# Causal Graph Summarization

by

Anna Zeng

B.S., Stanford University (2018)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2023



Authored by: Anna Zeng
Department of Electrical Engineering and Computer Science
May 19, 2023

Certified by: Michael Cafarella
Principal Research Scientist
MIT Computer Science Artificial Intelligence Laboratory
Thesis Supervisor

Accepted by: Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

# Causal Graph Summarization

by

Anna Zeng

Submitted to the Department of Electrical Engineering and Computer Science
on May 19, 2023, in partial fulfillment of the
requirements for the degree of
Master of Science

## Abstract

Causal inference is critical for scientific progress, especially in social sciences like public health and education—however, analysts often only have access to partial data which may lead to erroneous conclusions if critical confounding biases are not accounted for. To do this, they critically rely on (often unavailable or incomplete) domain knowledge to identify attributes to include for causal analysis, which is often tediously, manually specified in the form of a causal DAG.

Given state-of-the-art methods, analysts might automatically gather and causally organize a much more comprehensive set of attributes to include in their analysis; however, at best, such tools provide large, nearly-complete causal graphs which are difficult to comprehend, let alone verify to use in causal analysis tasks; as these graphs get bigger and denser with the growth of automated causal discovery methods, domain experts will struggle to comprehend, interpret, and correct causal graphs for practical applications. Existing methods for graph summarization developed in other domains, such as graphics, social networking, and mapping, are not guaranteed to provide a summarized graph eligible for use in causal analysis tasks; some methods even result in introducing spurious causal relationships that render erroneous conclusions if used in causal analysis.

We hypothesize that causality-specific graph summarization algorithms could surmount these challenges. To demonsrate this, we introduce CAMBA, a prototype causal graph summarization algorithm that efficiently generates high-quality causal graph summaries that are interpretable and usable for causal inference. In this thesis, we formalize the Causal DAG Summarization problem, identify a causal information metric, extend causal inference foundations to summary graphs, identify graph summarization techniques which can preserve this causal information, and propose a range of possible causal-specific graph summarization optimizations, and evaluate such methods on a range of causal analysis scenarios.

# Acknowledgments

This whole thing simply wouldn't be possible without my advisor Mike Cafarella. Your everlasting enthusiasm about what I'm going to do—and same for any of your advisees—is probably what powers the Stata building. The kind of support I've felt from you has made it possible for me to learn that this kind of academic work is achievable, enjoyable, and meaningful, and to grow into the shoes of an academic researcher. Leslie Kolodziejski, you've been there since my first year, bringing boiled eggs and breakfast every Friday morning, and eager to lend a listening ear—especially as the Epstein and Stallman developments unfolded here at MIT. Without a doubt, huge thanks to the Presidential Jacobs Family Fellowship and the Center for Advancing Safety of Machine Intelligence for your financial support and giving me the foundational stability to embark on this work.

No amount of coffee beans in the world can express how much I appreciate Brit Youngmann, without whom I would not have ever finished this! Your bubbling excitement and sharp eye for what's important and what's not ('bluh-bla-bla-blah') has served as a critical guidepost for how to live a balanced—and effective—academic life. Babak Salimi, your straight-to-the-point guidance—always with a smile!—and appeal towards doing things *right* has been a huge help in making this work technically coherent. If it weren't for your timeless enthusiasm and everlasting support during my undergraduate degree, I wouldn't be in academia at all: Professor Pat Hanrahan, Will Crichton, and Professor Dawson Engler, thank you so much, I am immensely humbled and grateful for your support. Likewise, many thanks to Professors Tim Kraska, Martin Rinard, Mike Carbin, Daniel Jackson, and Elena Glassman, as your guidance and support was instrumental in helping me continue growing as an academic here at MIT. I'd also like to thank Armando Solar-Lezama and Rastislav Bodík for your academic guidance leading up to this point. Also, thank you, Ibrahim Sabek for our discussions about positioning and drafting the narrative of this project. Of course, to Matt, Joana, Siva, Jialin, Kapil, Tianyu, and the rest of Data Systems Group (DSG)—thank you for the all-round support and camaraderie. Our in-house rock band, Impedence Mismatch, has been a wonderful experience in whole-hearted noise-making—especially our thrilling first gig this past Wednesday! (When we're old we'll chuckle to think about it.)

My partner Calvin has been an absolute bedrock of support—staying up late with me to support my progress and momentum, debugging algorithms together, keeping my software engineering game afloat—and words fail to describe how much that helped make this possible. Immense thanks to all of my friends in dance (Lillian Zhu, my Harvard waltz co-instructor; Daniel Mark, Quan Nguyen, and others from the MIT Lindy Hop Society; those in the Argentine Tango Society), the MIT Rowing Club, and the Graduate Women in Course 6, who instill in me the energy to persist and to give. Last but not least, thank you to my patient mom, dad, and brother, who are willing to come visit me all the way in Boston when I'm hunkered down simply learning how academic research works.

# Contents

# List of Figures

# List of Tables

# List of Algorithms

# Chapter 1

# Introduction

Scientific progress can be viewed as an exercise in causal analysis: understanding systems of cause-and-effect relationships which influence some data-generating or observable phenomena. In practice, many disciplines, including sociology, medicine, and economics [29, 15, 63], commonly study causal mechanisms which are:

- *probabilistic*, such as "Are frequent exposures to second-hand smoke related to a different chance of getting lung cancer compared with the general population?" [39, 42],

- *type-level*, as opposed to singular, as in "If students spend more time studying, do they get better test scores?" rather than "If Joe spends more time studying, would Joe get a better math test grade?" [42],

- *partially observable*, where analysts do not assume they have observations of all causal mechanisms contributing to an outcome, as in "How do we study genetic inheritance without full knowledge of which organisms having which genes?" [60], and

- *high-dimensional*, as in "Given patients' electronic medical records of emergency room and hospital visits, what cause-and-effect relationships about human health can we infer?" [62].

Knowledge about a collection of causal mechanisms is commonly manually curated in the form of a *causal graph*; this graph is then broadly distributed in survey papers as a form of communication and used in causal analysis tasks like causal inference [42, 36, 38].

Literature in causal inference which builds upon Pearl's causal model [43] formally describes how to conduct causal analysis on causal directed acyclic graphs (DAGs) and resolves limitations with correlative analysis, like confounding bias. Efforts like [26] illustrate the benefits of conducting *causal* analysis rather than correlative analysis—the insights found in causal analysis can directly translate into actionable interventions. Such interventions—like policy changes, shifts in treatment plans, experiment ideas, and more—provide more promise than correlative insights due to the mitigation of confounding and other biases.

However, to determine a sufficient set of confounding variables to account for confounding bias, *background knowledge* about the data generative process is required. This knowledge is often given in the form of a *causal DAG* depicting causal relationships between the attributes [43] (and is wildly used in econometric and social sciences [15, 29]). Pearl [43] presented sufficient and necessary conditions for identifying[1] the *adjustment set* of variables to include in the analysis, which can be checked against a causal DAG. However, analysts often lack such a DAG [12]. While associational assumptions are testable from data, causal relations cannot, in general, be fully recovered from data [39, 63]. With the development of causal discovery [53, 57], data integration [33], data mining [63], and *causal* integration methods [62], such causal graphs can be constructed automatically, but they lack fundamental guarantees which limit practical applicability. With this in mind, we illustrate the perspective of an analyst looking to conduct causal analysis to determine the impact that the choice of airport has on flight departure delay:

**Example 1.** Jan is an analyst in the US Department of Transportation, looking to conduct causal analysis to determine the direct impact that the choice of airport

[1]In causal inference, identifiability typically refers to the conditions that permit measuring causal effect from observed data. Here, we discuss the ability to identify a sufficient adjustment set of variables for accurate causal inference.

has on flight departure delay. Jan has the data detailing flight departures across the country, like the departure airport and flight number, but it's missing key confounding variables, like weather-related attributes including precipitation and airline-related attributes like number of staff members.

Jan reasons that these attributes are feasibly related to the phenomenon of interest, but requires more background knowledge of causal relationships to determine which of these variables are confounding variables to account for. Here, variables missing from the dataset are critical for the analysis to avoid confounding bias. Thus, Jan must *integrate the data* with external data sources using data discovery and integration tools (e.g., [33, 64]). After augmenting the data to include the thousands of newly found variables, the next step is determining a sufficient adjustment set of variables to account for confounding bias. Since Jan does not have the required background knowledge to do that, Jan uses a causal discovery tool (e.g., [47, 53]) to automatically construct a causal DAG and identify the sufficient adjustment set. For the sake of illustration, we include Figure 1-1 as a visual guide.

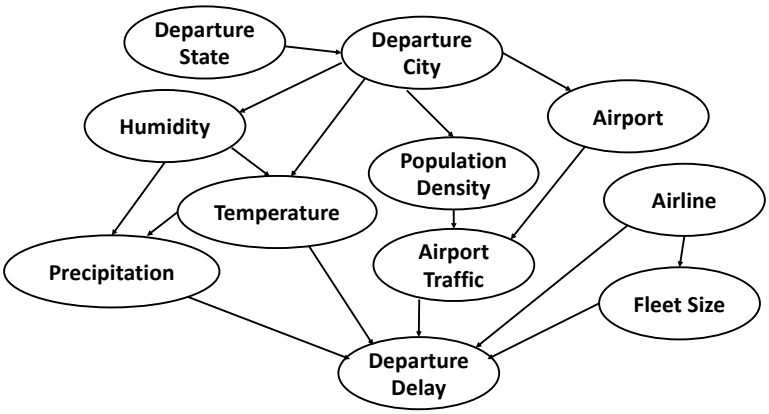


Figure 1-1: 'Full' Causal DAG from Example 1. For clarity, the mentioned causal DAG is simplified to 11 nodes.

After letting the tool process all of this newly-augmented dataset and automati-

cally discover a causal DAG, Jan finds that the algorithm suggests an adjustment set of more than 200 individual attributes, many of which are meaningfully overlapping. Jan does not have the required background knowledge to determine which subset of those attributes are actually necessary to include in the adjustment set for conducting the desired analysis; Jan is not convinced that using all automatically-suggested 200 attributes point-blank lends confidence to the causal conclusions of the analysis. Even if Jan notices any incorrect edges in the causal DAG, Jan has no idea how to correct the causal DAG without disrupting useful causal relationships (or lack thereof) that the automated system discovered.

Jan wished to have had a causal DAG that looked much like Figure 1-2, with the summarized graph involving only two nodes to be included in the adjustment set.

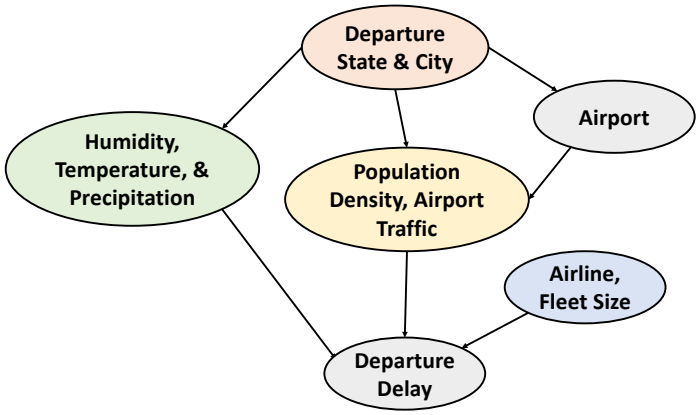


Figure 1-2: Summarized Causal DAG from Example 1.

As we saw in Jan's scenario, state-of-the-art in causal inference struggles to scale in both graph size and scope of applicability. Causal graphs suitable for use in Pearl's causal framework [43] require a number of assumptions which are difficult to demonstrate adherence to in practical settings, including: sufficiency (there are no unobserved confounders); faithfulness (the graph captures all conditional independence

relations between the variables); the Markov property (every variable is independent of its non-descendants, given its parents); and the DAG assumption (the causal graph is a directed acyclic graph); depending on the specific causal graph discovery and inference method used, additional distributional assumptions (assuming all causal contributions are functionally linear, for example) and implicit data modeling assumptions (adequate attribute granularity and a lack of functional dependencies, for example) [17, 12, 7] also must apply. Most causal graphs found in causal inference literature contain around 10 or 20 causal attributes [57] which are curated manually or semi-manually [42, 36, 38]; while methods are capable of incorporating large populations of study units, often represented as rows in observational relational data, our current methods are not built for high-dimensional datasets with significantly more causal attributes.

Data summarization has made big data analytics feasible [10, 61, 58, 19, 21, 22]; in particular, graph summarization has succeeded in road, social, and computer network analysis [49, 16, 20, 30]. To combat the limitations of current causal analysis methods and enable them to scale to high-dimensional datasets, we can endeavor to summarize the graph before use in causal analysis tasks. However, the state-of-the-art in graph summarization is inadequate for graphs with cause-and-effect relationships. The semantics of an edge in a causal graph differ significantly from, say, that of a road network—summarization by outright edge or node removal would introduce spurious causal information in a causal graph, whereas visualizing a road network would be much faster and just as accurate by not drawing the residential side streets. Even when compared with a Bayesian network, the semantics of edge directionality results in different solution spaces for graph summarization. Consequentially, objectives like minimum description length (MDL), edge minimization, conductance, or flow minimization do not necessarily have out-of-the-box alignment with summarization in the causal graph domain.

We endeavor to explore whether causality-specific graph summarization methods could surmount this causal analysis challenge, while preserving critical invariants that enable incorporation into causal analysis with minimal loss of quality. In this work,

we present the following contributions:

1. We formalize the Causal DAG Summarization problem in which a large causal graph can be summarized for comprehensibility and causal information retention.

2. We define a structural causal information metric of a summarized graph which does not rely on any information beyond the labeled causal graph.

3. We introduce $C$-separation, an extension of a foundational Pearl causal analysis method, $d$-separation, for use on summarized causal graphs.

4. We characterize graph summarization operations that are sound relative to this metric.

5. We present a brute force algorithm to concretely quantify this structural metric for any given causal graph.

6. We further develop fast, approximate methods to estimate this structural causal information metric.

7. We present CAMBA, our illustrative prototype of a causal graph summarization method, to highlight opportunities for greedy, bottom-up, and top-down causality-specific optimizations.

8. We demonstrate performance and outcome characteristics of these algorithms on a range of causal analysis scenarios.

# Chapter 2

# Data Model and Background

An analyst is interested in investigating the causal relationship between an *exposure attribute* $T$ and an *outcome* $O$. To do so, she investigates an input dataset $\mathcal{D}$ containing the exposure $T$ and outcome $O$ attributes. Here, we use upper case letters (e.g., $X$) to represent an attribute from $\mathcal{D}$ and bold symbols (e.g., $\boldsymbol{C}$) to represent a set of attributes. A *causal DAG* $\mathcal{G}$ for $\mathcal{D}$ is a directed acyclic graph (DAG) whose nodes $\boldsymbol{V}$ are the attributes in $\mathcal{D}$ and whose edges $\boldsymbol{E}$ capture all causal relationships between the attributes [43]. Generally, because causes must precede effects, the literature considers causal DAGs rather than graphs with cycles [43]. Causal DAGs provide a simple way of graphically representing causal relationships and are particularly helpful in understanding potential sources of bias in causal estimations.

Using this dataset, the analyst may wish to estimate, e.g., the *direct, indirect, or total effects* [43] of $T$ on $O$. The total effect is the overall degree to which $O$ is altered by $T$, which is the sum of the direct and indirect effects. To estimate the total effect, an *adjustment set* of variables is necessary, which includes *confounding variables* that influence both $T$ and $O$. For the direct effect, *mediating variables* that mediate the relationship between $T$ and $O$ additionally need to be identified. The adjustment sets can be identified using graphical criteria (e.g., the backdoor or the front door criterion [43]) that can be checked against a causal DAG. Another important type of variable is a *collider*. A variable is a collider if it is causally influenced by two or more variables; namely, a variable is a collider if its node $V$ in the causal DAG $\mathcal{G}$ has

two inbound edges. We illustrate *confounding*, *mediating*, and *collider* variables in Example 2 below:

**Example 2.** Consider the causal DAG shown in Figure 1-1 that displays causal relationships in a flight delays dataset. In this diagram, the variable `temperature` acts as a confounding variable for the `precipitation` and `flight departure delay` variables. The variable `population density` serves as a mediator for the relationship between `departure city` and `airport traffic`. The variable `flight departure delay` acts as a collider variable between `temperature` and `fleet size`.

As we will explain next, it is important to carefully consider the presence of collider variables in causal inference, as they can significantly impact the results and interpretation of the analysis.

## 2.1 Causal Interpretation

A *causal diagram* (also known as a causal structure or causal graph) is a directed graph where each edge encodes the existence of a functional contribution from one variable to another that relates to real-world underlying *causal* mechanisms; restated, it is a directed graph with a *causal interpretation* [42]. Ideally, causal DAGs contain nodes that represent all variables in a causal system and edges that represent all functional contributions that relate to underlying causal mechanisms in the causal systems [39].

Causal DAGs encode conditional independence relationships between variables which constitute foundational, testable assertions about the functional mechanisms underlying the causal system of interest [40]. In our work, we treat these relationshipsas the information content of the graph that we stand to lose in the process of summarization. A *graphoid* is a more specific term which can be used to describe a set of conditional independence statements read from a causal DAG.

## 2.2 $d$-separation

A causal DAG's set of Conditional Independence statements (CIs) can be read off the graph using $d$-separation [42]. These statements describe the absence of a direct causal relationship between variables when conditioning on other variables. A *path* is a sequence of adjacent edges from one node to another regardless of directionality (e.g., `fleet size`—`departure delay`—`airline`). We say that two nodes $X$ and $Y$ are *d-separated* by a set of variables $\boldsymbol{Z}$ in causal DAG $G$, denoted as $(X \perp\!\!\!\perp Y \mid \boldsymbol{Z})$, if, for every path between them, one of the following conditions holds: (1) the path contains a *chain* $(X \rightarrow Z \rightarrow Y)$ or a *fork* $(X \leftarrow Z \rightarrow Y)$ such that $Z \in \boldsymbol{Z}$ and (2) the path contains a *collider* $(X \rightarrow Z \leftarrow Y)$ such that $Z \notin \boldsymbol{Z}$, and no descendants of $Z$ are in $\boldsymbol{Z}$. For example, `population density` $\perp\!\!\!\perp$ `airport` | `departure city`.

## 2.3 Quotient Graphs

In graph theory, a quotient graph of a graph $\mathcal{G} = (V, E)$ is a graph whose nodes are partitioned into blocks according to a given equivalence relation between the nodes in $V$ [4]. Give a partition $P$ on $V$, written $P = \{\boldsymbol{C_1}, \ldots, \boldsymbol{C_k}\}$, where each block $\boldsymbol{C_i}$ is a subset of nodes (such that $\boldsymbol{V} = \bigcup_{i=1,\ldots,k} \boldsymbol{C_i}$ and $\boldsymbol{C_i} \cap \boldsymbol{C_j} = \emptyset$ for $i \neq j$), the quotient graph of $\mathcal{G}$ with respect to $P$, denoted as $\mathcal{G}_P = \langle \boldsymbol{V}_P, \boldsymbol{E}_P \rangle$, is a graph with the nodes $\boldsymbol{V}_P = \{\boldsymbol{C_1}, \boldsymbol{C_2}, \ldots, \boldsymbol{C_k}\}$. $\boldsymbol{E}_P$ is the set of edges where we have an edge $(\boldsymbol{C_i}, \boldsymbol{C_j}) \in \boldsymbol{E}_P$ if and only if there exists a node $X \in \boldsymbol{C_i}$ and a node $Y \in \boldsymbol{C_j}$ such that $(X, Y) \in \boldsymbol{E}$.

Given a graph $\mathcal{G} = (V, E)$ and partition of the nodes $P$, we obtain a quotient graph $\mathcal{G}_P$ by applying *vertex contraction* operations [45]. The contraction of a pair of nodes $X, Y \in V$ is the operation that produces a graph in which the two nodes $X$ and $Y$ are replaced with a single node $\boldsymbol{C} = \{X, Y\}$ such that $\boldsymbol{C}$ is adjacent to the union of the nodes to which $X$ and $Y$ were originally adjacent (directionality is preserved). In vertex contraction, it does not matter if $X$ and $Y$ are connected by an edge; if they are, the edge is removed upon contraction.

We say that a graph $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ is *compatible* with a quotient graph $\mathcal{Q}$ if there

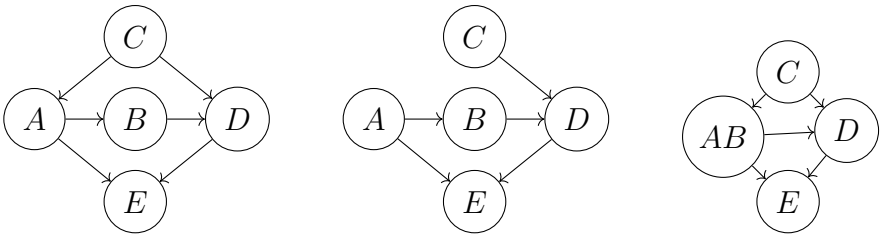


Figure 2-1: There are three graphs shown, with a quotient graph located on the right-hand side. The graph on the left is compatible with the quotient graph, while the one in the middle is not.

is a partition $P$ of $\boldsymbol{V}$ such that when we apply it to $\mathcal{G}$ using vertex contractions, the result is the quotient graph $\mathcal{Q}$.

Figure 2-1 depicts two graphs and a quotient graph. The graph on the left is compatible with the quotient graph (achieved by contacting the nodes $A$ and $B$). The graph in the middle is not compatible with the quotient graph, since neither $A$ nor $B$ are connected to $C$.

# Chapter 3

# Problem Formulation

Informally, given a causal DAG $\mathcal{G}$ with an exposure $T$ and an outcome $O$, our goal is to summarize $\mathcal{G}$ by grouping related attributes and only specifying causal relationships between those groups of attributes. By doing so, a sufficient adjustment set of variables (to be accounted for to answer a causal question of the relationship between $T$ and $O$) should still be identifiable from this summarized causal DAG. Here, we introduce a few essential prerequisite topics and concretize our goal into a formal problem description.

## 3.1 Quotient Causal DAGs

A quotient causal DAG $\mathcal{G}_P = (\mathcal{V}_P, \mathcal{E}_P)$ is a DAG that is derived from some causal DAG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ via vertex contraction according to the partition $P$. To ensure the quotient graph is a DAG, only vertex contraction operations that result in no cycles are permitted. We say that the resulting quotient causal DAG is a summary of the original causal DAG. In what follows, if the original causal DAG is unknown, we denote a quotient causal DAG using the symbol $\mathcal{G}$.

A quotient causal DAG $\mathcal{G}_P$ is obtained by partitioning the nodes of the original causal DAG $\mathcal{G}$ into sets of nodes, thus losing the information about the causal relationships of nodes within each set. This means that some paths in $\mathcal{G}$ are no longer represented in $\mathcal{G}_P$, and $\mathcal{G}_P$ may also represent paths that did not exist in $\mathcal{G}$.

Nevertheless, we prove that the following proposition holds:

**Proposition 1.** *A vertex contraction operation applied on two nodes of a causal DAG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ does not introduce new conditional independencies in the obtained quotient causal DAG $\mathcal{G}_P$.*

*Proof.* By contradiction, we claim we can introduce a new conditional independence relation to $\mathcal{G}_P$ via some acyclicity-preserving vertex contraction operation. Given that the global Markov property holds in causal DAGs, we will use $d$-separation as the means to discuss conditional independence relations in $\mathcal{G}$. Therefore, the introduction of a new conditional independence relation $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} | \boldsymbol{Z}$ (where $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}$ are all mutually exclusive) would mean the following must occur: $\boldsymbol{X}$ is $d$-connected to $\boldsymbol{Y}$ given $\boldsymbol{Z}$ along at least one path in $\mathcal{G}$. After two vertices are contracted into one, we should find that all paths between $\boldsymbol{C}_X$ and $\boldsymbol{C}_Y$ are blocked and therefore $\boldsymbol{C}_X$ is $d$-separated from $\boldsymbol{C}_Y$ given $\boldsymbol{C}_Z$ in $\mathcal{G}_P$, producing the new conditional independence relation. We break this proof sketch down into the following cases:

**Case 1. Vertex contraction leads to $\boldsymbol{C}_X$ representing $X_1$ and $X_2$** Without loss of generality, let's say $\boldsymbol{C}_X$ is a block of nodes containing $X_1$ and $X_2$, where $X_1, X_2$ are nodes in $\mathcal{G}$ and $\boldsymbol{C}_X$ is a node in $\mathcal{G}_P$; also without loss of generality, let's say $X_1$ is $d$-connected to, or has an unblocked path to, $Y$ given $Z$ in $G$. The vertex contraction operation results in $\boldsymbol{C}_X$ in $\mathcal{G}$ where all edges relating $X_1, X_2$ to neighbors $N(X_1, X_2)$ would now relate $\boldsymbol{C}_X$ to $N(X_1, X_2)$. $\boldsymbol{C}_Y$, $\boldsymbol{C}_Z$, and all other nodes in $\mathcal{G}_P$ map to the corresponding node $Y$, $Z$, and so on in $\mathcal{G}$, and all edges amongst $V \setminus \{X_1, X_2\}$ are retained amongst $P \setminus \{\boldsymbol{C}_X\}$. By contradiction, we claim that $\boldsymbol{C}_X$ is $d$-separated from $\boldsymbol{C}_Y$ given $\boldsymbol{C}_Z$ in $\mathcal{G}_P$, producing the new conditional independence relation $\boldsymbol{C}_X \perp\!\!\!\perp \boldsymbol{C}_Y | \boldsymbol{C}_Z$ in $\mathcal{G}_P$. This statement represents $X_1, X_2 \perp\!\!\!\perp Y | Z$, implying that we have introduced $X_1 \perp\!\!\!\perp Y | Z$. However, since all edges between $X_1, X_2$ and their neighbors are preserved by $\boldsymbol{C}_X$, the $d$-connected path from $X_1$ to $Y$ given $Z$ in $\mathcal{G}$ is retained in $\mathcal{G}_P$ as a $d$-connected path from $\boldsymbol{C}_X$ to $\boldsymbol{C}_Y$ given $\boldsymbol{C}_Z$.

**Case 2. Vertex contraction leads to $\boldsymbol{C}_Y$ representing $Y_1$ and $Y_2$** Given that $d$-connectedness holds between $\boldsymbol{X}$ and $\boldsymbol{Y}$ regardless of direction, this case simplifies into Case 1.

**Case 3. Vertex contraction leads to $\boldsymbol{C}_Z$ representing $Z_1$ and $Z_2$** Without loss of generality, let's say $\boldsymbol{C}_Z$ is a block of nodes containing $Z_1$ and $Z_2$, where $Z_1, Z_2$ are nodes in $\mathcal{G}$ and $\boldsymbol{C}_Z$ is a node in $\mathcal{G}_P$; also without loss of generality, let's say when given $Z_1$ and $Z_2$, there is at least one unblocked, or $d$-connected, path between $X$ and $Y$. The vertex contraction operation results in $\boldsymbol{C}_Z$ in $\mathcal{G}$ where all edges relating $Z_1, Z_2$ to neighbors $N(Z_1, Z_2)$ would now relate $\boldsymbol{C}_Z$ to $N(Z_1, Z_2)$. $\boldsymbol{C}_X$, $\boldsymbol{C}_Y$, and all other nodes in $\mathcal{G}_P$ map to the corresponding node $X$, $Y$, and so on in $\mathcal{G}$, and all edges amongst $V \setminus \{Z_1, Z_2\}$ are retained amongst $P \setminus \{\boldsymbol{C}_Z\}$. By contradiction, we claim that $\boldsymbol{C}_X$ is $d$-separated from $\boldsymbol{C}_Y$ given $\boldsymbol{C}_Z$ in $\mathcal{G}_P$, producing the new conditional independence relation $\boldsymbol{C}_X \perp\!\!\!\perp \boldsymbol{C}_Y | \boldsymbol{C}_Z$ in $\mathcal{G}_P$ which represents $X \perp\!\!\!\perp Y | Z_1, Z_2$. However, this directly contradicts the premise that there is at least one unblocked, or $d$-connected, path between $\boldsymbol{C}_X$ and $\boldsymbol{C}_Y$ when given $\boldsymbol{C}_Z$. Since the vertex contraction operation preserved all edges relating $Z_1, Z_2$ to their neighbors in $\boldsymbol{C}_Z$'s edges, if any any $d$-connected path between $X$ and $Y$ through $Z_1$ and/or $Z_2$ existed, it would be persisted in the analogous path between $\boldsymbol{C}_X$ and $\boldsymbol{C}_Y$ through $\boldsymbol{C}_Z$. If the $d$-connected path between $X$ and $Y$ did not pass through $Z_1$ or $Z_2$, then the path is entirely unchanged, as each node along that path $N$ is replaced with a cluster $\boldsymbol{C}_N$ that contains just $N$.

**Case 4. Vertex contraction leads to $\boldsymbol{C}_A$ representing $A_1$ and $A_2$** Without loss of generality, let's say $\boldsymbol{C}_A$ is a block of nodes containing $A_1$ and $Z_2$, where $A_1, A_2$ are nodes in $\mathcal{G}$ and $\boldsymbol{C}_A$ is a node in $\mathcal{G}_P$, and when given $Z$, there is at least one unblocked, or $d$-connected, path between $X$ and $Y$. The vertex contraction operation maps all edges relating $A_1, A_2$ to its neighbors $N(A_1, A_2)$, and all other nodes and edges between $P \setminus \{\boldsymbol{C}_A\}$ in $\mathcal{G}_P$ map to the corresponding nodes and edges between $\boldsymbol{V} \setminus \{A_1, A_2\}$. By contradiction, we claim that $\boldsymbol{C}_X$ is now $d$-separated from $\boldsymbol{C}_Y$

given $\boldsymbol{C}_Z$, producing the new conditional independence relation $\boldsymbol{C}_X \perp\!\!\!\perp \boldsymbol{C}_Y | \boldsymbol{C}_Z$ in $\mathcal{G}_P$. Trivially, if that $d$-connected path between $X$ and $Y$ does not run through $A_1$ and/or $A_2$, the $d$-connected path is retained in $\mathcal{G}_P$. Otherwise, the $d$-connected path between $X$ and $Y$ which does run through $A_1$ and/or $A_2$ is also analogously retained in $\mathcal{G}_P$ due to adjacency preservation of the vertex contraction.

**Case 5.** The above cases can be naturally extended into situations where any nodes in $\mathcal{G}_P$ each represent multiple nodes in $\mathcal{G}$. □

However, we might lose some of the CIs's we had in a causal DAG $\mathcal{G}$ if we contract its nodes. This is illustrated in the following corollary:

**Corollary 1.1.** *The set of* CIs *induced by a quotient causal DAG is a subset of the set of* CIs *induced by the original causal DAG.*

Thus, a quotient causal DAG $\mathcal{G}$ can be seen as an intersection of all causal DAGs that are compatible with it. However, a remaining question is how to read off the set of all CIs induced by a quotient causal DAG $\mathcal{G}_P$.

## 3.2 $C$-separation

If two nodes $X$ and $Y$ are $d$-separated in all causal DAGs compatible with a given quotient causal DAG $\mathcal{G}$, then $\mathcal{G}$ should also contain this corresponding $d$-separation (completeness). Additionally, if $\mathcal{G}$ contains a $d$-separation between $X$ and $Y$, then this $d$-separation should hold in all causal DAGs compatible with $\mathcal{G}$ (soundness). However, not all CIs statements are expressible in a quotient causal DAG (we cannot directly read off from this graph any statement regarding subsets of nodes within a cluster node). To address this gap, we introduce the notion of $C$-separation, which extends the notion of $d$-separation for quotient causal DAGs.

Referring back to Figure 1-1, if we merged `humidity` and `temperature`, we'd still like to recover that `departure state` and `humidity` are $d$-separated, or conditionally independent from each other, given `departure city`.

**Definition 1** ($C$-separation)**.** A set of nodes $\boldsymbol{Z}$ $C$-separates node $X$ from node $Y$ in a graph $\mathcal{G} = \langle \boldsymbol{V}, \boldsymbol{E} \rangle$ compatible with quotient graph $\mathcal{G}_P = \langle \boldsymbol{V}_P, \boldsymbol{E}_P \rangle$ if and only if $\boldsymbol{Z}$ can be expressed on $\mathcal{G}_P$ where $\exists \boldsymbol{S} \subset \boldsymbol{V}_P . \bigcup_{\boldsymbol{C}_i \in \boldsymbol{S}} \boldsymbol{C}_i = Z$ and every path between $\boldsymbol{C}_X$ and $\boldsymbol{C}_Y$ in $\mathcal{G}_P$ contains adjacent nodes $A, B, C$ with one of the following patterns: (1) a $A \to B \to C$ pattern and $B \in \boldsymbol{C}_Z$, (2) a $A \leftarrow B \to C$ pattern and $B \in \boldsymbol{C}_Z$, or (3) a $A \to B \leftarrow C$ pattern while $B \notin \boldsymbol{C}_Z$ and any node with a directed path from $B$ is also not in $\boldsymbol{C}_Z$.

$C$-separation captures not only the CIs that are expressible directly by a quotient causal DAG $\mathcal{G}$ via $d$-separation, but also those that are *implied* by this DAG through $C$-separation.

Intuitively, we capture $C$-separation-expressible conditional independence statements from a quotient causal DAG using a two-step process: We first use $d$-separation to find all CIs expressible by the quotient casual DAG. Then, we extend this set of CIs by applying some *graphoid axioms* [44], resulting in additional CIs that hold in all causal DAGs that are compatible with the quotient casual DAG. By utilizing these axioms, we expand the set of CIs statements that can be directly read off from a quotient causal DAG (using $d$-separation) to include statements that are valid in all compatible causal DAGs, but cannot be obtained directly using $d$-separation.

For the completeness of this paper, we next list the graphoid axioms we use.

- **Symmetry**: $(\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} | \boldsymbol{Z}) \Rightarrow (\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{X} | \boldsymbol{Z})$
- **Decomposition**: $(\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} \bigcup \boldsymbol{W} | \boldsymbol{Z}) \Rightarrow (\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} | \boldsymbol{Z}) \& (\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{W} | \boldsymbol{Z})$
- **Weak Union**: $(\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} \bigcup \boldsymbol{W} | \boldsymbol{Z}) \Rightarrow (\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} | \boldsymbol{Z} \bigcup \boldsymbol{W}) \&$ $(\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{W} | \boldsymbol{Z} \bigcup \boldsymbol{Y})$

Given a quotient causal DAG $\mathcal{G}$, let $\mathcal{CI}_{\mathcal{G}}$ denote the set of all CIs that can be read off from $\mathcal{G}$ using $d$-separation, and let $\mathcal{CI}_{\mathcal{G}}^{imp}$ denote the set of statement after expanding $\mathcal{CI}_{\mathcal{G}}$ by applying the three axioms mentioned above to derive the implied CIs statements. The soundness of our extended $C$-separation for quotient causal DAGs is proved in the following theorem.

**Theorem 3.2.1.** *Given a quotient causal DAG $\mathcal{G}$ and its corresponding set of* CIs *$\mathcal{CI}_{\mathcal{G}}$, every conditional independent statement derived by applying the Symmetry, Decomposition, and Weak Union axioms hold in all causal DAGs that are compatible with $\mathcal{G}$.*

*Proof.* Without loss of generality, we say that $\boldsymbol{C}_X$ represents $X_1, X_2$; $\boldsymbol{C}_Y$ represents $Y$; and $\boldsymbol{C}_Z$ represents $Z$. Given that the global Markov property holds in $\mathcal{G}$, we find through $d$-separation that $\boldsymbol{C}_X$ is $d$-separated from $\boldsymbol{C}_Y$ given $\boldsymbol{C}_Z$, giving rise to $\boldsymbol{C}_X \perp\!\!\!\perp \boldsymbol{C}_Y | \boldsymbol{C}_Z$. Given the vertex contraction mechanism of construction, any $\mathcal{G}$-compatible causal DAG's edges between $X_1$ and $X_2$ are not preserved, but the edges relating $X_1$ and $X_2$ with their neighbors and all edges amongst $V \setminus \{X_1, X_2\}$ are retained in $\mathcal{G}$. Therefore, the presence of any $d$-connected path between $X_1$ and $Y$ given $Z$ or $X_2$ and $Y$ given $Z$ in any $\mathcal{G}$-compatible causal DAG would be preserved as a $d$-connected path between $\boldsymbol{C}_X$ and $\boldsymbol{C}_Y$ given $\boldsymbol{C}_Z$. Due to this contrapositive argument, we reason that if $\boldsymbol{C}_X$ and $\boldsymbol{C}_Y$ in $\mathcal{G}$ are $d$-separated given $\boldsymbol{C}_Z$, all compatible causal DAGs would also preserve the corresponding statement that $X_1, X_2$ is $d$-separated from $Y$ given $Z$, and consequentially also preserve $X_1, X_2 \perp\!\!\!\perp Y | Z$. Since the graphoid axioms of Symmetry, Decomposition, and Weak Union hold in all compatible causal DAGs, we can recover $Y \perp\!\!\!\perp X_1, X_2 | Z$, $X_1 \perp\!\!\!\perp Y | Z$, $X_2 \perp\!\!\!\perp Y | Z$, $X_1 \perp\!\!\!\perp Y | X_2, Z$, and $X_2 \perp\!\!\!\perp Y | X_1, Z$. Therefore, we can apply Symmetry, Decomposition, and Weak Union axioms to any CIs obtained from a quotient causal DAG to describe any compatible causal DAG. □

The above theorem demonstrates the soundness of $C$-separation; however, there is still an open question as to whether it is also complete. Specifically, can $C$-separation identify *all* CIs that hold across all compatible causal DAGs? Future research will focus on proving the completeness of our proposed $C$-separation.

## 3.3 Semantic Constraint

We next define the concept of a *semantic constraint* imposed on a quotient casual DAG in order to formally define the optimization problem studied in this paper.

Semantic similarity is a measure of the likeness of semantic content between two terms [14]; a group of concepts with high semantic similarity are considered synonymous, whereas one with low semantic similarity may be meaningfully different and/or contextually very distinct. In this work, we concentrate on the semantic similarity of attributes (i.e., variables in a causal DAG). This is evaluated by determining the semantic similarity between their names. We can also consider alternative ways to assess semantic similarity, such as by taking into account the attribute values as well. Semantic similarity between attribute names can be measured using embedding techniques [31], ontological relationships [18], or large language models [1].

We define a semantic constraint using a semantic similarity measure $sim(X, Y)$, which assigns a value in $[0, 1]$ to a pair of variables (i.e., nodes in a causal DAG). This will ensure that the resulting quotient causal DAG combines only attributes that are semantically related.

Recall that a quotient causal DAG defines a partition of the nodes into disjoint clusters of nodes. A semantic constraint is used to set a threshold on the semantic similarity within each cluster, referred to as the *inter-cluster semantic similarity*. Our framework is capable of supporting inter-cluster semantic similarity defined in any of the following ways: (a) between the closest nodes within a cluster, (b) between the farthest nodes within a cluster, or (c) as the average similarity between all the nodes in the cluster. Given a semantic similarity measure $sim$ and a cluster (i.e., a set) of nodes $\boldsymbol{C}$, let $InterSim(\boldsymbol{C})$ denote its inter-cluster semantic similarity.

Given a semantic similarity measure $sim$, a quotient causal DAG $\mathcal{G}$ and a threshold $\tau$, we say that $\mathcal{G}$ *satisfies the semantic constraint* if for every node $\boldsymbol{C}$ in $\mathcal{G}$ (representing a cluster of nodes), $InterSim(\boldsymbol{C}) \geq \tau$.

**Example 3.** Continuing with our example, consider the following semantic similarity values: $sim($ `humidity, temperature`$) = 0.9$, $sim($`precipitations, temperature`$) = 0.8$, $sim($`precipitations, humidity`$) = 0.85$, and $sim($`airport, temperature`$) = 0.6$. Here we define the inter-cluster semantic similarity as the minimum pairwise semantic similarity within the cluster.

Given a semantic threshold of $\tau = 0.8$, consider the cluster $\boldsymbol{C} = \{$`precipitations, temperature,`

`humidity`}. In this case, this cluster satisfies the semantic constraint as $InterSim(\boldsymbol{C}) = 0.8 \geq \tau$. By adding the node `airport` to $\boldsymbol{C}$, the updated inter-cluster semantic similarity becomes $InterSim(\boldsymbol{C}) = 0.6$, which is less than the semantic threshold $\tau = 0.8$. Thus, including `airport` in $\boldsymbol{C}$ would violate the semantic constraint.

## 3.4 Problem Definition

Preventing degradation of causal query results on a quotient causal DAG requires minimal loss of conditional independence information. However, unbridled optimization according to that success metric can easily lead to two failure modes:

1. *Output DAG is identical to the input graph*: All CIs relationships between variables are retained in this scenario, but this means we did not achieve any graph summarization. We, therefore, include a size constraint to discourage providing a trivial or unsatisfactory output.

2. *Output DAG cannot be inspected, interpreted, or verified*: The system may provide an output DAG that analysts can use to answer causal queries. However, they may struggle to confirm if the quotient causal DAG still retains its *causal interpretation*: namely, whether the graph contains the existence of functional contributions between variables that relate to real-world causal mechanisms. Therefore, we include a semantic constraint to ensure that only nodes of similar semantic meaning are clustered together.

We, therefore, formulate our problem as an optimization problem over maximal retention of causally-relevant information with size and semantic constraints:

**Problem 1** (Causal DAG Summarization)**.** Given a causal DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$, an exposure variable $T$ and outcome variable $O$, a bound $3 \leq k < |V|$, a semantic similarity measure $sim(\cdot, \cdot)$, and a threshold $\tau$, the goal is to generate a quotient causal DAG $\mathcal{G}_P$ such that:

- **Size Constraint**: the number of nodes in $\mathcal{G}_P$ is $\leq k$, and $\mathcal{G}_P$ maintains $T$ and $O$ in separated single-node clusters.

- **Semantic Constraint** the inter-cluster semantic similarity of every cluster of nodes $\boldsymbol{C}$ ($InterSim(\boldsymbol{C})$) in $\mathcal{G}_P$ is $\geq \tau$
- **Objective**: the number of CIs statements present and *entailed* by $C$-separation in $\mathcal{G}_P$ is maximized.

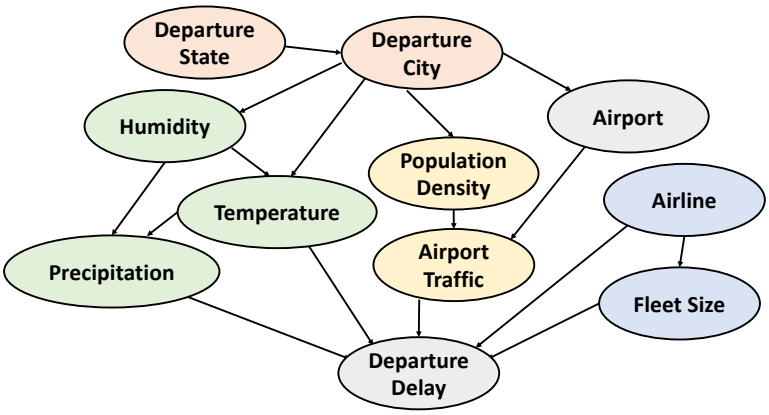


Figure 3-1: Causal DAG used in Example 1, with semantic similarity groups annotated by color.

**Example 4.** Consider again the causal DAG in Figure 3-1, this time with colored nodes. Here, $T$ is set to be `airport` and $O$ is `departure delay`. The node colors indicate semantic similarity, where only nodes of the same color can be grouped. We set $k = 7$ as the desired number of nodes in the quotient causal DAG and $\tau = 1$ as the inter-cluster semantic similarity threshold. Our objective is to find a solution that maximizes the number of (explicit and implied) CIs while satisfying the size and semantic constraints.

The original causal DAG encodes 103353 CI statements. Considering our problem definition, the optimal solution that adheres to the semantic and size constraints is one which separates `airline` and `fleet size`. In this optimal solution, the number of CIs statements is 49515. It is worth noting that there are other solutions,

such as choosing to separate `precipitation` from `humidity` and `temperature`, that satisfy the size and semantic constraints. However, these alternative solutions have a suboptimal number of recoverable CIs statements (39497). The optimal solution for our optimization metric, without regard for the constraints, results in 73973 CIs statements; in this quotient graph, the `departure state` and `departure city` are in separate clusters, and `airport` is merged with `airport traffic`. This is suboptimal due to the presence of the functional dependency between state and city as well as the merging of the treatment attribute with other attributes in the graph.

We believe this problem is $NP$-hard, similar to many graph partitioning problems [25], as we can establish a reduction from the max clique problem to this problem. Future work can formalize this connection to determine the complexity characteristics of our problem.

## 3.5 Objective Evaluation

To evaluate the objective of maximizing the number of unique conditional independence statements present and *entailed* by $C$-separation, we must be able to identify which of two quotient causal DAGs $\mathcal{G}, \mathcal{G}'$ encodes more conditional independence statements. A naive general approach may involve naive enumeration (and deduplication) of all conditional independence statements through $d$-separation and $C$-separation; however, the maximum number of such statements, which is equal to the *Stirling partition number* [13] $S(n, k)$ where $k = 4$ and $n = |\boldsymbol{V}|$, is exponential with respect to the number of nodes. Furthermore, applying $C$-separation to further enumerate conditional independence statements would, for each $d$-separation statement, incur at least an exponential time complexity relative to the size of the clustered nodes in each statement. Such enumeration can easily introduce duplicate conditional independence statements; any unaccounted-for duplicate statements that imply further statements could vastly inflate the metric exponentially beyond its true value.

Therefore, we propose a novel method to approximate the size of the set of con-

ditional independence statements which hold for a given quotient causal DAG as a means of comparison. This method is able to deliver both a sound lower bound and an estimated upper bound which empirically (See Section 5.2) provides meaningful hints of which graph holds more causally relevant information.

As our estimation metric is purely used as a means of comparison between two quotient causal DAGs, we focus on counting conditional independence statements where the separated sets contain only one variable. Namely, we focus on counting CIs of the form $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} | \boldsymbol{Z}$ where $|\boldsymbol{X}| = 1$ and $|\boldsymbol{Y}| = 1$, and $|\boldsymbol{Z}|$ is unrestricted. Via the composition axiom, all CIs statements involving multiple attributes in each separated set can be reconstructed from conditional independence statements involving single, separated attributes. Since the symmetry axiom does not contribute to a meaningful difference in the outcome of the comparator, we do not explicitly consider conditional independence statements attained due to the symmetry axiom. In Section 5.2, we experimentally demonstrate that the size of this set of CIs we count is strongly correlated with the overall size of the CIs set.

Our estimation mechanism is similar for both upper and lower bounds: we transform the summarized graph into an undirected graph; for each pair of nodes in the graph, we use the size of the minimal cut to estimate the number of conditional independence statements which hold between causal attributes within the two nodes. Depending on the transformation we choose, we obtain the desired lower or upper bound. We include Algorithm 1 as an illustrative example for both algorithms, as the upper bound analogue is virtually identical except for a few differences.

As our estimation metric is purely used as a means of comparison between given quotient causal DAGs, we focus on counting conditional independence statements where the separated sets contain only one causal attribute. Via the composition axiom, all conditional independence statements involving multiple attributes in each separated set can be reconstructed from conditional independence statements involving single separated causal attributes. Since the symmetry axiom does not contribute to a meaningful difference in the outcome of the comparator, we do not explicitly consider conditional independence statements attained due to the symmetry axiom.

**Algorithm 1:** Lower Bound CI Statement Counting Algorithm

**Input :** A quotient causal DAG $\mathcal{G}$
**Output:** A lower bound on the number of singleton conditional independence statements of $\mathcal{G}$.

```
lb ← 0.
ug ← MORALIZE(G).                          // Get the moral graph of G
for (U, V) ∈ G.nodes do
    min_cut ← FINDMINCUT (ug, U, V)
    exp_factor ← |G.nodes| − |min_cut| − 2
    count ← |U| * |V| * 2^exp_factor
    lb ← lb + count
return lb
```

### 3.5.1 Lower Bound

To obtain the lower bound, we transform the summarized graph via *moralization* [8], where each parent of the same node gains undirected edges to all other parents of the same node, in addition to converting all existing directed edges into undirected edges.

In this scenario, all conditional independence relationships along paths without colliders are retained, but those involving colliders are lost. Therefore, the transformation is sound—no spurious conditional independence statements are introduced—but not complete—conditional independence statements that hold in the graph are lost.

Then, for each pair of nodes $U, V$ in the graph $\mathcal{G}$, we find a minimum vertex cut $C(U, V)$. Now, we've identified one conditional independence statement where these two nodes in the graph are conditionally independent given the vertex cut set: $U \perp\!\!\!\perp V | C(U, V)$. All other nodes in the graph, $\boldsymbol{V}_{I(U,V)} = \boldsymbol{V} \setminus (C(U, V) \cup \{u, v\})$, are considered irrelevant. This means that more conditional independence statements can be expressed as $\forall I \subset \boldsymbol{V}_{I(U,V)}. U \perp\!\!\!\perp V | C(U, V) \cup I$, where a unique conditioning set can be derived from the original conditional independence statement by including some subset of irrelevant nodes. The number of possible subsets of the irrelevant nodes is $2^{|\boldsymbol{V}_{I(U,V)}|}$; since each subset can emit one unique conditioning set, we find that number to also be the number of conditional independence statements retrievable via $d$-separation in the given graph $G$. In this manner, we are able to count these

conditional independence statements without explicitly enumerating them, simply by knowing the number of irrelevant nodes.

In order to account for statements retrievable via $C$-separation, we must account for each node in the summarized graph mapping to one or more attributes in the underlying causal graph. Due to the aforementioned singleton constraint, each element in the separated set can be chosen as the singleton that remains in the separated set; the rest of the elements can be arbitrarily included in the conditioning set (due to the Weak Union axiom) or elided (due to the Decomposition axiom). Suppose we wish to count conditional independence statements implied by $\boldsymbol{U} \perp\!\!\!\perp \boldsymbol{V} | \boldsymbol{Z}$ attained through $d$-separation above. We can select any $U \in \boldsymbol{U}$, which leads to $|\boldsymbol{U}|$ possible statements $U \perp\!\!\!\perp \boldsymbol{V} | \boldsymbol{Z}$; without loss of generality, the same can be done for $\boldsymbol{V}$ in order to find that $\boldsymbol{U} \perp\!\!\!\perp \boldsymbol{V} | \boldsymbol{Z}$ can generate $|\boldsymbol{U}| * |\boldsymbol{V}|$ statements of the form $U \perp\!\!\!\perp V | \boldsymbol{Z}$. Since the sets $\boldsymbol{U} \setminus \{U\}$ and $\boldsymbol{V} \setminus \{V\}$ are now also comprised of irrelevant nodes, any subset of $(\boldsymbol{U} \setminus \{U\}) \cup (\boldsymbol{V} \setminus \{V\})$ can be added to the conditioning set, leading to a contribution of $2^{|(\boldsymbol{U} \setminus \{U\}) \cup (\boldsymbol{V} \setminus \{V\})|} = 2^{|\boldsymbol{U}| + |\boldsymbol{V}| - 2}$ for each choice of separated causal attributes. In Algorithm 1 the $|\boldsymbol{U}| + |\boldsymbol{V}|$ cancels out with the $\{U, V\}$ term in $\boldsymbol{V}_{I(U,V)}$ to produce the written exponential factor.

### 3.5.2 Estimated Upper Bound

To obtain the estimated upper bound, the only difference is in line 2 of Algorithm 1: we transform the quotient causal DAG by replacing all directed edges with undirected edges, without adding edges between parents of the same node. In this transformation, we still retain the conditional independence relationships along paths without colliders. Therefore, the transformation is complete—all conditional independence statements that hold in the graph are retained—but not sound—conditional independence statements that do not hold in the graph are introduced.

It should be noted that the upper bound we have obtained is not accurate (and may be lower than the real number of CI statements) as it is based on selecting only one cut between two nodes in the quotient DAG. As the quotient DAG increases in size and diameter, the assumption that there exists just one valid cut to $d$-separate

two arbitrary nodes in the quotient DAG (and treat all other nodes as irrelevant nodes) becomes decreasingly true. We experimentally demonstrate this, and that our method is still effective in determining which quotient causal DAG contains a greater number of CI statements in Section 5.2.

# Chapter 4

# Causality-Aware Summarization Algorithms

Our goal of finding a quotient causal DAG that fulfills the constraints and maximizes the objective is likely $NP$-hard. Optimistically, ifwe used an approximation-preserving reduction, we would adopt any approximation ratios that apply to the max clique problem. Unfortunately, the max clique problem is fixed-parameter intractable, namely that there does not exist any polynomial time algorithm which solves the problem where the size of the clique is an input to the problem. **(author?)** [6] adds that this problem cannot be solved in subexponential time unless the exponential time hypothesis does not hold. All this to say, the max clique problem's solution space does not afford us any off-the-shelf performant graph summarization algorithms that we can easily adapt for the causal graph summarization problem.

We hypothesize that causality-aware summarization algorithms are able to better maximize our objective while holding to the constraints than extant graph summarization algorithms. To investigate this hypothesis, we explore a few causality-aware summarization algorithms which fulfill our graph summarization goal with various performance outcomes.

To begin, we discuss the limitations of a naive brute force algorithm and gradually introduce our proposed optimized algorithm. Our optimizations include:

- A greedy approach to identify promising candidate node pairs for grouping,
- An inexpensive update method to maintain a set of CIs,
- A top-down partitioning approach to identify sufficiently small subgraphs for the greedy procedure to handle, and
- A method for obtaining an approximation of the CIs set, which may be considerably large.

To simplify the presentation, we assume in this section that there is no semantic constraint, meaning that every node merge is possible. However, to support a semantic constraint, the algorithm checks whether each merge does not violate the constraint before proceeding.

## 4.1 Brute Force Search

In this algorithm, we exhaustively explore all possible quotient causal DAGs, disregard any which do not adhere to our constraints, and then for each remaining candidate, enumerate the conditional independence relationships through $d$-separation and $C$-separation.

In Algorithm 2, the brute force enumeration of conditional independence statements via $d$-separation involves enumerating all possible $d$-separation statements from the nodes in the graph and testing, in linear time, if that statement holds in the given graph.

The subsequent $C$-separation subroutine (presented in Algorithm 3) is a succinct fix-point algorithm, inspired by the Chase algorithm [9]: given a conditional independence statement set, the algorithm finds all conditional independence statements that the input graphoid implies, adds these new statements to a new input graphoid, and re-runs on this new input graphoid; this algorithm continues to run until the fixpoint, when the input graphoid is equivalent to the output, indicating that all recoverable statements from the initial graphoid have been found.

**Algorithm 2:** Brute Force Search Algorithm, `brute_force_search`

**Input :** A causal DAG $\mathcal{G}$, a node size upper bound $k$
**Output:** A quotient causal DAG $\mathcal{G}_P$

$\mathcal{G}_{\text{output}} \leftarrow \mathcal{G}$
$\mathcal{CI}_{\text{output}} \leftarrow \emptyset$
**forall** *$k$-sized partitions $P$ of $\mathcal{G}$* **do**
  $\mathcal{G}_P \leftarrow \mathcal{G}/P$
  **if** *$\mathcal{G}_P$ contains cycles* **then**
    **continue**
  $\mathcal{CI}_d \leftarrow$ `find_cis_d_sep`$(\mathcal{G}_P)$
  $\mathcal{CI}_c \leftarrow$ `find_cis_c_sep`$(\mathcal{CI}_d)$
  **if** $|\mathcal{CI}_{output}| < |\mathcal{CI}_c|$ **then**
    $\mathcal{G}_{\text{output}} \leftarrow \mathcal{G}_P$
    $\mathcal{CI}_{\text{output}} \leftarrow \mathcal{CI}_c$
**return** $\mathcal{G}_P$

**Algorithm 3:** $C$-separation Fixpoint Algorithm, `find_cis_c_sep`

**Input :** A set of conditional independence statements $\mathcal{CI}$
**Output:** A set of conditional independence statements which include the statements implied by the input CI set $\mathcal{CI}^{imp}$.

$\mathcal{CI}_{new} \leftarrow \emptyset$
**forall** $CI \in \mathcal{CI}$ **do**
  add statements to $\mathcal{CI}_{new}$ according to symmetry, decomposition, and weak union axioms
**if** $|\mathcal{CI}_{new}| = 0$ **then**
  **return** $\mathcal{CI}$
**else**
  **return** *find_cis_c_sep*$(\mathcal{CI} \cup \mathcal{CI}_{new})$

This Brute Force Search (Algorithm 2) is a globally optimal algorithm but runs in exponential time ($O(e^n)$), due to the exponential number of partitions to explore and the exponential number of conditional independence statements to be enumerated for each partition's quotient causal DAG.

## 4.2 Greedy Search

Greedy search algorithms for graph summaries take on two primary approaches: top-down (often called partitioning) and bottom up (often called clustering). In our greedy search, we take an iterative, bottom-up approach: for each step of the algorithm, we select two nodes to merge into a single node; we stop when the desired summary size is reached.

**Algorithm 4:** Greedy Search Algorithm, `greedy_search`

**Input :** A causal DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$, a node size upper bound $k$
**Output:** A quotient causal DAG $\mathcal{G}_P$

$\mathcal{G}_{\text{output}} \leftarrow \mathcal{G}$
$\mathcal{CI}_{\text{output}} \leftarrow$ `find_cis_d_sep`$(\mathcal{G})$
**while** $|\mathcal{G}_{output}| > k$ **do**
  **forall** $(U, V) \in \mathcal{V}_{output}$ **do**
    $\mathcal{G}_P \leftarrow \mathcal{G}_{\text{output}}.\text{contract}(U, V)$
    **if** $\mathcal{G}_P$ *contains cycles* **then**
      **continue**
    $\mathcal{CI}_P \leftarrow$ `update_cis_given_merge`$(\mathcal{CI}_{\text{output}}, (U, V), \mathcal{G}_P)$ **if** $|\mathcal{CI}_{output}| < |\mathcal{CI}_P|$ **then**
      $\mathcal{G}_{\text{output}} \leftarrow \mathcal{G}_P$
      $\mathcal{CI}_{\text{output}} \leftarrow \mathcal{CI}_P$
**return** $\mathcal{G}_P$

Our greedy form of the brute force search improves on the exhaustive search by reducing the space of possible partitions explored during the search; on each choice of a pair of nodes to merge, we narrow the partition search space to all partitions which place both nodes in the same partition.

Furthermore, the graphoid of each quotient causal DAG no longer needs to be

**Algorithm 5:** Merge-Aware Graphoid Update, `update_cis_with_merge`

**Input :** A set of conditional independence statements $\mathcal{CI}$, a pair of nodes to merge $(U, V)$, a quotient causal DAG $\mathcal{G}_P$

**Output:** A subset of $\mathcal{CI}$ which holds in $\mathcal{G}_P$ given the merge between $(U, V)$

$\mathcal{CI}_{\text{keep}} \leftarrow \emptyset$
$\mathcal{CI}_{\text{check}} \leftarrow \emptyset$
**forall** *CI statements of the form* $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} | \boldsymbol{Z}) \in \mathcal{CI}$ **do**
  **if** *remove_condition* **then**
    **continue**
  **else if** *keep_condition* **then**
    $\mathcal{CI}_{\text{keep}} \xleftarrow{+} CI$
  **else if** *second_pass_condition* **then**
    $\mathcal{CI}_{\text{check}} \xleftarrow{+} CI$
  **else if** *check_graph_condition* **then**
    $\boldsymbol{Z}' \leftarrow Z$ but replace $U, V$ with the merged $C_{U,V}$ **if** $\boldsymbol{X}$ *d-separates* $\boldsymbol{Y}$ *given* $\boldsymbol{Z}$ *in* $\mathcal{G}_P$ **then**
      $\mathcal{CI}_{\text{keep}} \xleftarrow{+} CI$
  **forall** $CI \in \mathcal{CI}_{check}$ **do**
    **if** *second_pass_keep_condition* **then**
      $\mathcal{CI}_{\text{keep}} \xleftarrow{+} CI$
**return** $\mathcal{CI}_{keep}$

enumerated from scratch as in the Brute Force Search (Algorithm 2); in our Greedy Search (Algorithm 4), we update the graphoid of the original causal DAG based on the local merge in linear time with respect to the input graphoid. We are able to accomplish this by recognizing that the vertex contraction operation is a sound (but not complete) transformation for conditional independence relation preservation as illustrated in Proposition 1, so we use $d$-separation and $C$-separation to determine which conditional independence relations would be lost, given a particular contraction operation between two designated vertices.

Algorithm 5 describes five boolean conditions which determine subsequent behavior of the Merge-Aware Graphoid Update subroutine, illustrated in Table 4.1. The notation $[A]$ refers to some nodeset which contains $A$; namely, $A \in [A]$; similarly, $[AB]$ refers to a nodeset containing both $A$ and $B$, as in $A, B \in [AB]$.

Since both exponential time bottlenecks in the Brute Force Search (Algorithm 2)

| Condition | Qualifying CI Statements (output `TRUE` if holds) |
|---|---|
| `remove_condition` | $[A] \perp\!\!\!\perp [B] \mid \boldsymbol{Z}$ |
| `keep_condition` | $[AB] \perp\!\!\!\perp \boldsymbol{Y} \mid \boldsymbol{Z}$ |
| `second_pass_condition` | $[A] \perp\!\!\!\perp \boldsymbol{Y} \mid \boldsymbol{Z}$<br>or $[A] \perp\!\!\!\perp \boldsymbol{Y} \mid [B]$<br>or $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} \mid [A]$ |
| `check_graph_condition` | $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} \mid \boldsymbol{Z}$ where $A, B \notin \boldsymbol{X} \cup \boldsymbol{Y} \cup \boldsymbol{Z}$<br>or $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} \mid [AB]$ |
| `second_pass_keep_condition` | $[A] \perp\!\!\!\perp \boldsymbol{Y} \mid \boldsymbol{Z}$ if $[AB] \perp\!\!\!\perp \boldsymbol{Y} \mid \boldsymbol{Z}$<br>$[A] \perp\!\!\!\perp \boldsymbol{Y} \mid [B]$ if $[AB] \perp\!\!\!\perp \boldsymbol{Y} \mid \boldsymbol{Z}$<br>$\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} \mid [A]$ if $[AB] \perp\!\!\!\perp \boldsymbol{Y} \mid \boldsymbol{Z}$ |

Table 4.1: Conditions used in the Merge-Aware Graphoid Update in Algorithm 5

are unblocked in this Greedy Search (Algorithm 4), the algorithm should return in polynomial time with respect to the size of the graphoid in the original graph.

However, the size of the original graph's graphoid is exponential in size relative to the number of nodes in the graph. Despite the algorithmic improvements in partition enumeration and evaluating the sizes of graphoids, this Greedy Search is still not fast enough for our task.

## 4.3 Causal Acyclic Markov-Boundary Algorithm

We introduce our Causal Acyclic Markov-Boundary Algorithm (CAMBA) as an example which improves upon the naive and greedy summarization algorithms using insights about counting conditional independence statements to cheaply make partitioning and clustering decisions. We recognize that using the Markov boundary is one way to isolate one portion of the graph from another—thereby being able to use it both as a means to partition the graph and cheaply approximate the graph's graphoid. To overcome the exponential scale of the original graph's graphoid, we add two optimizations based on the *Markov boundary* [41]: 1) we include a top-down partitioning approach to identify sufficiently small subgraphs for the Greedy Search to handle, and 2) we improve the initial graphoid initialization in the first step of Greedy Search by using conditional independence statements derived from each nodes' Markov boundary. Furthermore, we add a heuristic initial node-merging pass to capture some

**Algorithm 6:** The CAMBA Algorithm

**input :** A causal DAG $\mathcal{G}$, and a number $k$.
**output:** A quotient causal DAG $\mathcal{G}'$.

$\mathcal{G}' \leftarrow$ HeuristicMerges$(\mathcal{G}, k)$
**if** $|\mathcal{G}'.nodes| \leq k$ **then**
  return $\mathcal{G}'$
$\mathcal{G}' \leftarrow$ MarkovBoundaryPartitioning$(\mathcal{G}', k$, threshold_size=greedy_max$)$
**return** $\mathcal{G}'$

high-value low-cost choices before entering the partitioning phase of the algorithm.

Since the heuristic algorithm takes $O(n^2)$, and the partitioning takes $O(nd * log(n))$, and the greedy takes $O(md)$ plus the fixpoint algorithm $O(e^m)$, we have $O(n^2 + nd * log(n) + mde^m)$ where $m$ is the internally-set maximum size of a subgraph where the partitioning algorithm stops and the greedy algorithm runs.

### 4.3.1 Top-Down Markov Boundary Partitioning

To support the goal of preserving as many conditional independence statements as possible, we also introduce a top-down component of the algorithm that greedily finds bisections of the graph that preserve the most conditional independence statements. In such an algorithm, the secret sauce is the guiding metric—how we decide, among exponentially many cuts, which would preserve the most conditional independence statements if taken.

One way to approximately find our metric-maximizing split at each step of our algorithm is to find a single representative conditional independence statement which implies the most conditional independence relationships. That way, if we partition the graph according to the 'densest' representative conditional independence statement, we assert that all conditional independence relationships that cross that partition would be retained in the output summary graph.

In light of avoiding the exponentially-large spaces of cut enumeration and conditional independence statement evaluation, we identify such representative conditional independence statements by identifying the *Markov boundary* [41] for each node in the graph. The Markov boundary of a node $V$, denoted as $MB(V)$, includes its

**Algorithm 7:** Markov Boundary Graphoid Approximation, `get_mb_cis`

**Input :** A graph $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$
**Output:** A partial set of conditional independence statements $\mathcal{CI}$

$\mathcal{CI} \leftarrow \emptyset$
**forall** $V \in \boldsymbol{V}$ **do**
  $\boldsymbol{P} \leftarrow \{U | (U, V) \in \boldsymbol{E}\}$
  $\boldsymbol{C} \leftarrow \{U | (V, U) \in \boldsymbol{E}\}$
  $\boldsymbol{S} \leftarrow \{U | \forall N \in \boldsymbol{C}.(U, N) \in \boldsymbol{E} - \{V\}\}$
  $\boldsymbol{MB} \leftarrow \boldsymbol{P} \cup \boldsymbol{C} \cup \boldsymbol{S}$
  $\mathcal{CI} \overset{+}{\leftarrow} (V, \boldsymbol{V} - \boldsymbol{MB} - \{V\}, \boldsymbol{MB})$
**return** $\mathcal{CI}$

parents, children, and the other parents of all of its children. It is the minimal set of nodes required to separate $V$ from the rest of the nodes in the DAG $\mathcal{V}$. Each node $V \in \mathcal{V}$ and its Markov boundary $MB(V)$ can produce the following conditional independence statement:

$$V \perp\!\!\!\perp \mathcal{V} \setminus \{V\} \setminus MB(V) | MB(V)$$

this is illustrated in Algorithm 7.

Based on $C$-separation, each of those statements could be broken down into further conditional independence statements as discussed in Section 3.5. However, we can avoid explicitly counting such statements by approximating the 'densest' conditional independence statement by picking one with the largest $|V - n - MB(n)|$, which is the one with the smallest Markov boundary.

Since this partitioning algorithm does not guarantee acyclicity out-of-the-box, we randomly perturb the partition borders until we meet our acyclicity goal.

### 4.3.2 Markov Boundary Graphoid Initialization

We also improve our graphoid initialization of the input graph in our greedy search by enumerating two conditional independence statements from each node as a seed: one relating the node to the rest of the graph and one relating the node to its ancestors. This takes the naive $O(e^n)$ enumeration of all $d$-separation statements out of the

graphoid initialization step and replaces it with two components: (1) a polynomial-time initialization, and (2) a modified form of the $C$-separation fixpoint algorithm to include the intersection axiom and capture $d$-separation as a whole.

### 4.3.3 Merge Heuristics

Finally, as a pre-pruning step, we merge nodes with the same children and parents and nodes connected along an isolated path or string of nodes.

Merging nodes with the same children and parents minimize changes made to any conditional independence relationships for nodes in the overall graph, since the only conditional independence statements which are lost are those among the merged nodes (which would happen anyway between any arbitrary merged nodes). Due to this trait, of all possible merges, often, these are the merges which result in the least total loss in conditional independence statements.

Merging nodes along a non-branching path in a graph, similarly, does not change conditional independence relationships between nodes in the overall graph, since all paths that run through the merged nodes are still intact, and no new paths are introduced.

**Algorithm 8:** Markov Boundary Partitioning Algorithm

**input :** A causal DAG $\mathcal{G}$, a number $k$, and a size threshold $\tau$.
**output:** A quotient causal DAG $\mathcal{G}'$.

```
indivisibles ← ∅
pqueue← 𝒢
while /pqueue/ > 0 and k </pqueue/ + /indivisibles/ do
    subgraph ← pqueue.pop()
    if subgraph is a complete graph or /subgraph/ ≤ tau then
        indivisibles.put(subgraph)
        continue
    sgMB ← subgraph.GetMBCIs (subgraph)
    XZComponent ← subgraph with the smallest Markov boundary and its
     node
    YComponents ← weakly connected components in subgraph \
     XZComponent
    partitions ← XZcomponent + YComponents
    testAcyclicityGraph ← subgraph/partitions
    while testAcyclicityGraph.HasCycles do
        frontierXZ ← XZComponent.nodes with an edge connecting to node out
         of the component
        frontierY ← YComponents.nodes with an edge connecting to a node in
         XZComponent
        chosenNode ← random(frontierXZ ∪ frontierY)
        if chosenNode ∈ XZComponent then
            move chosenNode to a random YComponent it has an edge to
        else
            move chosenNode to XZComponent
            partitions ← XZComponent ∪ YComponents
            testAcyclicityGraph ← subgraph / partitions
    pqueue.put(partitions)
partition ← pqueue + indivisibles
𝒢′ ← 𝒢/partition
return 𝒢′
```

**Algorithm 9:** Faster CI Set Initialization

**input :** A causal DAG $\mathcal{G}$.
**output:** A set of CIs $\mathcal{CI}$.

$\mathcal{CI} \leftarrow \emptyset$
**while** $|\mathcal{G}.nodes()| > 0$ **do**
  $\mathcal{CI}.add(\texttt{GETMBCIs}(\mathcal{G}))$
  leaves $\leftarrow$ nodes in $\mathcal{G}$ with no outbound edges
  $\mathcal{G}.remove$(leaves)
**return** FINDCISCSEP ($\mathcal{CI}$)

**Algorithm 10:** Heuristic Pre-Pruning, `heuristic_merges`

**Input :** A causal DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$, a node size upper bound $k$
**Output:** A quotient causal DAG $\mathcal{G}_{\text{output}}$

$\mathcal{G}_{\text{output}} \leftarrow \mathcal{G}$
**forall** $(U, V) \in \boldsymbol{V}_{output}$ **do**
  **if** *$U, V$ have the same parents and children* **then**
    $\mathcal{G}_P \leftarrow \mathcal{G}_{\text{output}}.\text{contract}(U, V)$
    **if** *$\mathcal{G}_P$ contains cycles* **then**
      **continue**
    **else if** $|\boldsymbol{V}_P| = k$ **then**
      **return** $\mathcal{G}_P$
    $\mathcal{G}_{\text{output}} \leftarrow \mathcal{G}_P$
**forall** $V \in \boldsymbol{V}_{output}$ **do**
  **if** *$V$ has at most 1 parent and 1 child* **then**
    **if** *$V$'s parent $P$ has at most 1 parent and 1 child* **then**
      $\mathcal{G}_{\text{output}} \leftarrow \mathcal{G}_{\text{output}}.\text{contract}(P, V)$
    **else if** *$V$'s child $C$ has at most 1 parent and 1 child* **then**
      $\mathcal{G}_{\text{output}} \leftarrow \mathcal{G}_{\text{output}}.\text{contract}(V, C)$
  **if** $|\boldsymbol{V}_P| = k$ **then**
    **return** $\mathcal{G}_{output}$
**return** $\mathcal{G}_{output}$

# Chapter 5

# Experimental Evaluation

In this experimental evaluation, we empirically demonstrate the following claims:

1. Our proposed objective evaluation method is highly useful in identifying which quotient causal DAG encodes more CIs.
2. CAMBA performs better than competing methods at causal DAG summarization.
3. CAMBA is able to perform well within a reasonable runtime.

## 5.1 Experimental Setting

All algorithms are implemented in Python 3.7. The experiments were executed on a Macbook with the Apple M1 chip and 16GB RAM.

### 5.1.1 Examined Datasets

We demonstrate our results over two commonly used datasets that are associated with real-life causal DAGs to summarize. The datasets and causal DAG were taken from [62].

1. **Flights** is a relational dataset from the U.S. Bureau of Transportation Statistics which describes the timeliness of, and related attributes of, domestic flights

in the US during 2015. This dataset was enriched with other attributes describing the weather, population, and properties of the airline carriers (such as fleet size and annual revenue). The overall number of nodes and edges in the full causal DAG is 9 nodes and 17 edges.

2. **COVID** is a relational dataset describing COVID-19 infections per country between January and July of 2020. This dataset was enriched with properties describing the population, weather and economies of each country. The overall number of nodes and edges in the full causal DAG is 12 nodes and 23 edges.

### 5.1.2 Competing Methods

We evaluate the performance of our CAMBA algorithm against the following baselines:

- **Brute-Force**: The optimal solution according to the CAUSAL DAG SUMMARIZATION problem (Problem 1). This algorithm implements an exhaustive search over all possible quotient causal DAGs.
- **Greedy-Only** A restricted variant of our CAMBA algorithm that uses only the greedy search procedure.
- **SNAP** [54]: a state-of-the-art grouping-based graph summarization algorithm. We explore the performance of SNAP with and without imposing semantic constraints.

### 5.1.3 Default Configuration

In our implementation, we used a FastText model [32] to generate similarity scores. The semantic threshold is meant to be a tunable parameter; we set $\tau = 0.33$ for both the FLIGHTS and the COVID datasets to strike a balance between the space of possible partitions and semantic validity. For comparison purposes, we set the desired size of the summarized graph to be the same as whatever summary size SNAP was capable of summarizing; otherwise, the objective metric comparison would be impossible to

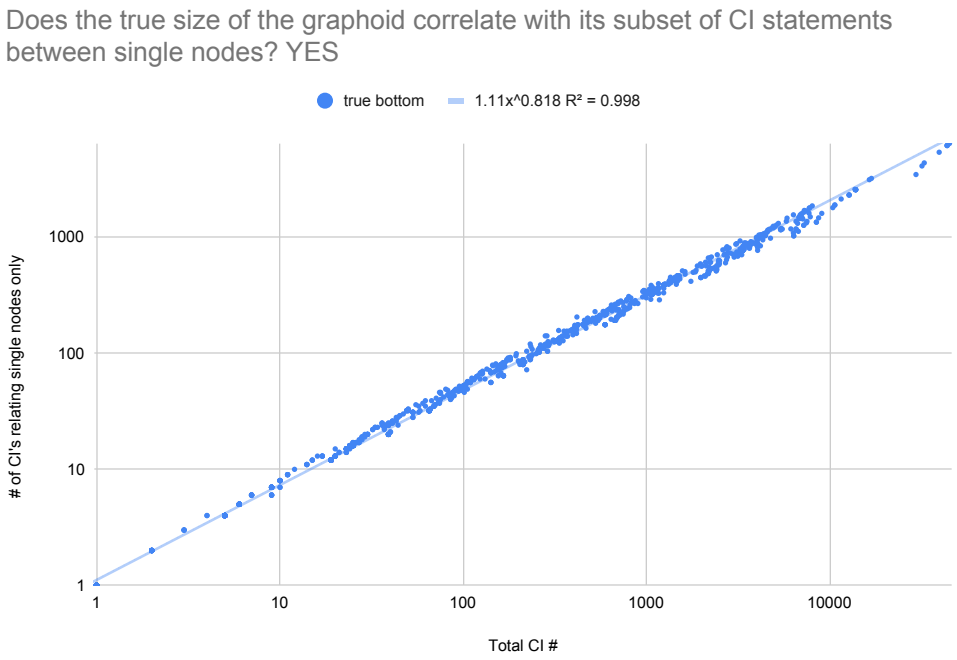


Figure 5-1: The size of the overall CI set *can* be captured by the subset concerning singletons.

do between quotient DAGs of different sizes as the metric scales exponentially with the size of the graph. The runtime cutoff was set at 3 hours.

## 5.2 Evaluating the Objective

Evaluating causal graph summarization algorithms naively and exactly is computationally infeasible, so in Section 3.5, we introduced both lower and estimated upper bounds to conduct a best-case evaluation given realistic computing constraints. Here, before proceeding with the performance evaluation, we investigate how these estimations compare with the true number of CIs.

In this minor foray, we randomly generated 1295 quotient causal DAGs with up to 13 nodes and calculated the true size of the CI statements they encode, and estimations with a 30-second timeout.

First, we ask: "Is it a valid means to approximate the size of the overall size

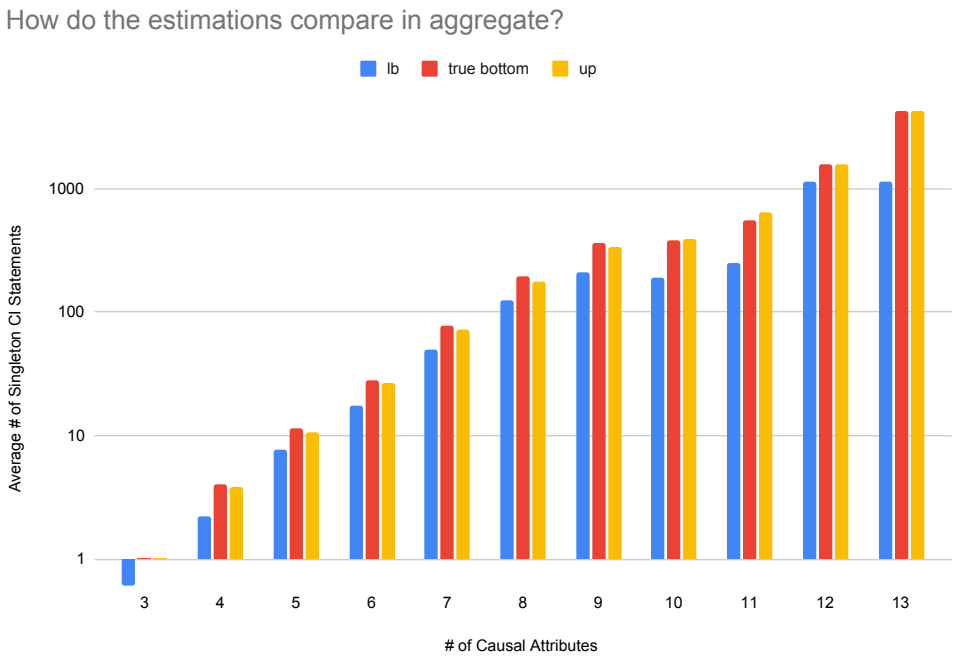


Figure 5-2: Overall, lower bound and upper bound estimations track closely with the true brute-force value.

of CI statements set by only counting conditional independence statements between singletons (except for the conditioning set)?" As illustrated in Figure 5-1, our answer is yes. We found that, in practice, there is a strong correspondence between the graph's entire CI set size and the subset containing only singleton statements. Understanding what causes these subtle differences may be a fruitful question to develop in future work.

From a bird's-eye view, the lower and upper bounds seem very promising in Figure 5-2; of course, the aggregation may portray a smaller skew than actuality on a per-estimation basis, so we also analyzed how each estimation compared to the true value to better understand the skew.

As illustrated in Figure 5-3, the lower bound is sound in practice and is exactly the right value 16.6% of the time. However, 20.6% of the lower bound estimations were 0. In these cases, all CI statements in the quotient graph were lost in the moralization step. This would mean that in such graphs, the entire CI set captures

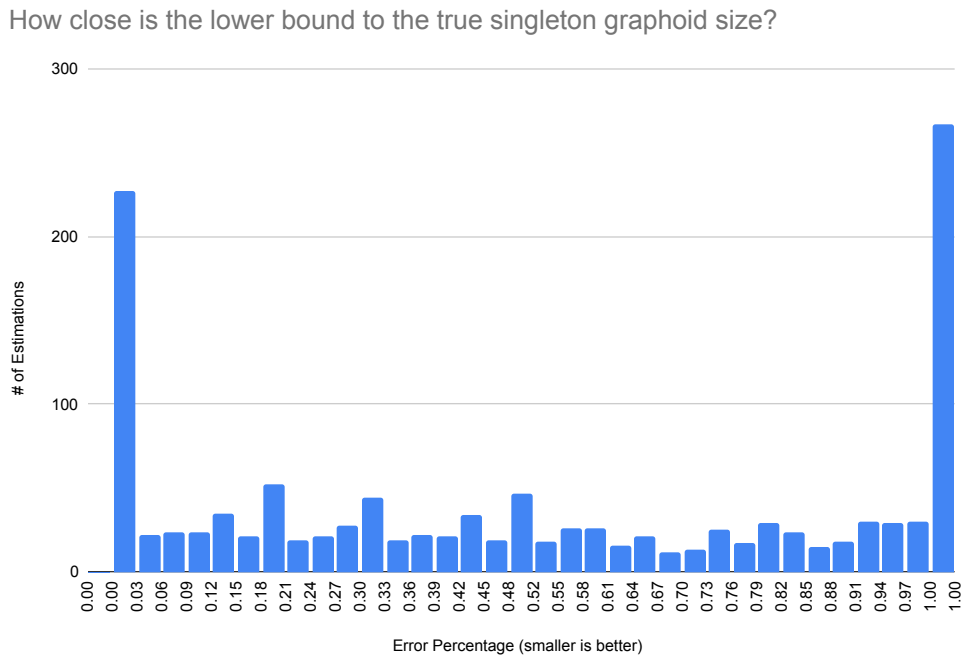


Figure 5-3: Comparing the lower bound estimation with the true size of the CI set of singleton statements, we find a bimodal distribution of error percentages.

independencies that happen along paths of colliders. Beyond the two extremes, it remains the case that the remaining 62.8% of the estimations are roughly uniformly distributed between 0-100% below the true size of the CI set. While this lower bound is theoretically sound, the lower bound estimations are often (at least 20% of the time) not sufficiently informative to determine promising causal graph summarizations.

While the upper bound is not theoretically sound or complete, our results in Figure 5-4 seem to indicate this estimation is exactly correct 39% of the time and within 20% of the true value 71% of the time. Furthermore, 79.4% of the lower-bound estimations of 0 have an exact upper bound estimation, giving us a guideline of having an exact estimation 1/3 of the time, roughly twice that of having only the lower bound.

Lastly, we endeavored to evaluate these estimations as a comparator. To do this, we generated 573 graphs with up to 10 nodes, randomly created two different quotient graphs with the same node count from each synthetic graph, and ran both brute force counting and estimations to identify whether the estimations are effective in deter-

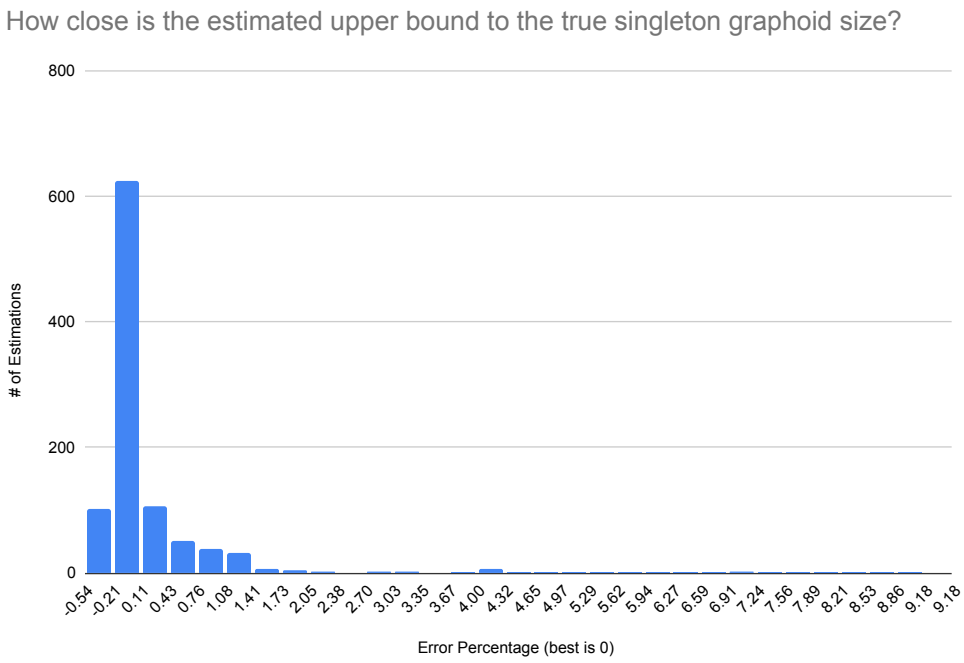


Figure 5-4: Comparing the upper bound estimation with the true size of the CI set of singleton statements, we find a distribution of error percentages that is much more tightly distributed around 0.

mining which graph preserved more CI statements. Again, we used a timeout of 30 seconds for each estimation. We found that simply using the upper bound estimation as the metric of comparison resulted in precision and recall of 94.2% compared to brute force. As seen in Figure 5-5, we see a decrease in accuracy as the size of the quotient causal DAGs gets bigger, likely due to the assumption that having a single minimum cut represent how two nodes are separated is less true as the quotient DAG gets larger.

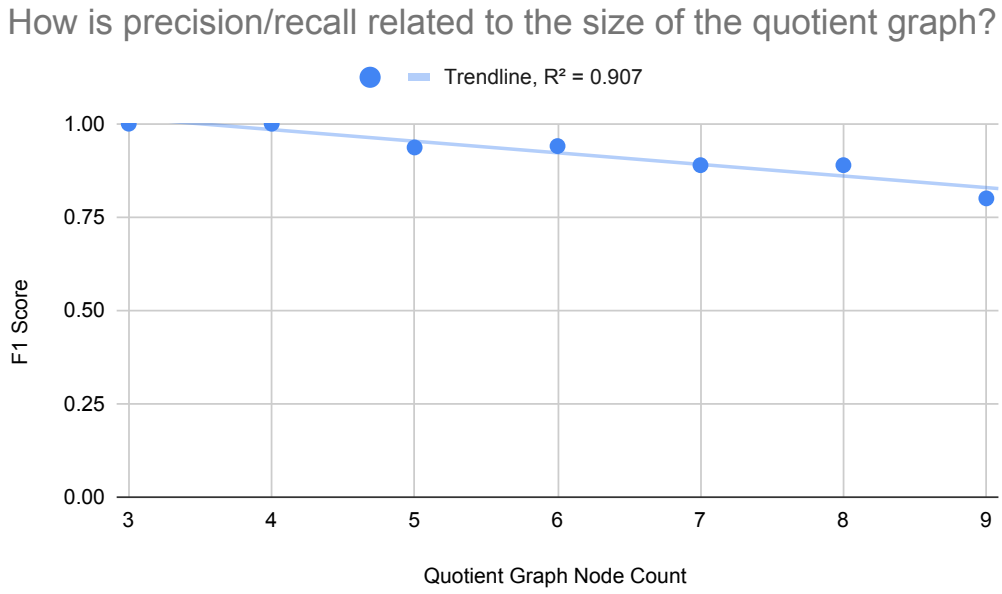


Figure 5-5: The resulting size of the quotient DAG impacts the quality of our upper-bound-based comparator

## 5.3 Evaluating Effectiveness

We observe that CAMBA consistently outperforms competing methods with regard to the number of preserved CI statements. We had to remove the semantic constraint in order for SNAP to summarize the causal DAGs; otherwise, SNAP with the given semantic constraints would refuse to summarize the graph.

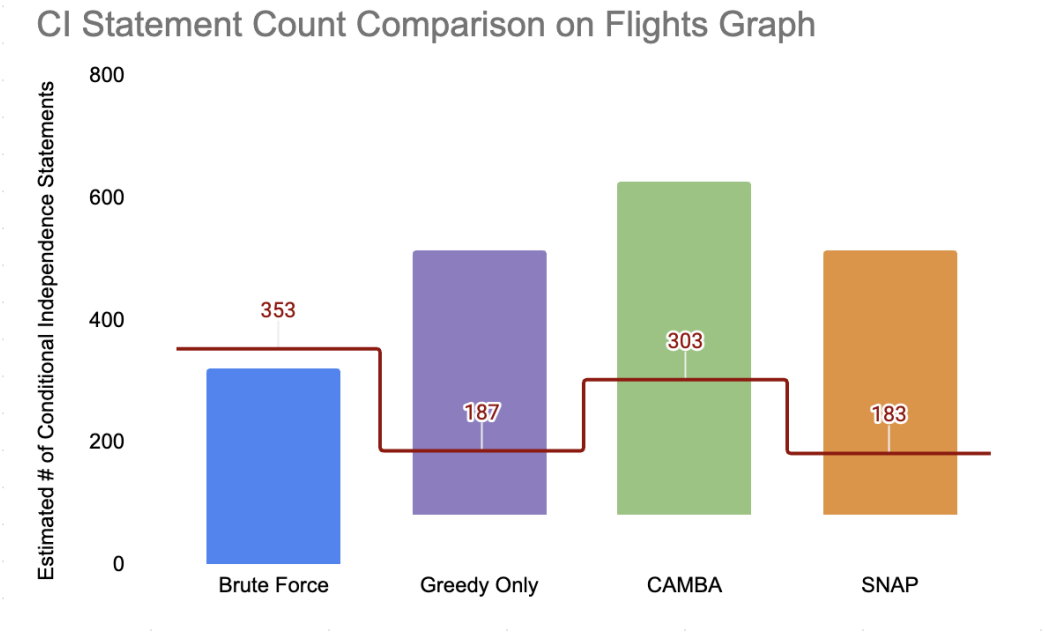


Figure 5-6: Objective Metric Evaluation for Flights Dataset, with the brute force true values annotated and labeled in red.

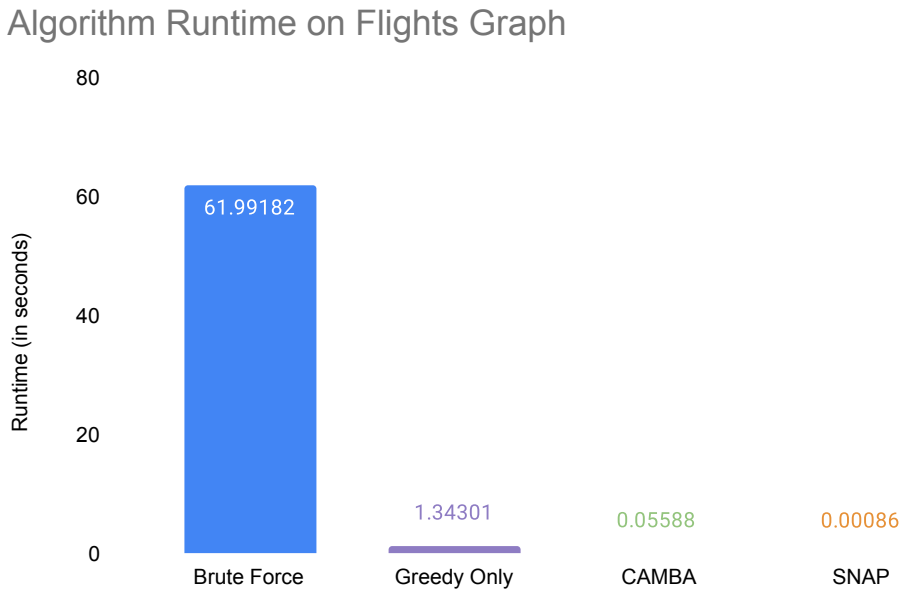


Figure 5-7: Runtime Evaluation for Flights Dataset

The evaluations for the FLIGHTS dataset for all baselines were all completed within the runtime cutoff as seen in Figure 5-7; since the brute force evaluation was also completed within our runtime cutoff, we included the true objective values in addition to the estimated values using the red lines in Figure 5-6. CAMBA found the second-best quotient DAG according to the objective metric alone and identified a better semantic cluster than the brute force approach, clustering together `area` and `geography` rather than `area` and `state`.

In the COVID dataset, both the brute force and greedy algorithms timed out as in Figure 5-9, highlighting the exponentially increasing impact of the causal graph size on the algorithms; among the other two evaluations, the upper and lower bound estimations seem to reflect a clear quality difference in Figure 5-8. In the absence of a computationally infeasible ground truth evaluation, we deem CAMBA to succeed in optimizing the objective metric over SNAP.

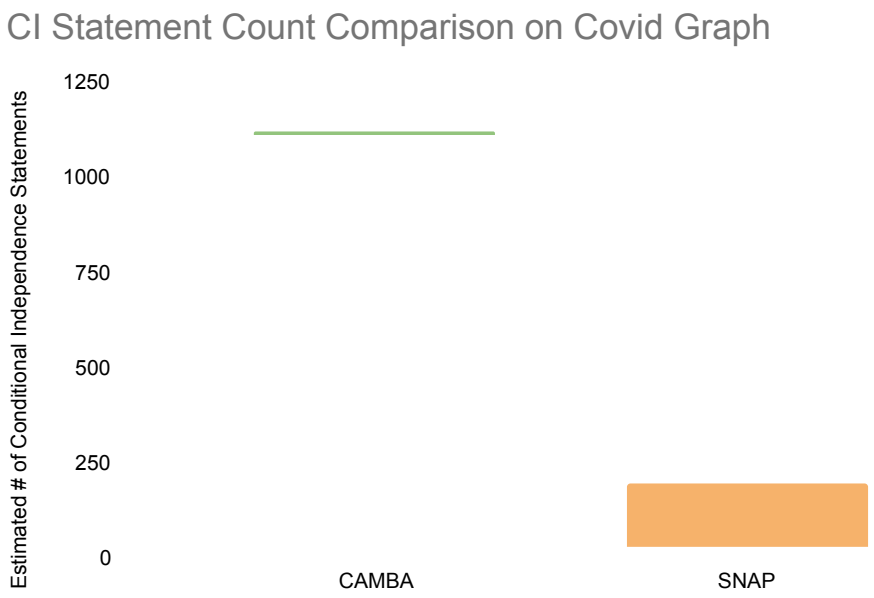


Figure 5-8: Objective Metric Evaluation for Covid Dataset; Brute Force and Greedy Only both timed out after 3 hours in this evaluation.

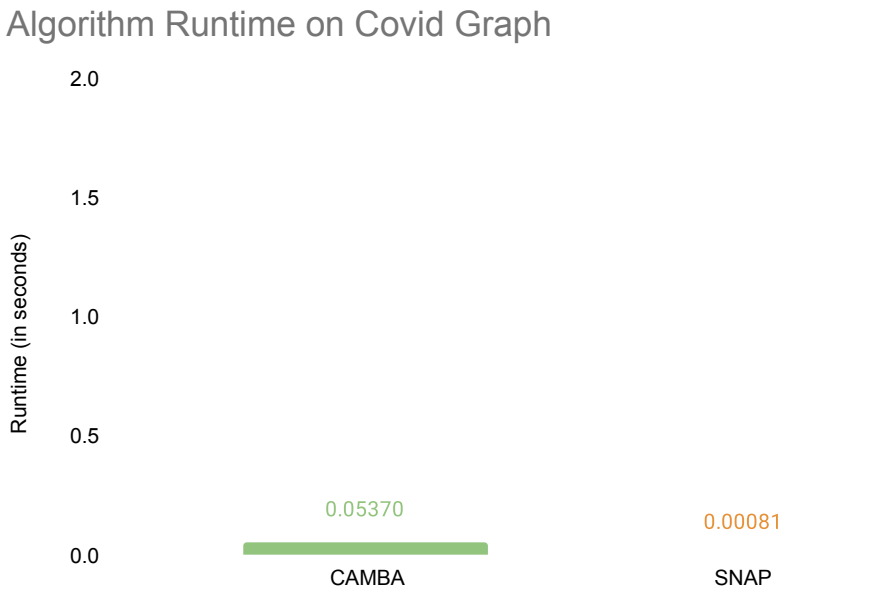


Figure 5-9: Runtime Evaluation for Covid Dataset; Brute Force and Greedy Only both timed out after 3 hours in this evaluation.

# Chapter 6

# Related Work

## 6.1 Graph Summarization

Graph summarization aims to generate a condensed overview of a graph, emphasizing its key characteristics. As a result, summary graphs are typically smaller and more comprehensible. Various graph summarization approaches in the literature utilize fundamental techniques such as grouping, compression, simplification, and influence-based methods [25].

Grouping, often referred to as clustering, is a widely utilized technique in graph summarization [27, 66, 52, 51, 46, 55]. Several node-grouping approaches involve aggregating nodes into "supernodes" based on optimization functions that are application-specific [66, 2]. These optimization functions can be designed considering either structural or semantic similarity [65]. Alternatively, some methods leverage existing clustering techniques to identify supernodes [23]. Moreover, edge grouping methods are also employed, where edges are aggregated into virtual nodes [27]. Our CAMBA algorithm follows this line of work by grouping sets of nodes into supernodes, ensuring that causal relationships in the underlying causal DAG can be recovered from the summarized DAG.

Compression techniques aim to reduce the number of bits required to represent the input graph [34, 28]. The objective is to generate a graph summary that is substantially smaller than the original graph while highlighting different patterns

that contribute to a better understanding of the original graph's structure [28].

Simplification-based summarization methods (e.g., [49, 24]) streamline the original graph by selectively removing nodes or edges considered less "important." This process leads to a sparsified graph, where the summary graph comprises only a subset of the original nodes and/or edges. By simplifying the graph, these methods help reduce complexity while retaining essential structural information for analysis and visualization purposes.

Influence-based methods (e.g.,[30]) aim to uncover concise and high-level representations of influence dynamics in large-scale graphs. The objective is to gain insights into the overall patterns of influence propagation. These summarization techniques have primarily been applied to social graphs, where crucial questions concerning influence dynamics arise.

The primary distinction in our work lies in the goal of summarization. Previous techniques have focused on graph summarization for query handling and efficiency [27, 11, 46]. Additionally, graph summarization has been widely employed for visualization and pattern discovery [49, 16, 20], as well as for extracting influence dynamics [30]. Therefore, we argue that existing techniques are not suitable for the causal DAG summarization problem. The objectives of graph summarization vary across different applications, and as a result, the current methods do not adequately address the specific goal of preserving causal relationships during the summarization process. Through our experimental study, we have shown that SNAP [55], a top-performing grouping-based graph summarization method, is inadequate for the causal DAG summarization task. This method falls short as it neglects the consideration of conditional independence statements during the process of node grouping. As a consequence, it fails to produce summarized causal DAGs that preserve causal relationships.

## 6.2 Data Summarization

Data summarization is the process of extracting meaningful information from a large dataset and representing it in a concise, easily understandable form. It is a crucial

step in data analysis and can help to identify trends, patterns, and correlations in complex data. A related line of work has focused on summarizing large tabular datasets [10, 61, 58, 19, 21, 22]. By summarizing tabular data, analysts can more easily gain insights into the data that would otherwise be difficult to discern. While our goal is different as we aim to summarize causal DAG, its impact is similar as it may help scientists in understanding the causal relationships between variables in high-dimensional datasets.

## 6.3 Causal Discovery

Causal discovery is a well-studied problem [12], whose goal is to infer causal relations among a set of attributes in a given dataset. Though it is well-known that background knowledge is required to determine causal relations [39], a causal DAG can be inferred from the data under some assumptions [12, 7] (e.g., sufficiency, faithfulness). Prominent methods include constraint-based algorithms (e.g., PC, FCI [53]) and score-based algorithms (e.g., LiNGAM [50], GES [7], including ML approaches [59, 67]). Pashami et al. [37] have presented a method that leverages clustering techniques for causal discovery. Their objective is to identify a causal DAG from observational data, rather than summarizing an existing one. They use a cluster-based conflict resolution mechanism whenever the algorithm cannot determine the relationship among variables.

Our objective is to provide a summary of a large input causal DAG that represents causal relationships within a high-dimensional dataset. Consequently, our work serves as a complementary endeavor to the existing research in causal discovery. In our approach, we assume that the input for our algorithm is a causal DAG, which can be automatically generated using one of the aforementioned algorithms. By focusing on summarizing the causal DAG, we aim to distill the essential causal relationships within the dataset while accommodating its complex nature.

## 6.4 Clustered Causal DAGs

The concept of abstracting causal models has recently received some attention in the literature. The authors of [36] study prospects of representing relationships between variable groups (referred to as macro-variables) in Bayesian networks. The authors of [5, 48] investigate the mapping of a cluster of micro-variables to a single macro-variable while preserving certain causal characteristics. Such mappings result in a higher-level structural causal model that retains causal properties similar to the original model at a lower level of abstraction. The aim of these investigations is to create simplified representations of causal models while maintaining key causal properties.

The concept of clustered causal diagrams (C-DAGs) was introduced in [3]. In C-DAGs, causal relationships are depicted among clusters of variables, rather than between individual variables within the clusters. To facilitate causal inference tasks over C-DAGs, the authors extended the *do-calculus* framework [42] for C-DAGs. However, a question that remains unanswered in [3] pertains to the effective learning of C-DAGs from data. Two recent studies have tackled this problem. In [56], a method for clustering an input causal DAG is introduced. However, their transit cluster approach is limited to clustering only mediator variables. Moreover, it disregards the semantic relationships among variables, and the user lacks control over the size of the resulting clustered DAG. In [35], the authors propose a greedy search approach for directly learning a desirable clustered DAG from the data. Their algorithm employs an information-theoretic scoring function that takes into account variable correlations. However, similar to [56], semantic information is not considered, potentially leading to a less interpretable DAG. Additionally, the proposed algorithm does not prioritize the preservation of conditional independence statements among variables. In contrast, our proposed CAMBA algorithm provides analysts with the ability to regulate the size of the summarized causal DAG, and ensures it remains interpretable. Furthermore, CAMBA is optimized to preserve conditional independence statements from the original DAG within the summarized DAG. This ensures that valuable information regarding causal relationships is retained and recoverable, enhancing the

overall reliability and fidelity of the summarized DAG.

# Chapter 7

# Conclusion

Thus, we have illustrated that causality-specific graph summarization algorithms show promise in enabling automatic construction of large causal graphs to be useful in practical settings—our CAMBA algorithm is capable of generating high-quality causal graph summaries that are interpretable and usable for causal inference. With the introduction of this new problem domain, our exploration of the solution space is far from over. CAMBA practically ends up merging a central node together and isolating the less-centrally-connected nodes—ideally, we would split the graph into three partitions to retain conditional independence statements, based on the vertex cut that maximizes the number of nodes that are now conditionally independent from each other on either side of the cut, and then summarize within those three partitions. Furthermore, these (and other!) methods can be extended to mixed graphs, bayesian networks, and other representations of causal and probabilistic relationships between data. In the case of Bayesian networks and other probabilistic graphical models, there are more valid vertex contraction operations available than in our case due to the flexible interpretation of edge directionality and lack of an acyclicity constraint. The theoretical work of demonstrating problem complexity and completeness of $C$-separation is also fertile grounds for future work.

# Bibliography


[1] OpenAI introducing chatgpt. `https://openai.com/blog/chatgpt`. Accessed: 2023-05-08.

[2] Bijaya Adhikari, Yao Zhang, Aditya Bharadwaj, and B Aditya Prakash. Condensing temporal networks using propagation. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 417–425. SIAM, 2017.

[3] Tara V Anand, Adèle H Ribeiro, Jin Tian, and Elias Bareinboim. Effect identification in cluster causal diagrams. *arXiv preprint arXiv:2202.12263*, 2022.

[4] David A Bader, Henning Meyerhenke, Peter Sanders, and Dorothea Wagner. *Graph partitioning and graph clustering*, volume 588. American Mathematical Society Providence, RI, 2013.

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

[6] Jianer Chen, Xiuzhen Huang, Iyad A. Kanj, and Ge Xia. Strong computational lower bounds via parameterized complexity. *Journal of Computer and System Sciences*, 72(8):1346–1367, 2006.

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

[8] Robert G Cowell, Philip Dawid, Steffen L Lauritzen, and David J Spiegelhalter. *Probabilistic networks and expert systems: Exact computational methods for Bayesian networks*. Springer Science & Business Media, 2007.

[9] Alin Deutsch, Alan Nash, and Jeff Remmel. The chase revisited. In *Proceedings of the Twenty-Seventh ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '08, page 149–158, New York, NY, USA, 2008. Association for Computing Machinery.

[10] Kareem El Gebaly, Parag Agrawal, Lukasz Golab, Flip Korn, and Divesh Srivastava. Interpretable and informative explanations of outcomes. *Proceedings of the VLDB Endowment*, 8(1):61–72, 2014.

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

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

[13] Ronald L Graham, Donald E Knuth, Oren Patashnik, and Stanley Liu. Concrete mathematics: a foundation for computer science. *Computers in Physics*, 3(5):106–107, 1989.

[14] Sébastien Harispe, Sylvie Ranwez, Stefan Janaqi, and Jacky Montmain. Semantic similarity from natural language and ontology analysis. *ArXiv*, abs/1704.05295, 2015.

[15] V. Henderson and J.F Thisse. *Handbook of regional and urban economics: cities and geography*, volume 4. Elsevier, 2004.

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

[17] Diviyan Kalainathan, Olivier Goudet, Isabelle Guyon, David Lopez-Paz, and Michèle Sebag. Structural agnostic modeling: Adversarial learning of causal graphs. *arXiv preprint arXiv:1803.04929*, 2018.

[18] Vipul Kashyap and Amit Sheth. Semantic and schematic similarities between database objects: a context-based approach. *The VLDB journal*, 5(4):276–304, 1996.

[19] Alexandra Kim, Laks VS Lakshmanan, and Divesh Srivastava. Summarizing hierarchical multidimensional data. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*, pages 877–888. IEEE, 2020.

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

[21] Laks VS Lakshmanan, Raymond T Ng, Christine Xing Wang, Xiaodong Zhou, and Theodore J Johnson. The generalized mdl approach for summarization. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*, pages 766–777. Elsevier, 2002.

[22] Laks VS Lakshmanan, Jian Pei, and Jiawei Han. Quotient cube: How to summarize the semantics of a data cube. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*, pages 778–789. Elsevier, 2002.

[23] Kristen LeFevre and Evimaria Terzi. Grass: Graph structure summarization. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, pages 454–465. SIAM, 2010.

[24] Cheng-Te Li and Shou-De Lin. Egocentric information abstraction for heterogeneous social networks. In *2009 International Conference on Advances in Social Network Analysis and Mining*, pages 255–260. IEEE, 2009.

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

[26] Pingchuan Ma, Rui Ding, Shuai Wang, Shi Han, and Dongmei Zhang. Xinsight: explainable data analysis through the lens of causality. *arXiv preprint arXiv:2207.12718*, 2022.

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

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

[29] R. McNamee. Confounding and confounders. *Occupational and environmental medicine*, 60(3):227–234, 2003.

[30] Yasir Mehmood, Nicola Barbieri, Francesco Bonchi, and Antti Ukkonen. 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*, pages 48–63. Springer, 2013.

[31] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[32] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[33] Fatemeh Nargesian, Erkang Zhu, Ken Q Pu, and Renée J Miller. Table union search on open data. *Proceedings of the VLDB Endowment*, 11(7):813–825, 2018.

[34] Saket Navlakha, Rajeev Rastogi, and Nisheeth Shrivastava. Graph summarization with bounded error. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 419–432, 2008.

[35] Xueyan Niu, Xiaoyun Li, and Ping Li. Learning cluster causal diagrams: An information-theoretic approach.

[36] Pekka Parviainen and Samuel Kaski. Bayesian networks for variable groups. In *Conference on Probabilistic Graphical Models*, pages 380–391. PMLR, 2016.

[37] Sepideh Pashami, Anders Holst, Juhee Bae, and Sławomir Nowaczyk. Causal discovery using clusters from observational data. In *FAIM'18 Workshop on CausalML, Stockholm, Sweden, July 15, 2018*, 2018.

[38] Marisa A. Patti, Noelle B. Henderson, Priya Gajjar, Melissa Eliot, Medina Jackson-Browne, and Joseph M. Braun. Gestational triclosan exposure and infant birth weight: A systematic review and meta-analysis. *Environment International*, 157:106854, 2021.

[39] J. Pearl and D. Mackenzie. *The book of why: the new science of cause and effect*. Basic books, 2018.

[40] Judea Pearl. *Probabilistic reasoning in intelligent systems : networks of plausible inference*. Morgan Kaufmann Series in Representation and Reasoning. Morgan Kaufmann Publishers, Inc., San Francisco, California, revised second printing. edition, 1988.

[41] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan kaufmann, 1988.

[42] Judea Pearl. *Causality : models, reasoning, and inference*. Cambridge University Press, Cambridge, U.K., 2000.

[43] Judea Pearl et al. Models, reasoning and inference. *Cambridge, UK: CambridgeUniversityPress*, 19(2), 2000.

[44] Judea Pearl, Dan Geiger, and Thomas Verma. Conditional independence and its representations. *Kybernetika*, 25(7):33–44, 1989.

[45] Sriram Pemmaraju, Steven Skiena, et al. *Computational discrete mathematics: Combinatorics and graph theory with mathematica®*. Cambridge university press, 2003.

[46] Sriram Raghavan and Hector Garcia-Molina. Representing web graphs. In *Proceedings 19th International Conference on Data Engineering (Cat. No. 03CH37405)*, pages 405–416. IEEE, 2003.

[47] Joseph D Ramsey, Kun Zhang, Madelyn Glymour, Ruben Sanchez Romero, Biwei Huang, Imme Ebert-Uphoff, Savini Samarasinghe, Elizabeth A Barnes, and Clark Glymour. Tetrad—a toolbox for causal discovery.

[48] Paul K Rubenstein, Sebastian Weichwald, Stephan Bongers, Joris M Mooij, Dominik Janzing, Moritz Grosse-Wentrup, and Bernhard Schölkopf. Causal consistency of structural equation models. *arXiv preprint arXiv:1707.00819*, 2017.

[49] Zeqian Shen, Kwan-Liu Ma, and Tina Eliassi-Rad. Visual analysis of large heterogeneous social networks by semantic and structural abstraction. *IEEE transactions on visualization and computer graphics*, 12(6):1427–1439, 2006.

[50] Shohei Shimizu, Patrik O Hoyer, Aapo Hyvärinen, Antti Kerminen, and Michael Jordan. A linear non-gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7(10), 2006.

[51] Qi Song, Yinghui Wu, Peng Lin, Luna Xin Dong, and Hui Sun. Mining summaries for knowledge graph search. *IEEE Transactions on Knowledge and Data Engineering*, 30(10):1887–1900, 2018.

[52] Daniel A Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 563–568, 2008.

[53] P. Spirtes et al. *Causation, prediction, and search*. MIT press, 2000.

[54] Yuanyuan Tian, Richard A. Hankins, and Jignesh M. Patel. Efficient aggregation for graph summarization. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 567–580, New York, NY, USA, 2008. Association for Computing Machinery.

[55] Yuanyuan Tian, Richard A Hankins, and Jignesh M Patel. Efficient aggregation for graph summarization. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 567–580, 2008.

[56] Santtu Tikka, Jouni Helske, and Juha Karvanen. Clustering and structural robustness in causal diagrams. *arXiv preprint arXiv:2111.04513*, 2021.

[57] Matthew J. Vowels, Necati Cihan Camgoz, and Richard Bowden. D'ya like dags? a survey on structure learning and causal discovery, 2021.

[58] Yuhao Wen, Xiaodan Zhu, Sudeepa Roy, and Jun Yang. Interactive summarization and exploration of top aggregate query answers. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 11, page 2196. NIH Public Access, 2018.

[59] Marco A Wiering et al. Evolving causal neural networks. In *Benelearn'02: Proceedings of the Twelfth Belgian-Dutch Conference on Machine Learning*, pages 103–108, 2002.

[60] Sewall Wright. Correlation and Causation. pages p. 557–585., 1921. Num Pages: USDA.

[61] Brit Youngmann, Sihem Amer-Yahia, and Aurelien Personnaz. Guided exploration of data summaries. *arXiv preprint arXiv:2205.13956*, 2022.

[62] Brit Youngmann, Michael Cafarella, Babak Salimi, and Zeng Anna. Causal data integration. *arXiv preprint arXiv:2305.08741*, 2023.

[63] Jiaming Zeng, Michael F Gensheimer, Daniel L Rubin, Susan Athey, and Ross D Shachter. Uncovering interpretable potential confounders in electronic medical records. *Nature communications*, 13(1):1–14, 2022.

[64] Y Zhang and Z. Ives. Finding related tables in data lakes for interactive data science. In *SIGMOD*, pages 1951–1966.

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

[66] Linhong Zhu, Majid Ghasemi-Gol, Pedro Szekely, Aram Galstyan, and Craig A Knoblock. Unsupervised entity resolution on multi-type graphs. In *The Semantic Web–ISWC 2016: 15th International Semantic Web Conference, Kobe, Japan, October 17–21, 2016, Proceedings, Part I 15*, pages 649–667. Springer, 2016.

[67] Shengyu Zhu, Ignavier Ng, and Zhitang Chen. Causal discovery with reinforcement learning. *arXiv preprint arXiv:1906.04477*, 2019.