Brute-Force	CaGreS	k-Snap	Random	TC	CIC
Brute-Force	-	83.3%	50%	50%	16.6%	16.6%
CaGreS	50%	-	60%	16.6%	0%	16%
k-Snap	0%	16.6%	-	50%	16.6%	0%
Random	16.6%	0%	50%	-	0%	16.6%
TC	0%	0%	16.6%	16.6%	-	50%
CIC	0%	0%	0%	16.6%	0%	-



(a) CaGreS (b) k-Snap

Figure 11: Summary causal DAGs for the Flights dataset.

supported by a lower number of additional edges (7 for CaGreS, 13 for Brute-Force, and 14 for k-Snap). Intuitively, this stems from k-Snap's decision to form two 3-size clusters, connected by an edge. In the resulting canonical causal DAG, every pair of nodes within and between the clusters is connected by an edge.

Next, for each dataset, we report the runtime and the number of additional edges. The results are depicted in Fig. 12. Only CaGreS, k-Snap, and Random can handle causal DAGs with more than 20 nodes within a responsive runtime. While Random and k-Snap exhibit runtimes comparable to that of CaGreS, CaGreS consistently produces summary DAGs with fewer additional edges. As expected, Brute-Force outperforms CaGreS in terms of quality but is impractical for interactive interaction. CIC exhibits relatively low performance, primarily due to a causal discovery component. Transit-Cluster cannot handle large causal DAGs, as the algorithm materializes all transit clusters to select the maximal one.

We next analyze the influence of different parameters on performance. In these experiments, our focus shifts to synthetic data, which enables us to manipulate data-related factors.

Input DAG size We vary the number of nodes in the input DAG by generating a series of random DAGs varying the number of nodes (5 DAGs per node count) and keeping all other parameters constant. The results are shown in Fig. 13. As expected, k-Snap and CaGreS exhibit a polynomial increase in runtime (Fig. 13(a)). The improvement relative to k-Snap is attributed to our caching mechanisms. CaGreS consistently generates summary DAGs with fewer additional edges (Fig. 13(b)), indicating better quality.

Summary size We vary the size constraint k. Here, the node count is set to 50, and the graph density is held constant at 0.3. The results are depicted in Fig.14. The runtime of both CaGreS and k-Snap demonstrate a linear increase with k (Fig. 14(a)). This is because larger k values necessitate more merges. As expected, CaGreS manages to generate summary DAGs corresponding to canonical causal DAGs with fewer edges (Fig. 14(b)).

Graph density We investigate the influence of graph density on performance. We observe a nearly linear increase in runtime for both CaGreS and k-Snap as graph density rises (Fig. 15(a)). This is because both algorithms examine neighboring nodes of each node pair, and higher density increases the number of neighbors. As density increases, both algorithms add more edges. However, at high densities (above 0.7), fewer edges remain to be added, so the number of additional edges decreases (Fig. 15(b)).

Data size We report that the data size, i.e., number of tuples, has no effect on the performance of CaGreS and k-Snap. This is because both algorithms only examine the input causal DAGs.

8.5 Optimizations (C4)

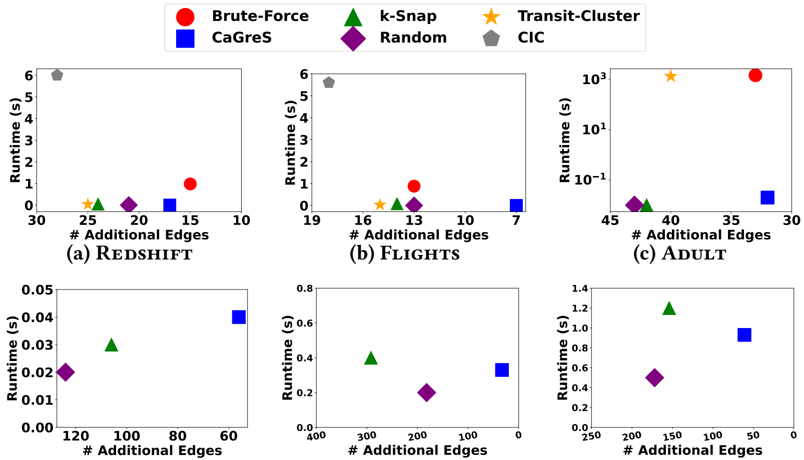
We assess the effect of our optimizations on the performance of CaGreS. To this end, we examine three variants of CaGreS: (I) No Cache, a version of CaGreS without caching, (ii) No Preprocessing, a version without the low-cost merge optimization, and (iii) No Optimizations: a variant without either optimization.

Here, we again used synthetic data to create input causal DAGs, varying the number of nodes and graph density as was done in Section 8.4. The results are displayed in Figure 16. For quality, we observed that across all baselines, the number of additional edges remained the same despite changes in the number of nodes, and thus the relevant plot is omitted from presentation. This suggests that *our optimizations improve runtime without compromising the quality.* The results indicate that the *caching mechanisms significantly improve runtime (making it run nearly three times faster), while the impact of preprocessing is modest.* In fact, for large high-denisty input causal DAGs, preprocessing may slightly slow down the algorithm, though it provides a runtime improvement for smaller sparser DAGs. Concluding, we recommend using this optimization specifically for relatively small, sparse DAGs.

9 RELATED WORK

Summary Causal DAGs. The abstraction of causal models has been studied in literature [90]. Previous work [12, 71] investigated the problem of determining under what assumptions DAGs over sets of variables can represent the same CIs. The authors of [17, 18, 85] explored the problem of determining the causes of a target behavior (macro-variable) from micro-variables (e.g., image pixels). Other works consider chain or ancestral causal graphs [46, 113]. [80] presented a method to compress causal graphs by removing nodes to eliminate redundancy. In contrast, our work addresses causal DAG summarization, where some causal information is lost but the summary DAG still supports reliable causal inference. The authors of [9] expanded the *do-calculus* framework [73] to clustered causal graphs, a related but distinct concept. Our contribution lies in presenting a more streamlined proof of this principle, relying on the connection between node contraction and edge addition.

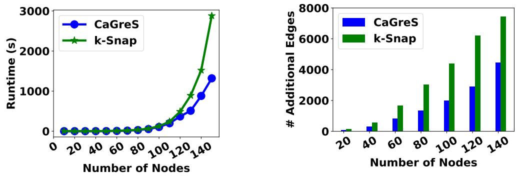
General-Purpose Summary Graphs Graph summarization aims to condense an input graph into a more concise representation. Graph summarization has been extensively studied within the data management community [14, 25, 50, 54, 66], as summarized graphs not only reduce the graph's size but also enable efficient query answering [25, 52, 81, 95], enables enhanced data visualization and pattern discovery [22, 24, 41, 44, 48, 93], and supports extraction of influence dynamics [57]. Various graph summarization techniques have been explored, including grouping nodes based on similarity measures [47, 50, 61, 66, 81, 95, 96, 98, 102, 109, 115], reducing the number bits required to represent graphs [15, 54, 66, 82, 91], and



(a) Redshift (b) Flights (c) Adult

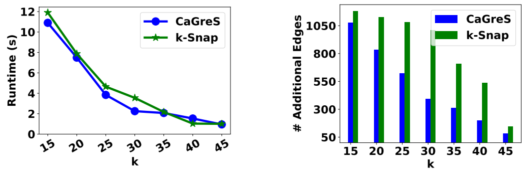
(d) German (e) Accidents (f) Urls

Figure 12: Number of additional edges vs. runtime. The optimal solution should be located in the lower right region.



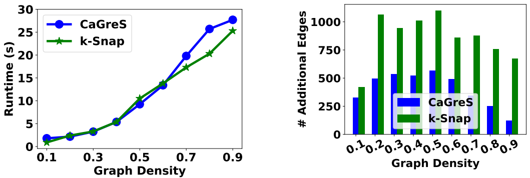
(a) Runtime (b) Quality

Figure 13: Number of nodes vs. running times and quality.



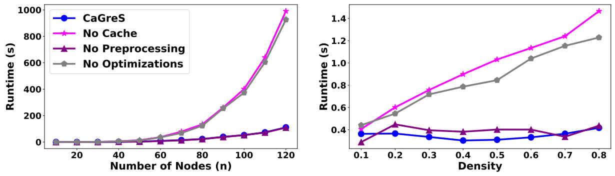
(a) Runtime (b) Quality

Figure 14: Summary size k vs. running times and quality.



(a) Runtime (b) Quality

Figure 15: Graph density vs. running times and quality.



(a) Varying # nodes (density = 0.4) (b) Varying density (n = 30)

Figure 16: Optimizations.

removing unimportant nodes and edges [52, 99]. We argue that existing techniques are ill-suited for the causal DAG summarization problem. Graph summarization objectives differ across applications, often prioritizing minimizing the reconstruction error [47, 109], facilitating accurate query answering [52, 95], selecting contractions that preserve shortest path distances to facilitate routing queries [34], or enhancing visualizations [41, 44]. Consequently, existing methods inadequately cater to the objective of preserving causal information, often yielding graphs unsuitable for causal inference, as shown in Section 1. Our algorithm follows a previous line of work [34, 102], where a bottom-up greedy approach is used to identify promising node pairs for contraction. Our main contribution lies in how it estimates merge costs related to the causal interpretation of the graph and the objective of preserving causal information.

Causal Discovery. Causal discovery is a well-studied problem [35, 112, 114], whose goal is to infer causal relationships among variables. While background knowledge is crucial [74], causal DAGs can be inferred from data under certain assumptions [20, 35]. Existing methods include constraint-based [100] and score-based algorithms [20, 94, 107, 118]. Pashami et al. [72] proposed a clustering-based method, using a cluster-based conflict resolution mechanism to determine the causal relationship among variables. Recent works [16, 105] have explored the use of LLMs for causal discovery. Our aim is to summarize causal DAGs representing relationships in high-dimensional data. Consequently, our work serves as a complementary endeavor to existing research in causal discovery.

10 LIMITATIONS & CONCLUSIONS

A mixed graph, incorporating both directed and undirected edges, is a typical output of causal discovery algorithms [19, 76, 100]. For simplicity in exposition, we concentrated on regular causal DAGs throughout this paper. Nevertheless, our results and algorithms apply to mixed graphs as well.

The size constraint can greatly impact the generated summary DAG, and users may need to adjust it to obtain a desirable summary. Future research will explore methods to recommend an optimal value for this parameter. For example, a heuristic stopping condition could be added to the algorithm, signaling it to halt if the next merge would result in a significant loss of information.

This paper opens up promising future research directions. This includes the development of compact representations of node sets tailored specifically for causal inference, addressing additional size constraints, and refining algorithms with theoretical guarantees.

REFERENCES

[1] 2016. Adult Census Income Dataset. https://www.kaggle.com/datasets/uciml/adult-census-income. Accessed: 2024-04-04.

[2] 2020. Kaggle Datasets: Flights Delay. https://www.kaggle.com/usdot/flight-delays.

[3] 2024. Code Repository and Technical Report. https://github.com/brityoungmann/CausalDAGSummarization. Accessed: 2024-07-30.

[4] 2024. Kaggle Datasets: malicious url detection. https://www.kaggle.com/datasets/pilarpieiro/tabular-dataset-ready-for-malicious-url-detection.

[5] 2024. OpenAI ChatGPT (3.5) [Large language model]. https://openai.com/blog/chatgpt. Accessed: 2024-04-04.

[6] 2024. SYS_QUERY_HISTORY - Amazon Redshift. https://docs.aws.amazon.com/redshift/latest/dg/SYS_QUERY_HISTORY.html.

[7] Abdullah Alomar, Pouya Hamadanian, Arash Nasr-Esfahany, Anish Agarwal, Mohammad Alizadeh, and Devavrat Shah. 2023. {CausalSim}: A Causal Framework for Unbiased {Trace-Driven} Simulation. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*. 1115–1147.

[8] Amazon Web Services. 2024. Amazon Redshift Serverless. https://aws.amazon.com/redshift/redshift-serverless/.

[9] Tara V Anand, Adele H Ribeiro, Jin Tian, and Elias Bareinboim. 2023. Causal Effect Identification in Cluster DAGs. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[10] Arthur Asuncion and David Newman. 2007. UCI machine learning repository.

[11] Sander Beckers. 2022. Causal explanations and XAI. In *Conference on causal learning and reasoning*. PMLR, 90–109.

[12] Sander Beckers and Joseph Y Halpern. 2019. Abstracting causal models. In *Proceedings of the aaai conference on artificial intelligence*, Vol. 33. 2678–2685.

[13] Leopoldo Bertossi and Babak Salimi. 2017. From causes for database queries to repairs and model-based diagnosis and back. *Theory of Computing Systems* 61 (2017), 191–232.

[14] Sourav S Bhowmick and Byron Choi. 2022. Data-driven visual query interfaces for graphs: Past, present, and (near) future. In *Proceedings of the 2022 International Conference on Management of Data*. 2441–2447.

[15] Paolo Boldi and Sebastiano Vigna. 2004. The webgraph framework I: compression techniques. In *Proceedings of the 13th international conference on World Wide Web*. 595–602.

[16] Alessandro Castelnovo, Riccardo Crupi, Fabio Mercorio, Mario Mezzanzanica, Daniele Potertì, and Daniele Regoli. 2024. Marrying LLMs with Domain Expert Validation for Causal Graph Generation. (2024).

[17] Krzysztof Chalupka, Frederick Eberhardt, and Pietro Perona. 2016. Multi-level cause-effect systems. In *Artificial intelligence and statistics*. PMLR, 361–369.

[18] Krzysztof Chalupka, Pietro Perona, and Frederick Eberhardt. 2015. Visual causal feature learning. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 181–190.

[19] Wenyu Chen, Mathias Drton, and Ali Shojaie. 2023. Causal Structural Learning via Local Graphs. *SIAM Journal on Mathematics of Data Science* 5, 2 (2023), 280–305.

[20] D.M Chickering. 2002. Optimal structure identification with greedy search. *JMLR* 3, Nov (2002), 507–554.

[21] Anthony C Constantinou, Yang Liu, Kiattikun Chobtham, Zhigao Guo, and Neville K Kitson. 2021. Large-scale empirical validation of Bayesian Network structure learning algorithms with noisy data. *International Journal of Approximate Reasoning* 131 (2021), 151–188.

[22] Diane J Cook and Lawrence B Holder. 1993. Substructure discovery using minimum description length and background knowledge. *Journal of Artificial Intelligence Research* 1 (1993), 231–255.

[23] Nilesh Dalvi and Dan Suciu. 2007. Efficient query evaluation on probabilistic databases. *The VLDB Journal* 16 (2007), 523–544.

[24] Cody Dunne and Ben Shneiderman. 2013. Motif simplification: improving network visualization readability with fan, connector, and clique glyphs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 3247–3256.

[25] Wenfei Fan, Jianzhong Li, Xin Wang, and Yinghui Wu. 2012. Query preserving graph compression. In *Proceedings of the 2012 ACM SIGMOD international conference on management of data*. 157–168.

[26] Sainyam Galhotra, Amir Gilad, Sudeepa Roy, and Babak Salimi. 2022. Hyper: Hypothetical reasoning with what-if and how-to queries using a probabilistic causal approach. In *Proceedings of the 2022 International Conference on Management of Data*. 1598–1611.

[27] Sainyam Galhotra, Yue Gong, and Raul Castro Fernandez. 2023. Metam: Goal-oriented data discovery. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. IEEE, 2780–2793.

[28] Sainyam Galhotra, Romila Pradhan, and Babak Salimi. 2021. Explaining black-box algorithms using probabilistic contrastive counterfactuals. In *Proceedings of the 2021 International Conference on Management of Data*. 577–590.

[29] Yu Gan, Mingyu Liang, Sundar Dev, David Lo, and Christina Delimitrou. 2021. Sage: Practical and Scalable ML-Driven Performance Debugging in Microservices. In *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*. 135–151.

[30] Markus Gangl. 2010. Causal inference in sociological research. *Annual review of sociology* 36 (2010), 21–47.

[31] M. R. Garey, David S. Johnson, and Larry J. Stockmeyer. 1976. Some Simplified NP-Complete Graph Problems. *Theor. Comput. Sci.* 1, 3 (1976), 237–267.

[32] Dan Geiger and Judea Pearl. 1988. On the logic of causal models. In *UAI '88: Proceedings of the Fourth Annual Conference on Uncertainty in Artificial Intelligence, Minneapolis, MN, USA, July 10-12, 1988*. 3–14.

[33] Dan Geiger, Thomas Verma, and Judea Pearl. 1990. Identifying independence in bayesian networks. *Networks* 20, 5 (1990), 507–534.

[34] Robert Geisberger, Peter Sanders, Dominik Schultes, and Christian Vetter. 2012. Exact Routing in Large Road Networks Using Contraction Hierarchies. *Transportation Science* 46, 3 (2012), 388–404.

[35] Clark Glymour, Kun Zhang, and Peter Spirtes. 2019. Review of causal discovery methods based on graphical models. *Frontiers in genetics* 10 (2019), 524.

[36] Helga Gudmundsdottir, Babak Salimi, Magdalena Balazinska, Dan RK Ports, and Dan Suciu. 2017. A demonstration of interactive analysis of performance measurements with viska. In *Proceedings of the 2017 ACM International Conference on Management of Data*. 1707–1710.

[37] Sébastien Harispe, Sylvie Ranwez, Stefan Janaqi, and Jacky Montmain. 2015. Semantic Similarity from Natural Language and Ontology Analysis. *ArXiv* abs/1704.05295 (2015).

[38] Juris Hartmanis. 1982. Computers and intractability: a guide to the theory of np-completeness (michael r. garey and david s. johnson). *Siam Review* 24, 1 (1982), 90.

[39] Weidong Huang, Peter Eades, and Seok-Hee Hong. 2009. Measuring effectiveness of graph visualizations: A cognitive load perspective. *Information Visualization* 8, 3 (2009), 139–152.

[40] Johannes Huegle, Christopher Hagedorn, Lukas Boehme, Mats Poerschke, Jonas Umland, and Rainer Schlosser. 2021. MANM-CS: Data generation for benchmarking causal structure learning from mixed discrete-continuous and nonlinear data. *WHY-21 at NeurIPS* 2021 (2021).

[41] Lisa Jin and Danai Koutra. 2017. Ecoviz: Comparative vizualization of time-evolving network summaries. In *ACM SIGKDD 2017 Workshop on Interactive Data Exploration and Analytics*.

[42] Arijit Khan, Sourav S. Bhowmick, and Francesco Bonchi. 2017. Summarizing Static and Dynamic Big Graphs. *Proc. VLDB Endow.* 10, 12 (2017), 1981–1984.

[43] Samantha Kleinberg and George Hripcsak. 2011. A review of causal inference for biomedical informatics. *Journal of biomedical informatics* 44, 6 (2011), 1102–1112.

[44] Danai Koutra, U Kang, Jilles Vreeken, and Christos Faloutsos. 2014. Vog: Summarizing and understanding large graphs. In *Proceedings of the 2014 SIAM international conference on data mining*. SIAM, 91–99.

[45] Oliver Kramer and Oliver Kramer. 2013. K-nearest neighbors. *Dimensionality reduction with unsupervised nearest neighbors* (2013), 13–23.

[46] Steffen L Lauritzen and Thomas S Richardson. 2002. Chain graph models and their causal interpretations. *Journal of the Royal Statistical Society Series B: Statistical Methodology* 64, 3 (2002), 321–348.

[47] Kyuhan Lee, Hyeonsoo Jo, Jihoon Ko, Sungsu Lim, and Kijung Shin. 2020. Ssumm: Sparse summarization of massive graphs. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 144–154.

[48] Chenhui Li, George Baciu, and Yunzhe Wang. 2015. Modulgraph: modularity-based visualization of massive graphs. In *SIGGRAPH Asia 2015 Visualization in High Performance Computing*. 1–4.

[49] Aristidis Likas, Nikos Vlassis, and Jakob J Verbeek. 2003. The global k-means clustering algorithm. *Pattern recognition* 36, 2 (2003), 451–461.

[50] Xingjie Liu, Yuanyuan Tian, Qi He, Wang-Chien Lee, and John McPherson. 2014. Distributed graph summarization. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. 799–808.

[51] Yike Liu, Tara Safavi, Abhilash Dighe, and Danai Koutra. 2018. Graph summarization methods and applications: A survey. *ACM computing surveys (CSUR)* 51, 3 (2018), 1–34.

[52] Antonio Maccioni and Daniel J Abadi. 2016. Scalable pattern matching over compressed graphs via dedensification. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1755–1764.

[53] Sara Magliacane, Thijs Van Ommen, Tom Claassen, Stephan Bongers, Philip Versteeg, and Joris M Mooij. 2018. Domain adaptation by using causal inference to predict invariant conditional distributions. *Advances in neural information processing systems* 31 (2018).

[54] Sebastian Maneth and Fabian Peternek. 2016. Compressing graphs by grammars. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE, 109–120.

[55] Markos Markakis, An Bo Chen, Brit Youngmann, Trinity Gao, Ziyu Zhang, Rana Shahout, Peter Baile Chen, Chunwei Liu, Ibrahim Sabek, and Michael Cafarella. 2024. Sawmill: From Logs to Causal Diagnosis of Large Systems. In *SIGMOD*.

444–447.
[56] Markos Markakis, Ziyu Zhang, Rana Shahout, Trinity Gao, Chunwei Liu, Ibrahim Sabek, and Michael Cafarella. 2024. Press ECCS to Doubt (Your Causal Graph). In *Proceedings of the Conference on Governance, Understanding and Integration of Data for Effective and Responsible AI* (Santiago, AA, Chile) *(GUIDE-AI '24)*. Association for Computing Machinery, New York, NY, USA, 6–15. https://doi.org/10.1145/3665601.3669842
[57] Yasir Mehmood, Nicola Barbieri, Francesco Bonchi, and Antti Ukkonen. 2013. Csi: Community-level social influence analysis. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part II 13*. Springer, 48–63.
[58] Alexandra Meliou, Wolfgang Gatterbauer, Joseph Y Halpern, Christoph Koch, Katherine F Moore, and Dan Suciu. 2010. Causality in databases. *IEEE Data Engineering Bulletin* 33, 3 (2010), 59–67.
[59] Alexandra Meliou, Wolfgang Gatterbauer, Katherine F Moore, and Dan Suciu. 2009. Why so? or why no? functional causality for explaining query answers. *arXiv preprint arXiv:0912.5340* (2009).
[60] Alexandra Meliou, Sudeepa Roy, and Dan Suciu. 2014. Causality and explanations in databases. *Proceedings of the VLDB Endowment* 7, 13 (2014), 1715–1716.
[61] Arpit Merchant, Michael Mathioudakis, and Yanhao Wang. 2023. Graph Summarization via Node Grouping: A Spectral Algorithm. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 742–750.
[62] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[63] Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence* 267 (2019), 1–38.
[64] Karthika Mohan, Judea Pearl, and Jin Tian. 2013. Graphical models for inference with missing data. *Advances in neural information processing systems* 26 (2013).
[65] Sobhan Moosavi, Mohammad Hossein Samavatian, Srinivasan Parthasarathy, Radu Teodorescu, and Rajiv Ramnath. 2019. Accident risk prediction based on heterogeneous sparse data: New dataset and insights. In *Proceedings of the 27th ACM SIGSPATIAL international conference on advances in geographic information systems*. 33–42.
[66] Saket Navlakha, Rajeev Rastogi, and Nisheeth Shrivastava. 2008. Graph summarization with bounded error. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. 419–432.
[67] Xueyan Niu, Xiaoyun Li, and Ping Li. 2022. Learning Cluster Causal Diagrams: An Information-Theoretic Approach. (2022).
[68] Chris J Oates, Jessica Kasza, Julie A Simpson, and Andrew B Forbes. 2017. Repair of partly misspecified causal diagrams. *Epidemiology* 28, 4 (2017), 548–552.
[69] OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL]
[70] RT O'donnell, Ann E Nicholson, B Han, Kevin B Korb, MJ Alam, and LR Hope. 2006. Incorporating expert elicited structural information in the CaMML causal discovery program. In *Proceedings of the 19th Australian Joint Conference on Artificial Intelligence: Advances in Artificial Intelligence*. 1–16.
[71] Pekka Parviainen and Samuel Kaski. 2016. Bayesian networks for variable groups. In *Conference on Probabilistic Graphical Models*. PMLR, 380–391.
[72] Sepideh Pashami, Anders Holst, Juhee Bae, and Sławomir Nowaczyk. 2018. Causal discovery using clusters from observational data. In *FAIM'18 Workshop on CausalML, Stockholm, Sweden, July 15, 2018*.
[73] Judea Pearl. 2000. *Causality : models, reasoning, and inference*. Cambridge University Press.
[74] J. Pearl and D. Mackenzie. 2018. *The book of why: the new science of cause and effect*. Basic books.
[75] Sriram Pemmaraju, Steven Skiena, et al. 2003. *Computational discrete mathematics: Combinatorics and graph theory with mathematica®*. Cambridge university press.
[76] Jose M Peña. 2016. Learning acyclic directed mixed graphs from observations and interventions. In *Conference on Probabilistic Graphical Models*. PMLR, 392–402.
[77] Emilija Perkovic. 2020. Identifying causal effects in maximally oriented partially directed acyclic graphs. In *Conference on Uncertainty in Artificial Intelligence*. PMLR, 530–539.
[78] Alireza Pirhadi, Mohammad Hossein Moslemi, Alexander Cloninger, Mostafa Milani, and Babak Salimi. 2024. Otclean: Data cleaning for conditional independence violations using optimal transport. *Proceedings of the ACM on Management of Data* 2, 3 (2024), 1–26.
[79] Meikel Poess, Bryan Smith, Lubor Kollar, and Paul Larson. 2002. TPC-DS, Taking Decision Support Benchmarking to the Next Level. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*. 582–587.
[80] Cristina Puente, José Angel Olivas, E Garrido, and R Seisdedos. 2013. Compressing the representation of a causal graph. In *2013 Joint IFSA World Congress and NAFIPS Annual Meeting (IFSA/NAFIPS)*. IEEE, 122–127.
[81] Sriram Raghavan and Hector Garcia-Molina. 2003. Representing web graphs. In *Proceedings 19th International Conference on Data Engineering (Cat. No. 03CH37405)*. IEEE, 405–416.
[82] Ryan A Rossi and Rong Zhou. 2018. Graphzip: a clique-based sparse graph compression method. *Journal of Big Data* 5, 1 (2018), 1–14.
[83] Sudeepa Roy. 2022. Toward interpretable and actionable data analysis with explanations and causality. *Proc. VLDB Endow.* 15, 12 (2022), 3812–3820.
[84] Sudeepa Roy and Dan Suciu. 2014. A formal approach to finding explanations for database queries. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. 1579–1590.
[85] Paul K Rubenstein, Sebastian Weichwald, Stephan Bongers, Joris M Mooij, Dominik Janzing, Moritz Grosse-Wentrup, and Bernhard Schölkopf. 2017. Causal consistency of structural equation models. *arXiv preprint arXiv:1707.00819* (2017).
[86] Donald B Rubin. 2005. Causal inference using potential outcomes: Design, modeling, decisions. *J. Amer. Statist. Assoc.* 100, 469 (2005), 322–331.
[87] Babak Salimi, Johannes Gehrke, and Dan Suciu. 2018. Bias in olap queries: Detection, explanation, and removal. In *SIGMOD*. 1021–1035.
[88] Babak Salimi, Harsh Parikh, Moe Kayali, Lise Getoor, Sudeepa Roy, and Dan Suciu. 2020. Causal relational learning. In *Proceedings of the 2020 ACM SIGMOD international conference on management of data*. 241–256.
[89] Babak Salimi, Luke Rodriguez, Bill Howe, and Dan Suciu. 2019. Interventional fairness: Causal database repair for algorithmic fairness. In *Proceedings of the 2019 International Conference on Management of Data*. 793–810.
[90] Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. 2021. Toward causal representation learning. *Proc. IEEE* 109, 5 (2021), 612–634.
[91] Neil Shah, Danai Koutra, Lisa Jin, Tianmin Zou, Brian Gallagher, and Christos Faloutsos. 2017. On Summarizing Large-Scale Dynamic Graphs. *IEEE Data Eng. Bull.* 40, 3 (2017), 75–88.
[92] Amit Sharma and Emre Kiciman. 2020. DoWhy: An End-to-End Library for Causal Inference. *arXiv preprint arXiv:2011.04216* (2020).
[93] Zeqian Shen, Kwan-Liu Ma, and Tina Eliassi-Rad. 2006. Visual analysis of large heterogeneous social networks by semantic and structural abstraction. *IEEE transactions on visualization and computer graphics* 12, 6 (2006), 1427–1439.
[94] Shohei Shimizu, Patrik O Hoyer, Aapo Hyvärinen, Antti Kerminen, and Michael Jordan. 2006. A linear non-Gaussian acyclic model for causal discovery. *Journal of Machine Learning Research* 7, 10 (2006).
[95] Kijung Shin, Amol Ghoting, Myunghwan Kim, and Hema Raghavan. 2019. Sweg: Lossless and lossy summarization of web-scale graphs. In *The World Wide Web Conference*. 1679–1690.
[96] Maryam Shoaran, Alex Thomo, and Jens H Weber-Jahnke. 2013. Zero-knowledge private graph summarization. In *2013 IEEE International Conference on Big Data*. IEEE, 597–605.
[97] Karamjit Singh, Garima Gupta, Vartika Tewari, and Gautam Shroff. 2018. Comparative benchmarking of causal discovery algorithms. In *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*. Association for Computing Machinery, 46–56.
[98] Qi Song, Yinghui Wu, Peng Lin, Luna Xin Dong, and Hui Sun. 2018. Mining summaries for knowledge graph search. *IEEE Transactions on Knowledge and Data Engineering* 30, 10 (2018), 1887–1900.
[99] Daniel A Spielman and Nikhil Srivastava. 2008. Graph sparsification by effective resistances. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*. 563–568.
[100] P. Spirtes et al. 2000. *Causation, prediction, and search*. MIT press.
[101] Xinwei Sun, Botong Wu, Xiangyu Zheng, Chang Liu, Wei Chen, Tao Qin, and Tie-Yan Liu. 2021. Recovering latent causal factor for generalization to distributional shifts. *Advances in Neural Information Processing Systems* 34 (2021), 16846–16859.
[102] Yuanyuan Tian, Richard A Hankins, and Jignesh M Patel. 2008. Efficient aggregation for graph summarization. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. 567–580.
[103] Santtu Tikka, Jouni Helske, and Juha Karvanen. 2021. Clustering and Structural Robustness in Causal Diagrams. *arXiv preprint arXiv:2111.04513* (2021).
[104] Hal R Varian. 2016. Causal inference in economics and marketing. *Proceedings of the National Academy of Sciences* 113, 27 (2016), 7310–7315.
[105] Aniket Vashishtha, Abbavaram Gowtham Reddy, Abhinav Kumar, Saketh Bachu, Vineeth N Balasubramanian, and Amit Sharma. 2023. Causal inference using llm-guided discovery. *arXiv preprint arXiv:2310.15117* (2023).
[106] Thomas Verma and Judea Pearl. 1988. Causal networks: semantics and expressiveness. In *UAI '88: Proceedings of the Fourth Annual Conference on Uncertainty in Artificial Intelligence, Minneapolis, MN, USA, July 10-12, 1988*, Ross D. Shachter, Tod S. Levitt, Laveen N. Kanal, and John F. Lemmer (Eds.). North-Holland, 69–78.
[107] Marco A Wiering et al. 2002. Evolving causal neural networks. In *Benelearn'02: Proceedings of the Twelfth Belgian-Dutch Conference on Machine Learning*. 103–108.
[108] Raymond W. Yeung. 2008. *Information Theory and Network Coding* (1 ed.). Springer Publishing Company, Incorporated.
[109] Quinton Yong, Mahdi Hajiabadi, Venkatesh Srinivasan, and Alex Thomo. 2021. Efficient graph summarization using weighted lsh at billion-scale. In *Proceedings of the 2021 International Conference on Management of Data*. 2357–2365.

[110] Brit Youngmann, Michael Cafarella, Amir Gilad, and Sudeepa Roy. 2024. Summarized Causal Explanations For Aggregate Views. *Proceedings of the ACM on Management of Data* 2, 1 (2024), 1–27.

[111] Brit Youngmann, Michael Cafarella, Yuval Moskovitch, and Babak Salimi. 2023. On Explaining Confounding Bias. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. IEEE, 1846–1859.

[112] Brit Youngmann, Michael Cafarella, Babak Salimi, and Zeng Anna. 2023. Causal Data Integration. *Proceedings of the VLDB Endowment* 16, 1- (2023), 2665–2659.

[113] Jiji Zhang. 2008. On the completeness of orientation rules for causal discovery in the presence of latent confounders and selection bias. *Artificial Intelligence* 172, 16-17 (2008), 1873–1896.

[114] Boxiang Zhao, Shuliang Wang, Lianhua Chi, Qi Li, Xiaojia Liu, and Jing Geng. 2023. Causal Discovery via Causal Star Graphs. *ACM Transactions on Knowledge Discovery from Data* 17, 7 (2023), 1–24.

[115] Yang Zhou, Hong Cheng, and Jeffrey Xu Yu. 2009. Graph clustering based on structural/attribute similarities. *Proceedings of the VLDB Endowment* 2, 1 (2009), 718–729.

[116] Jiongli Zhu, Sainyam Galhotra, Nazanin Sabri, and Babak Salimi. 2023. Consistent Range Approximation for Fair Predictive Modeling. *Proceedings of the VLDB Endowment* 16, 11 (2023), 2925–2938.

[117] Jiongli Zhu and Babak Salimi. 2024. Overcoming Data Biases: Towards Enhanced Accuracy and Reliability in Machine Learning. *IEEE Data Eng. Bull.* 47, 1 (2024), 18–35.

[118] Shengyu Zhu, Ignavier Ng, and Zhitang Chen. 2020. Causal Discovery with Reinforcement Learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

A SEMIGRAPHOID-AXIOMS

The semi-graphoid axioms are the following:

(1) Triviality: $I(A;\emptyset|C){=}0$.
(2) Symmetry: $I(A;B|C){=}0 \implies I(A;C|B){=}0$.
(3) Decomposition: $I(A;BD|C){=}0 \implies I(A;B|D){=}0$.
(4) Contraction: $I(A;B|C){=}0, I(A;D|BC){=}0 \implies I(A;BD|C){=}0$.
(5) Weak Union: $I(A;BD|C){=}0 \implies I(A;B|CD){=}0, I(A;D|BC){=}0$.

The semi-graphoid axioms can be summarized using the following identity, which follows from the *chain-rule* for mutual information [108].

$$I(A;BD|C){=}0 \text{ if and only if } I(A;B|C){=}0 \text{ and } I(A;D|BC){=}0 \qquad (5)$$

B PROOFS

Next, we provide the missing proofs.

We begin with some basic definitions used in the proofs.

Let $\mathcal{G}$ be a causal DAG. and let $U, V{\in}\mathsf{V}(\mathcal{G})$ be two nodes. We say that U is a *parent* of V, and V a *child* of U if $(U \to V) \in \mathsf{E}(\mathcal{G})$. A *directed path* $t{=}(V_1, \dots, V_n)$ is a sequence of nodes such that there is an edge $(V_i{\to}V_{i+1}){\in}\mathsf{E}(\mathcal{G})$ for every $i{\in}\{1, \dots, n{-}1\}$. We say that V is a *descendant* of U, and U an *ancestor* of V if there is a directed path from U to V. We denote the child-nodes of V in $\mathcal{G}$ as $\mathsf{ch}_{\mathcal{G}}(V)$; the descendants of V (we assume that $V{\in}\mathsf{Dsc}_{\mathcal{G}}(V)$) as $\mathsf{Dsc}_{\mathcal{G}}(V)$, and the nodes of $\mathcal{G}$ that are not descendants of V as $\mathsf{NDsc}_{\mathcal{G}}(V)$. For a set of nodes $\mathsf{S}{\subseteq}\mathsf{V}(\mathcal{G})$, we let $\mathsf{Dsc}_{\mathcal{G}}(\mathsf{S})\overset{\text{def}}{=}\bigcup_{U\in\mathsf{S}}\mathsf{Dsc}_{\mathcal{G}}(U)$, and by $\mathsf{NDsc}_{\mathcal{G}}(\mathsf{S})\overset{\text{def}}{=}\bigcap_{U\in\mathsf{S}}\mathsf{NDsc}_{\mathcal{G}}(U)$.

A *trail* $t{=}(V_1, \dots, V_n)$ is a sequence of nodes such that there is an edge between V_i and V_{i+1} for every $i{\in}\{1, \dots, n{-}1\}$. That is, $(V_i{\to}V_{i+1}){\in}\mathsf{E}(\mathcal{G})$ or $(V_i{\leftarrow}V_{i+1}){\in}\mathsf{E}(\mathcal{G})$ for every $i{\in}\{1, \dots, n{-}1\}$. A node V_i is said to be *head-to-head* with respect to t if $(V_{i-1}{\to}V_i){\in}\mathsf{E}(\mathcal{G})$ and $(V_i{\leftarrow}V_{i+1}){\in}\mathsf{E}(\mathcal{G})$. A trail $t = (V_1, \dots, V_n)$ is *active* given $\mathbf{Z}{\subseteq}\mathcal{V}$ if (1) every V_i that is a head-to-head node with respect to t either belongs to $\mathbf{Z}$ or has a descendant in $\mathbf{Z}$, and (2) every V_i that is not a head-to-head node w.r.t. t does not belong to $\mathbf{Z}$. If a trail t is not active given $\mathbf{Z}$, then it is *blocked* given $\mathbf{Z}$.

B.1 Proofs for Section 3

Proof of Lemma 3.1. Let P be a directed path from A to B, such that $|P| \geq 2$. Let X be A's successor in P, and Y be B's predecessor in P. By our assumption that $|P| \geq 2$, $X \notin \{A, B\}$, but Y may be the same as X. Now, consider the graph H. By definition, H contains a node AB, with an incoming edge from Y, and an outgoing edge to X. If $X = Y$, we immediately get the cycle $AB \to X \to AB$. Otherwise, we consider the subpath P' (of P) from X to Y ($X \underset{P'}{\rightsquigarrow} Y$) in G. This results in the following cycle in H: $Y \to AB \to X \underset{P'}{\rightsquigarrow} Y$.

Now, suppose that H contains the cycle C. C must contain the node AB. Otherwise, the cycle is included in G, which leads to a contradiction that G is a DAG. Let Y and X be the incoming and outgoing vertices, respectively, to AB in C. Then, there is a directed path P from X to Y in H that avoids AB. That is, every vertex and edge on the path P belongs to G as well. Hence, P is a directed path from X to Y in G. Since Y is incoming to AB, then Y is incoming to either A or B in G. Assume, wlog, that $Y \to A \in E$. Since X is an outgoing vertex from AB, then it is outgoing from from either A or B (or both). If X is outgoing from A, then we get the following cycle in G: $Y \to A \to X \underset{P}{\rightsquigarrow} Y$. Since G is a DAG, this brings us to a contradiction. Therefore, X must be outgoing from B and not A. But this gives us the following directed path from B to A:

$$A \leftarrow Y \underset{P}{\leftsquigarrow} X \leftarrow B.$$

This completes the proof. □

Next, we show that the causal DAG summarization problem is NP-hard via a reduction from the k-Max-Cut problem [31]. Let G be an undirected graph with weighted edges (i.e., $w : \mathsf{V}(G) \to \mathbb{R}_{\geq 0}$). The k-Max-Cut problem consists of partitioning $\mathsf{V}(G)$ into k disjoint clusters so as to maximize the sum of weights of the edges joining vertices in different clusters. It is well-known that k-Max-Cut is NP-hard even if $k = 2$ [31], and the weight of every edge is 1. In other words, deciding whether there exists a k-clustering of $\mathsf{V}(G)$ to clusters $\{V_1, \dots, V_k\}$, where the sum of weights of edges between vertices in distinct clusters is at least a given threshold γ is NP-hard.

Let $\mathcal{S} \overset{\text{def}}{=} \{V_1, \dots, V_k\}$ be a k-clustering of G. We define $M_G(\mathcal{S})$ to be the number of edges of between vertices in a common cluster:

$$M_G(\mathcal{S}) \overset{\text{def}}{=} \sum_{i=1}^{k} \sum_{u,v\in V_i} \mathbb{1}[(u, v) \in \mathsf{E}(G)]. \qquad (6)$$

We define $\overline{M}_G(\mathcal{S})$ to be the number of non-edges between vertices in a common cluster:

$$\overline{M}_G(\mathcal{S}) \overset{\text{def}}{=} \sum_{i=1}^{k} \sum_{u,v\in V_i} \mathbb{1}[(u, v) \notin \mathsf{E}(G)]. \qquad (7)$$

Similarly, we define $B_G(\mathcal{S})$ to be the number of edges of between vertices in distinct clusters:

$$B_G(\mathcal{S}) \overset{\text{def}}{=} \sum_{1\leq i<j\leq k} \sum_{\substack{u\in V_i,\\ v\in V_j}} \mathbb{1}[(u, v) \in \mathsf{E}(G)]. \qquad (8)$$

Since every edge $(u, v) \in \mathsf{E}(G)$ is either between vertices in the same cluster or vertices in distinct clusters, then:

$$M_G(\mathcal{S}) + B_G(\mathcal{S}) = |\mathsf{E}(G)|.$$

In particular, $M_G(\mathcal{S}) \leq \tau$ if and only if $B_G(\mathcal{S}) \geq |\mathsf{E}(G)| - \tau$, for every $\tau \in [0, |\mathsf{E}(G)|]$.

Lemma B.1. *Let G be an undirected graph, $k \geq 2$, and $\tau \in [0, |\mathsf{E}(G)|]$. Deciding whether there exists a k-clustering $\mathcal{S}$ of G such that $M_G(\mathcal{S}) \leq \tau$ is NP-Complete.*

Proof. The problem is clearly in NP because given a k-clustering $\mathcal{S}$, computing $M_G(\mathcal{S})$ (see (6)) can be done in polynomial time.

We prove hardness by reduction from k-Max-Cut. Suppose there exists a P-Time algorithm that given an undirected graph G, and a threshold value $\tau \in [0, |\mathsf{E}(G)|]$, returns a k-clustering $\mathcal{S}$ such that $M_G(\mathcal{S}) \leq \tau$ if one exists, and null otherwise. We show that such an algorithm can be applied to solve k-Max-Cut in polynomial time.

Let G, k, and γ be an instance of the k-max-Cut problem where G is an undirected graph, $k \leq |\mathsf{V}(G)|$, and $\gamma \in [0, |\mathsf{E}(G)|]$. We define $\tau \overset{\text{def}}{=} |\mathsf{E}(G)| - \gamma$, and execute the algorithm for finding a k-clustering $\mathcal{S}$ such that $M_G(\mathcal{S}) \leq \tau$. Since $M_G(\mathcal{S}) + B_G(\mathcal{S}) = |\mathsf{E}(G)|$, then:

$$\begin{aligned}
B_G(\mathcal{S}) &= |\mathsf{E}(G)| - M_G(\mathcal{S}) && \Rightarrow_{M_G(\mathcal{S}) \leq \tau} \\
&\geq |\mathsf{E}(G)| - \tau && \Rightarrow_{\tau \overset{\text{def}}{=} |\mathsf{E}(G)| - \gamma} \\
&= |\mathsf{E}(G)| - (|\mathsf{E}(G)| - \gamma) \\
&= \gamma
\end{aligned}$$

Hence, we can decide in P-Time whether G has a k-cut whose cardinality is at least γ. □

Lemma B.2. *Let G be an undirected graph, $k \geq 2$, and $\tau \in [0, |\mathsf{E}(G)|]$. Deciding whether there exists a k-clustering $\mathcal{S}$ of G such that $\overline{M}_G(\mathcal{S}) \leq \tau$ is NP-Complete.*

Proof. The problem is clearly in NP because given a k-clustering $\mathcal{S}$, computing $\overline{M}_G(\mathcal{S})$, and verifying $\overline{M}_G(\mathcal{S}) \leq \tau$ can be done in polynomial time (see (7)).

So, suppose that there exists a P-Time algorithm that given an undirected graph G, and a threshold value $\tau \in [0, |\mathsf{E}(G)|]$, returns a k-clustering $\mathcal{S}$ such that $\overline{M}_G(\mathcal{S}) \leq \tau$ if one exists, and null otherwise. We show that such an algorithm can be applied to decide whether there exists a k-clustering of G where $M_G(\mathcal{S}) \leq \alpha$. By Lemma B.1, this problem is NP-Hard, and hence this will prove that minimizing $\overline{M}_G(\mathcal{S})$ is NP-hard as well.

So let G, k, and α be the input to the problem for deciding whether there exists a k-clustering $\mathcal{S}$ such that $M_G(\mathcal{S}) \leq \alpha$. Let $\overline{G}$ denote the complement graph of G. That is $\mathsf{V}(\overline{G}) = \mathsf{V}(G)$ and $(u, v) \in \mathsf{E}(G)$ if and only if $(u, v) \notin \mathsf{E}(\overline{G})$. In particular, $\overline{M}_{\overline{G}}(\mathcal{S}) = \alpha$ if and only if $M_G(\mathcal{S}) = \alpha$. Therefore, if we can, in P-Time find a clustering that minimizes $\overline{M}_{\overline{G}}(\mathcal{S})$, then we have also found a clustering that minimizes $M_G(\mathcal{S})$. This completes the proof. □

We now show that the causal DAG summarization problem is NP-hard. Specifically, we show that finding a summary DAG $(\mathcal{H}, f)$ whose canonical DAG $\mathcal{G}$ minimizes $|\mathsf{E}(\mathcal{G}_{\mathcal{H}})| - |\mathsf{E}(G)|$; that is, the canonical DAG $\mathcal{G}_{\mathcal{H}}$ results in the smallest number of added edges, is NP-Hard.

Theorem B.3. *The causal DAG summarization problem which minimizes the number of added edges to the canonical DAG is NP-Hard.*

Proof. Given a DAG D and a k-clustering $\mathcal{S}$ of D, it is straightforward to verify that $\overline{M}_D(\mathcal{S}) \leq \tau$, and hence this problem is in NP.

Let G be an undirected graph, $k > 0$, and $\gamma > 0$ a threshold. Let $\mathsf{V}(G) = \{v_1, \dots, v_n\}$ denote a complete order of $\mathsf{V}(G)$. Define D to be the directed graph where $\mathsf{V}(D) = \mathsf{V}(G)$ and $(v_i \rightarrow v_j) \in \mathsf{E}(D)$ if and only if $(v_u, v_j) \in \mathsf{E}(G)$ and $i < j$. Let $\mathcal{S} = \{V_1, \dots, V_k\}$ be a k-clustering of $\mathsf{V}(D) = \mathsf{V}(G)$ such that $\overline{M}_D(\mathcal{S}) \leq \gamma$. By the definition of $\mathsf{E}(D)$, we immediately get that $\overline{M}_G(\mathcal{S}) \leq \gamma$. By Lemma B.2, the causal DAG summarization problem is NP-hard. □

B.2 Proofs for Section 4

Next, we prove some simple lemmas that will be useful later on. We denote by $\mathcal{H}_{UV}$ the summary DAG where nodes U and V are contracted.

Lemma B.4. *The following holds:*

$$\pi_{\mathcal{H}_{UV}}(\mathbf{X_{UV}}) = \pi_{\mathcal{G}_{UV}}(U) \qquad \pi_{\mathcal{G}_{UV}}(V) = \pi_{\mathcal{G}_{UV}}(U) \cup \{U\} \tag{9}$$

$$\mathsf{ch}_{\mathcal{H}_{UV}}(\mathbf{X_{UV}}) = \mathsf{ch}_{\mathcal{G}_{UV}}(V) \qquad \mathsf{ch}_{\mathcal{G}_{UV}}(U) = \mathsf{ch}_{\mathcal{G}_{UV}}(V) \cup \{V\} \tag{10}$$

$$\mathsf{NDsc}_{\mathcal{H}_{UV}}(\mathbf{X_{UV}}) = \mathsf{NDsc}_{\mathcal{G}_{UV}}(U) \qquad \mathsf{NDsc}_{\mathcal{G}_{UV}}(V) = \mathsf{NDsc}_{\mathcal{G}_{UV}}(U) \cup \{U\} \tag{11}$$

Lemma B.5. *Let $T \in \mathcal{V}$ such that $T \notin \{U, V\} \cup \mathsf{ch}_{\mathcal{G}}(U) \cup \mathsf{ch}_{\mathcal{G}}(V)$. Then it holds that:*

$$\pi_{\mathcal{G}_{UV}}(T) = \pi_{\mathcal{H}_{UV}}(T) \textit{ and} \tag{12}$$

$$\mathsf{NDsc}_{\mathcal{G}_{UV}}(T) \backslash \{UV\} = \mathsf{NDsc}_{\mathcal{H}}(T) \backslash \{\mathbf{X_{UV}}\} \textit{ and} \tag{13}$$

$$\{U, V\} \subseteq \mathsf{NDsc}_{\mathcal{G}_{UV}}(T) \textit{ if and only if } \mathbf{X_{UV}} \in \mathsf{NDsc}_{\mathcal{H}}(T) \tag{14}$$

Now, let $T \in \mathcal{V}$ such that $T \notin \{U, V\} \cup \pi_{\mathcal{G}}(U) \cup \pi_{\mathcal{G}}(V)$. Then:

$$\mathsf{ch}_{\mathcal{G}_{UV}}(T) = \mathsf{ch}_{\mathcal{H}_{UV}}(T) \textit{ and} \tag{15}$$

$$\mathsf{Dsc}_{\mathcal{G}_{UV}}(T) \backslash \{U, V\} = \mathsf{Dsc}_{\mathcal{H}_{UV}}(T) \backslash \{\mathbf{X_{UV}}\} \tag{16}$$

$$\{U, V\} \subseteq \mathsf{Dsc}_{\mathcal{G}_{UV}}(T) \textit{ if and only if } \mathbf{X_{UV}} \in \mathsf{Dsc}_{\mathcal{H}_{UV}}(T) \tag{17}$$

proof of Lemma B.4. By the definition of G', it holds that $\pi_{G'}(u) = \pi_G(u) \cup \pi_G(v)$. Since $(u, v) \in \mathsf{E}(G')$, then $\pi_{G'}(v) = \pi_{G'}(u) \cup \{u\}$. Similarly, $\mathsf{ch}_{G'}(v) = \mathsf{ch}_G(u) \cup \mathsf{ch}_G(v)$, and since $(u, v) \in \mathsf{E}(G')$, then $\mathsf{ch}_{G'}(u) = \mathsf{ch}_{G'}(v) \cup \{v\}$. By definition of edge contraction, it holds that $\pi_H(X_{uv}) = \pi_G(u) \cup \pi_G(v) = \pi_{G'}(u)$, proving (9). Also, by definition of edge contraction, it holds that $\mathsf{ch}_H(X_{uv}) = \mathsf{ch}_G(u) \cup \mathsf{ch}_G(v) = \mathsf{ch}_{G'}(v)$, proving (10).

We now prove (11). Let $t \in \mathsf{NDsc}_H(X_{uv})$. If $t \notin \mathsf{NDsc}_{G'}(u)$, then $t \in \mathsf{Dsc}_{G'}(u) \backslash \{v\}$. This means that there is a directed path P from u to t in G'. Let s be the first vertex on this path (after u). Since $s \in \mathsf{ch}_{G'}(u) \backslash \{v\}$, then by the definition of G', $s \in \mathsf{ch}_G(u) \cup \mathsf{ch}_G(v)$. By the definition of edge contraction, $s \in \mathsf{ch}_H(X_{uv})$. Since $s \notin uv \cup \pi_G(u) \cup \pi_G(v)$, then every directed path starting at s in G remains a directed path in H. But this means that there is a directed path from X_{uv} to t (via s); contradicting the assumption that $t \in \mathsf{NDsc}_H(X_{uv})$. Now, let $t \in \mathsf{NDsc}_{G'}(u)$. If $t \notin \mathsf{NDsc}_H(X_{uv})$, then $t \in \mathsf{Dsc}_H(X_{uv}) \backslash \{X_{uv}\}$. This means that there is a directed path P from X_{uv} to t in H. Let s be the first vertex on this path (after X_{uv}). Since $s \in \mathsf{ch}_H(X_{uv})$, then by the definition of H, $s \in \mathsf{ch}_G(u) \cup \mathsf{ch}_G(v)$. But then, by the definition of G', it holds that $s \in \mathsf{ch}_{G'}(u)$. Since no edges are removed by the transition from G to G', there is a directed path from u to (via s) in G'; contradicting the assumption that $t \in \mathsf{NDsc}_{G'}(u)$. □

Proof of Lemma B.5. By the definition of contraction, the only vertices in G whose parent-set can potentially change following the contraction of u and v belong to the set $uv \cup \mathsf{ch}_G(u) \cup \mathsf{ch}_G(v)$. By the definition of $\mathsf{E}(G')$, the only vertices in G whose parent-set can potentially change belong to the set $uv \cup \mathsf{ch}_G(u) \cup \mathsf{ch}_G(v)$. Therefore, if $t \notin uv \cup \mathsf{ch}_G(u) \cup \mathsf{ch}_G(v)$, then $\pi_{G'}(t) = \pi_H(t) = \pi_G(t)$. This proves (12).

By the definition of contraction, the only vertices in G whose child-set can potentially change following the contraction of u and v belong to the set $uv \cup \pi_G(u) \cup \pi_G(v)$. By the definition of $\mathsf{E}(G')$, the only vertices in G whose child-set can potentially change belong to the set $uv \cup \pi_G(u) \cup \pi_G(v)$. Therefore, if $t \notin uv \cup \pi_G(u) \cup \pi_G(v)$, then $\mathsf{ch}_{G'}(t) = \mathsf{ch}_H(t) = \mathsf{ch}_G(t)$. This proves (15).

We now prove (13); Let $s \in \mathsf{NDsc}_H(t) \backslash \{X_{uv}\}$. If $s \notin \mathsf{NDsc}_{G'}(t)$, then $s \in \mathsf{Dsc}_{G'}(t)$. That is, there is a directed path P from t to s in G'. Let us assume wlog that P is the shortest directed path from t to s in G'. By this assumption, exactly one of the following holds: (1) $u, v \notin \mathsf{V}(P)$ (2) $u \in \mathsf{V}(P)$, $v \notin nodes(P)$ (3) $v \in \mathsf{V}(P)$, $u \notin nodes(P)$, or (4) $(u,v) \in \mathsf{E}(P)$. In the first case, every edge of P is also an edge of $\mathsf{E}(G)$, that does not enter or exit $\{u,v\}$. Therefore, P is a directed path in H, a contradiction. In case (2), since $\pi_{G'}(u) = \pi_H(X_{uv})$ (see (9)), and $\mathsf{ch}_{G'}(u) = \mathsf{ch}_H(X_{uv}) \backslash \{v\}$ (see (10)), then the path with nodes $X_{uv} \cup (\mathsf{V}(P) \setminus \{u\})$, is a directed path in H from s to t; a contradiction. In case (3), since $\mathsf{ch}_{G'}(v) = \mathsf{ch}_H(X_{uv})$ (see (10)), and $\pi_{G'}(v) = \pi_H(X_{uv}) \backslash \{u\}$ (see (9)), then the path with nodes $X_{uv} \cup (\mathsf{V}(P) \setminus \{v\})$, is a directed path in H from s to t; a contradiction. Finally, if $(u,v) \in \mathsf{E}(P)$, then since $\pi_{G'}(u) = \pi_H(X_{uv})$ and $\mathsf{ch}_{G'}(v) = \mathsf{ch}_H(X_{uv})$, then the path with nodes $X_{uv} \cup (\mathsf{V}(P) \backslash uv)$, is a directed path from s to t in H; a contradiction. For the other direction, let $s \in \mathsf{NDsc}_{G'}(t) \backslash uv$. If $s \notin \mathsf{NDsc}_H(t)$, then there is a directed path P from t to s in H. If $X_{uv} \notin \mathsf{V}(P)$, then $\mathsf{E}(P) \subseteq \mathsf{E}(G) \subseteq \mathsf{E}(G')$, and hence P is a directed path from t to s in G'. Otherwise, if $X_{uv} \in \mathsf{V}(P)$, then since $\pi_H(X_{uv}) = \pi_{G'}(u)$, $\mathsf{ch}_H(X_{uv}) = \mathsf{ch}_{G'}(v)$, and $(u,v) \in \mathsf{E}(G')$, then replacing X_{uv}, with the edge (u,v) results in a directed t,s-path in G'; a contradiction. □

Proof of Theorem 4.1. We first prove that $\Sigma_{\text{RB}}(G') \implies \Sigma_{\text{RB}}(H)$. We divide to cases. Let $(X_i; B_i | \pi_H(X_i)) \in \Sigma_{\text{RB}}(H)$, where $X_{uv} \notin B_i \cup \pi_H(X_i) \cup \{X_i\}$. In particular, $X_i \in \mathsf{V}(G)$, and $X_i \notin \{u,v\} \cup \mathsf{ch}_H(X_{uv}) = \{u,v\} \cup \mathsf{ch}_G(u) \cup \mathsf{ch}_G(v)$. By (13), we have that $\pi_{G'}(X_i) = \pi_H(X_i)$, and that $\mathsf{NDsc}_{G'}(X_i) \backslash uv = \mathsf{NDsc}_H(X_i) \backslash X_{uv}$. Therefore, we have that $(X_i; \mathsf{NDsc}_H(X_i) \backslash \{X_{uv}\} | \pi_{G'}(X_i))_{G'}$. Since $B_i \subseteq \mathsf{NDsc}_H(X_i) \backslash \{X_{uv}\}$, then, by decomposition, we have that $\Sigma_{\text{RB}}(G') \implies (X_i; B_i | \pi_H(X_i))$.

Now, let $(X_i; X_{uv} B_i | \pi_H(X_i)) \in \Sigma_{\text{RB}}(H)$. In this case as well $X_i \in \mathsf{V}(G)$, and $X_i \notin \{u,v\} \cup \mathsf{ch}_H(X_{uv}) = \{u,v\} \cup \mathsf{ch}_G(u) \cup \mathsf{ch}_G(v)$. By (13), we have that $\pi_{G'}(X_i) = \pi_H(X_i)$, and that $\mathsf{NDsc}_{G'}(X_i) \backslash uv = \mathsf{NDsc}_H(X_i) \backslash X_{uv}$. By , we have that $X_{uv} \in \mathsf{NDsc}_H(X_i)$ iff $uv \subseteq \mathsf{NDsc}_{G'}(X_i)$. Therefore, $B_i X_{uv} \subseteq \mathsf{NDsc}_H(X_i)$ iff $B_i uv \subseteq \mathsf{NDsc}_{G'}(X_i)$. This means that $\Sigma_{\text{RB}}(G') \implies (X_i; \mathsf{NDsc}_{G'}(X_i) | \pi_{G'}(X_i))$. By decomposition, we have that $\Sigma_{\text{RB}}(G') \implies (X_i; B_i uv | \pi_H(X_i))$ as required.

Now, suppose that $X_{uv} \in \pi_H(X_i)$, or that $X_i \in \mathsf{ch}_H(X_{uv})$. Since $X_i \in \mathsf{V}(G) \backslash \{u,v\}$, then by (10), we have that $X_i \in \mathsf{ch}_{G'}(v) \backslash \{v\}$. Therefore, $\pi_{G'}(X_i) = \pi_H(X_i) \backslash \{X_{uv}\} \cup \{u,v\}$. By (13), we have that:

$$\begin{aligned}
\mathsf{NDsc}_{G'}(X_i) \backslash \pi_{G'}(X_i) &= \mathsf{NDsc}_{G'}(X_i) \backslash (\pi_H(X_i) \backslash \{X_{uv}\} \cup uv) \\
&= (\mathsf{NDsc}_{G'}(X_i) \backslash uv) \backslash (\pi_H(X_i) \backslash \{X_{uv}\}) \\
&\underbrace{=}_{(13)} (\mathsf{NDsc}_H(X_i) \backslash \{X_{uv}\}) \backslash (\pi_H(X_i) \backslash \{X_{uv}\}) \\
&= \mathsf{NDsc}_H(X_i) \backslash \pi_H(X_i)
\end{aligned}$$

Therefore, $\Sigma_{\text{RB}}(G') \implies (X_i; \mathsf{NDsc}_H(X_i) \backslash \pi_H(X_i) \mid \pi_H(X_i) \backslash \{X_{uv}\} \cup \{u,v\})$. Finally, we consider the case where $X_i = X_{uv}$. By construction of G', and by (11), it holds that:

$$\Sigma_{\text{RB}}(G') \implies (u; \mathsf{NDsc}_{G'}(u) \backslash \pi_{G'}(u) \mid \pi_{G'}(u)) \tag{18}$$

$$\Sigma_{\text{RB}}(G') \implies (v; \mathsf{NDsc}_{G'}(u) \backslash \pi_{G'}(u) \mid \pi_{G'}(u) \cup \{u\}) \tag{19}$$

By applying the contraction axiom on (18) and (19), we get that

$$\Sigma_{\text{RB}}(G') \implies (uv; \mathsf{NDsc}_{G'}(u) \backslash \pi_{G'}(u) \mid \pi_{G'}(u)).$$

Using the fact that hat $\pi_H(X_{uv}) = \pi_{G'}(u)$ (see (9)), and that $\mathsf{NDsc}_H(X_{uv}) = \mathsf{NDsc}_{G'}(u)$ (see (11)), we get that

$$\Sigma_{\text{RB}}(G') \implies (uv; \mathsf{NDsc}_H(X_{uv}) \backslash \pi_H(X_{uv}) \mid \pi_H(X_{uv})).$$

Since $B_i \subseteq \mathsf{NDsc}_H(X_{uv}) \backslash (\pi_H(X_{uv}) \cup \{X_{uv}\})$, this proves the claim.

Now, for the other direction. Let $(X_i; B_i \mid \pi_{G'}(X_i)) \in \Sigma_{\text{RB}}(G')$. If $u, v \notin X_i \cup B_i \cup \pi_{G'}(X_i))$, then $X_i \notin \{u,v\} \cup \mathsf{ch}_G(u) \cup \mathsf{ch}_G(v)$. By (13), it holds that $\pi_H(X_i) = \pi_{G'}(X_i)$, and that $\mathsf{NDsc}_{G'}(X_i) \backslash uv = \mathsf{NDsc}_H(X_i) \backslash X_{uv}$. Since $B_i \subseteq \mathsf{NDsc}_{G'}(X_i) \backslash uv = \mathsf{NDsc}_H(X_i) \backslash X_{uv}$, then $\Sigma_{\text{RB}}(H) \implies (X_i; B_i \mid \pi_{G'}(X_i))$.

If $uv \subseteq B_i$, then $u, v \notin \pi_{G'}(X_i)$, then $X_i \notin uv \cup \mathsf{ch}_G(u) \cup \mathsf{ch}_G(v)$. By (13), we have that $\pi_H(X_i) = \pi_{G'}(X_i)$, and that $\mathsf{NDsc}_H(X_i) \backslash X_{uv} = \mathsf{NDsc}_{G'}(X_i) \backslash uv$. Therefore, $B_i \backslash uv \subseteq \mathsf{NDsc}_H(X_i)$, and by (14), if $uv \subseteq B_i \subseteq \mathsf{NDsc}_{G'}(X_i)$, then $X_{uv} \in \mathsf{NDsc}_H(X_i)$. Therefore, $\Sigma_{\text{RB}}(H) \implies (X_i; B_i \backslash uv \cup X_{uv} \mid \pi_{G'}(X_i))$, and since $X_{uv} = uv$, then $\Sigma_{\text{RB}}(H) \implies (X_i; B_i \mid \pi_{G'}(X_i))$.

Since $(u,v) \in \mathsf{E}(G')$, we are left with two other cases. First, that $(u; B_u \mid \pi_{G'}(u))$, and second $(v; B_v \mid \pi_{G'}(v))$. By d-separation in H, the following holds:

$$\begin{aligned}
\Sigma_{\text{RB}}(H) &\implies (X_{uv}; \mathsf{NDsc}_H(X_{uv}) \backslash \pi_H(X_{uv}) \mid \pi_H(X_{uv})) \\
&\underbrace{\implies}_{(9),\,(11)} (X_{uv}; \mathsf{NDsc}_{G'}(u) \backslash \pi_{G'}(u) \mid \pi_{G'}(u)) \\
&\implies (uv; \mathsf{NDsc}_{G'}(u) \backslash \pi_{G'}(u) \mid \pi_{G'}(u))
\end{aligned} \tag{20}$$

By (9), it holds that $\pi_{G'}(v) = \pi_{G'}(u) \cup \{u\}$. By (11), it holds that

$$\begin{aligned}
B_v \subseteq \mathsf{NDsc}_{G'}(v) \backslash \pi_{G'}(v) &= (\mathsf{NDsc}_{G'}(u) \cup \{u\}) \backslash (\pi_{G'}(u) \cup \{u\}) \\
&= \mathsf{NDsc}_{G'}(u) \backslash \pi_{G'}(u)
\end{aligned}$$

Therefore, $B_v \cup B_u \subseteq \mathsf{NDsc}_{G'}(u) \backslash \pi_{G'}(u)$. In other words, by (20), we have that;

$$\begin{aligned}
&\Sigma_{\text{RB}}(H) \implies (uv; B_u \cup B_v \mid \pi_{G'}(u)) \text{ if and only if} \\
&\Sigma_{\text{RB}}(H) \implies (u; B_u \cup B_v \mid \pi_{G'}(u)), (v; B_u \cup B_v \mid \pi_{G'}(u) \cup \{u\})
\end{aligned}$$

Since $\pi_{G'}(v) = \pi_{G'}(u) \cup \{u\}$, then overall, we have that $\Sigma_{\text{RB}}(H) \implies (u; B_u \mid \pi_{G'}(u))$, and $\Sigma_{\text{RB}}(H) \implies (v; B_v \mid \pi_{G'}(v))$. This completes the proof. □

To prove Theorem 4.2, we first show the following lemma that establishes the connection between d-separation on the canonical causal DAG and the original causal DAG.

Lemma B.6. *Let $\mathcal{G}$ and $\mathcal{G}'$ be causal DAGs defined over the same set of nodes, i.e., $\mathsf{V}(\mathcal{G}) = \mathsf{V}(\mathcal{G}')$, where $\mathcal{G}'$ is a supergraph of $\mathcal{G}$ ($\mathsf{E}(\mathcal{G}') \supseteq \mathsf{E}(\mathcal{G})$). Then, for any three disjoint subsets $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \subseteq \mathsf{V}(\mathcal{G})$, it holds that: $(\mathbf{X} \perp\!\!\!\perp_d \mathbf{Y}|\mathbf{Z})_{\mathcal{G}'} \implies (\mathbf{X} \perp\!\!\!\perp_d \mathbf{Y}|\mathbf{Z})_{\mathcal{G}}$*

Proof of Lemma B.6. Suppose that X and Y are d-separated by Z in G' (i.e., $(X; Y|Z)_{G'}$). If X and Y are d-connected by Z in G, then let P denote the unblocked path between X and Y, relative to Z. Since $\mathsf{E}(G') \supseteq \mathsf{E}(G)$, then clearly P is a path in G' as well. Consider any triple (x, w, y) on this path. If this triple has one of the forms

$$\{x \to w \to y, x \leftarrow w \leftarrow y, x \leftrightarrow w \to y, x \leftarrow w \leftrightarrow y\},$$

then since P is unblocked in G, relative to Z, then $w \notin Z$. Since $w \notin Z$, then the subpath (x, w, y) is also unblocked in G'. If the triple has one of the forms:

$$\{x \leftrightarrow w \leftarrow y, x \to w \leftrightarrow y, x \to w \leftarrow y\},$$

then since P is unblocked in G, relative to Z, then $\mathsf{Dsc}_G(w) \cap Z \neq \emptyset$. Since $\mathsf{E}(G') \supseteq \mathsf{E}(G)$, then $\mathsf{Dsc}_G(w) \subseteq \mathsf{Dsc}_{G'}(w)$. Therefore, $\mathsf{Dsc}_{G'}(w) \cap Z \neq \emptyset$. Consequently, we have, again, that the subpath (x, w, y) is unblocked in G'. Overall, we get that every triple (x, w, y) on the path P is unblocked in G', relative to Z, and hence X and Y are d-connected in G', a contradiction.

By definition, G' is compatible with G'. Therefore, if X and Y are d-connected by Z in G', then by the completeness of d-separation, there exists a probability distribution that factorizes according to G' in which the CI $(X; Y|Z)$ does not hold. This proves completeness. □

Proof of Theorem 4.2. Let $\mathcal{G}_{\mathcal{H}}$ denote the canonical causal DAG corresponding to $\mathcal{H}$. By Theorem 4.1, $\Sigma_{\text{RB}}(\mathcal{H}) \equiv \Sigma_{\text{RB}}(\mathcal{G}_{\mathcal{H}})$. Therefore, $(\mathbf{X} \perp\!\!\!\perp_d \mathbf{Y}|\mathbf{Z})_{\mathcal{H}} \iff (f^{-1}(\mathbf{X}) \perp\!\!\!\perp_d f^{-1}(\mathbf{Y})|f^{-1}(\mathbf{Z}))_{\mathcal{G}_{\mathcal{H}}}$ Since $\mathsf{E}(\mathcal{G}) \subseteq \mathsf{E}(\mathcal{G}_{\mathcal{H}})$, the claim immediately follows from Lemma B.6. □

B.3 Proofs for Section 6

We next show a smile lemma that will be useful for proving the soundness and completeness of do-calculus in summary graphs.

Lemma B.7. *Let G be ADMG, and let G' be an ADMG where $\mathsf{V}(G') = \mathsf{V}(G)$, and $\mathsf{E}(G') \supseteq \mathsf{E}(G)$. Let $A, B, C \subseteq \mathsf{V}(G)$ be disjoint sets of variables, and let $X, Z \subseteq \mathsf{V}(G)$. Then:*

$$(A; B|C)_{G'_{\overline{X}\underline{Z}}} \implies (A; B|C)_{G_{\overline{X}\underline{Z}}} \tag{21}$$

Corollary B.7.1. *Let G be ADMG, and let (H, f) be a summary-DAG for G. Let $A, B, C \subseteq \mathsf{V}(H)$ be disjoint sets of nodes, and let $X, Z \subseteq \mathsf{V}(H)$. Then:*

$$(A; B|C)_{H_{\overline{X}\underline{Z}}} \implies (\mathbf{A}; \mathbf{B}|\mathbf{C})_{G_{\overline{\mathbf{X}}\underline{\mathbf{Z}}}} \tag{22}$$

where for $U \subseteq \mathsf{V}(H)$, we denote $\mathbf{U} \overset{def}{=} f(U)$.

Theorem B.8 (Soundness of Do-Calculus in supergraphs). *Let $\mathcal{G}$ be a causal DAG encoding an interventional distribution $P(\cdot \mid do(\cdot))$. Let $\mathcal{G}'$ be a causal DAG where $\mathsf{V}(\mathcal{G}) = \mathsf{V}(\mathcal{G}')$ and $\mathsf{E}(\mathcal{G}) \subseteq \mathsf{E}(\mathcal{G}')$. For any disjoint subsets $\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \mathbf{W} \subseteq \mathsf{V}(\mathcal{G})$, the following three rules hold:*

$$\mathbf{R}_1: \quad (\mathrm{Y} \perp\!\!\!\perp \mathrm{Z}|\mathrm{X}, \mathbf{W})_{\mathcal{G}'_{\overline{\mathrm{X}}}} \implies P(\mathrm{Y} \mid do(\mathrm{X}), \mathrm{Z}, \mathbf{W}) = P(\mathrm{Y} \mid do(\mathrm{X}), \mathbf{W})$$

$$\mathbf{R}_2: \quad (\mathrm{Y} \perp\!\!\!\perp \mathrm{Z}|\mathrm{X}, \mathbf{W})_{\mathcal{G}'_{\overline{\mathrm{X}}\underline{\mathrm{Z}}}} \implies P(\mathrm{Y} \mid do(\mathrm{X}), do(\mathrm{Z}), \mathbf{W}) = P(\mathrm{Y} \mid do(\mathrm{X}), \mathrm{Z}, \mathbf{W})$$

$$\mathbf{R}_3: \quad (\mathrm{Y} \perp\!\!\!\perp \mathrm{Z}|\mathrm{X}, \mathbf{W})_{\mathcal{G}'_{\overline{\mathrm{XZ}(\mathbf{W})}}} \implies P(\mathrm{Y} \mid do(\mathrm{X}), do(\mathrm{Z}), \mathbf{W}) = P(\mathrm{Y} \mid do(\mathrm{X}), \mathbf{W})$$

where $\mathbf{Z}(\mathbf{W})$ is the set of nodes in $\mathbf{Z}$ that are not ancestors of any node in $\mathbf{W}$. That is, $\mathbf{Z}(\mathbf{W}) = \mathbf{Z} \backslash \mathsf{Ancs}_{\mathcal{G}'}(\mathbf{W})$ where $\mathsf{Ancs}_{\mathcal{G}'}(\mathbf{W}) \overset{def}{=} \bigcup_{W \in \mathbf{W}} \mathsf{Ancs}_{\mathcal{G}'}(W)$.

Theorem B.9 (Soundness of Do-Calculus in supergraphs). *Let G be a causal BN (CBN) encoding an interventional distributions $P(\cdot \mid do(\cdot))$. Let G' be an ADMG where $\mathsf{E}(G) \subseteq \mathsf{E}(G')$. For any disjoint subsets $X, Y, Z, W \subseteq \mathsf{V}(G)$, the following three rules hold:*

$$\mathbf{R}_1: \quad (Y; Z|X, W)_{G'_{\overline{X}}} \implies P(Y \mid do(X), Z, W) = P(Y \mid do(X), W)$$

$$\mathbf{R}_2: \quad (Y; Z|X, W)_{G'_{\overline{X}\underline{Z}}} \implies P(Y \mid do(X), do(Z), W) = P(Y \mid do(X), Z, W)$$

$$\mathbf{R}_3: \quad (Y; Z|X, W)_{G'_{\overline{XZ(W)}}} \implies P(Y \mid do(X), do(Z), W) = P(Y \mid do(X), W)$$

where $Z(W)$ is the set of vertices in Z that are not ancestors of any vertex in W. That is, $Z(W) = Z \backslash \mathsf{Ancs}_{G'}(W)$ where $\mathsf{Ancs}_{G'}(W) \overset{def}{=} \bigcup_{w \in W} \mathsf{Ancs}_{G'}(w)$.

Proof of Lemman B.7. We show that $\mathsf{E}(G_{\overline{X}\underline{Z}}) \subseteq \mathsf{E}(G'_{\overline{X}\underline{Z}})$, and the claim then follows from Theorem ??. Let $(u, v) \in \mathsf{E}(G_{\overline{X}\underline{Z}}) \subseteq \mathsf{E}(G) \subseteq \mathsf{E}(G')$. By definition, $u \notin Z$ and $v \notin X$. But this means that $(u, v) \in \mathsf{E}(G'_{\overline{X}\underline{Z}})$, which completes the proof. □

Proof of Corollary B.7.1. Let $G_{H_{\overline{X}\underline{Z}}}$ denote the grounded DAG corresponding to $H_{\overline{X}\underline{Z}}$. By Theorem 4.1, $\Sigma_{\text{RB}}(H_{\overline{X}\underline{Z}}) \equiv \Sigma_{\text{RB}}(G_{H_{\overline{X}\underline{Z}}})$, and hence $(A; B|C)_{H_{\overline{X}\underline{Z}}}$ if and only if $(\mathbf{A}; \mathbf{B}|\mathbf{C})_{G_{H_{\overline{\mathbf{X}}\underline{\mathbf{Z}}}}}$. Since $\mathsf{E}(G_H) \supseteq \mathsf{E}(G)$, then by Lemma B.7, it holds that if $(\mathbf{A}; \mathbf{B}|\mathbf{C})_{G_{H_{\overline{\mathbf{X}}\underline{\mathbf{Z}}}}}$, then $(\mathbf{A}; \mathbf{B}|\mathbf{C})_{G_{\overline{\mathbf{X}}\underline{\mathbf{Z}}}}$. Overall, we have that:

$$(A; B|C)_{H_{\overline{X}\underline{Z}}} \Leftrightarrow (\mathbf{A}; \mathbf{B}|\mathbf{C})_{G_{H_{\overline{\mathbf{X}}\underline{\mathbf{Z}}}}} \implies (\mathbf{A}; \mathbf{B}|\mathbf{C})_{G_{\overline{\mathbf{X}}\underline{\mathbf{Z}}}} \tag{23}$$

which proves the claim. □

Proof of Theorem B.9. If $(Y; Z|X, W)_{G'_{\overline{X}}}$, then by Lemma B.7, it holds that $(Y; Z|X, W)_{G_{\overline{X}}}$. By the soundness of do-calculus for causal BNs, we get that $P(\boldsymbol{Y} \mid do(\boldsymbol{X}), \boldsymbol{Z}, \boldsymbol{W}) = P(\boldsymbol{Y} \mid do(\boldsymbol{X}), \boldsymbol{W})$. If $(Y; Z|X, W)_{G'_{\overline{X}\underline{Z}}}$, then by Lemma B.7, it holds that $(Y; Z|X, W)_{G_{\overline{X}\underline{Z}}}$. By the soundness of do-calculus for causal BNs, we get that $P(Y \mid do(X), do(Z), W) = P(Y \mid do(X), Z, W)$. Finally, if $(Y; Z|X, W)_{G'_{\overline{XZ(W)}}}$, then by Lemma B.7, it holds that $(Y; Z|X, W)_{G_{\overline{XZ(W)}}}$. By the soundness of do-calculus for causal BNs, we get that $P(Y \mid do(X), do(Z), W) = P(Y \mid do(X), W)$. □

Proof of Theorem 6.1. If $(Y; Z|X, W)_{H_{\overline{X}}}$, then by Corollary B.7.1, it holds that $(\boldsymbol{Y}; \boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{W})_{G_{\overline{X}}}$. By the soundness of do-calculus for causal BNs, we get that $P(\boldsymbol{Y} \mid do(\boldsymbol{X}), \boldsymbol{Z}, \boldsymbol{W}) = P(\boldsymbol{Y} \mid do(\boldsymbol{X}), \boldsymbol{W})$. If $(Y; Z|X, W)_{H_{\overline{X}\underline{Z}}}$, then by Corollary B.7.1, it holds that $(\boldsymbol{Y}; \boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{W})_{G_{\overline{X}\underline{Z}}}$. By the soundness of do-calculus for causal BNs, we get that $P(\boldsymbol{Y} \mid do(\boldsymbol{X}), do(\boldsymbol{Z}), w) = P(\boldsymbol{Y} \mid do(\boldsymbol{X}), \boldsymbol{Z}, \boldsymbol{W})$. Finally, if $(Y; Z|X, W)_{H_{\overline{XZ(W)}}}$, then by Corollary B.7.1, it holds that $(\boldsymbol{Y}; \boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{W})_{G_{\overline{X}\underline{Z(W)}}}$. By the soundness of do-calculus for causal BNs, we get that $P(\boldsymbol{Y} \mid do(\boldsymbol{X}), do(\boldsymbol{Z}), \boldsymbol{W}) = P(\boldsymbol{Y} \mid do(\boldsymbol{X}), \boldsymbol{W})$. □

Proof of Theorem 6.2. Consider G_H, the grounded-DAG of (H, f), that is, by definition, compatible with H. If Y is d-connected to Z in $H_{\overline{X}}$ with respect to $X \cup W$, then by Definition 5, it holds that y is d-connected to z in $G_{H_{\overline{f(X)}}}$ with respect to $f(X \cup W)$, for every $y \in f(Y)$ and $z \in f(Z)$. Therefore, $f(Y)$ is d-connected to $f(Z)$ in $G_{H_{\overline{f(X)}}}$ with respect to $f(X \cup W)$. □

C HANDLING MIXED GRAPHS

While one dominant form of graph input for causal inference is a causal DAG, other graph representations are also used when a full causal DAG is not retrievable, say, by a causal discovery algorithm (e.g., [76]). Many of these graph representations are referred to as *mixed graphs* due to their inclusion of undirected, bidirected, and other types of edges [20, 77].

One commonly used of mix graph is an acyclic-directed mixed graph (ADMG), which consists of a DAG with bidirected edges. As mentioned in the introduction, all of our results apply to scenarios where the input graph is an ADMG. Subsequently, we present an extension to the CaGreS algorithm to accommodate an ADMG.

In this scenario, we modify the cost function (Algorithm 2) as follows: When we remove a bidirected edge between nodes U and V (i.e., $U \leftrightarrow V$) by merging U and V into a single node, the cost incurred is doubled compared to removing a "standard" directed edge. For instance, in line 4 of Algorithm 2, if **U** and **V** were linked by a bidirected edge, the line would be updated to:

$$cost \longleftarrow cost + 2 \cdot size(\mathbf{U}) \cdot size(\mathbf{V})$$

This adjustment is necessary because losing a bidirected edge should carry a higher cost than losing a regular directed edge, given that more information is lost.