Summer 2014

Causal Discovery for Relational Domains: Representation, Reasoning, and Learning

Marc Maier
University of Massachusetts - Amherst

CAUSAL DISCOVERY FOR RELATIONAL DOMAINS: REPRESENTATION, REASONING, AND LEARNING

A Dissertation Presented

by

MARC E. MAIER

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

September 2014

Computer Science

CAUSAL DISCOVERY FOR RELATIONAL DOMAINS: REPRESENTATION, REASONING, AND LEARNING

A Dissertation Presented

by

MARC E. MAIER

Approved as to style and content by:

David D. Jensen, Chair

Andrew G. Barto, Member

Hanna M. Wallach, Member

Andrew V. Papachristos, Member

Lori A. Clarke, Chair
Computer Science

To my wonderful children, Ella and Max.
May you always reach the goals you set.

Problems worthy
of attack
prove their worth
by hitting back.

—*Piet Hein*

ACKNOWLEDGMENTS

This thesis would not exist without the direction of my advisor, David Jensen. I have been incredibly fortunate to have had the opportunity to work with David; he has time and again shown me (and many others) the path to being a successful scientist and researcher. David continually inspires me, and he has always provided me with support—both professionally and personally. He is truly the most influential teacher and mentor I've ever had. I am also grateful for the feedback I've received from my committee—Andrew Barto, Hanna Wallach, and Andrew Papachristos. I look forward to more feedback throughout my career.

The students and staff (past and present) of the Knowledge Discovery Laboratory have been a pleasure to work alongside over my graduate career. The graduate students are some of the most intelligent and thoughtful individuals I know: David Arbour, James Atwood, Elisabeth Baseman, Andrew Fast, Lisa Friedland, Dan Garant, Amanda Gentzel, Michael Hay, Phil Kirlin, Katerina Marazopoulou, Jennifer Neville, Hüseyin Oktay, Matthew Rattigan, and Brian Taylor. I would like to thank Dan Corkill for offering helpful research advice, Agustin Schapira and Matt Cornell for helping me become a better coder and developing the infrastructure that has supported much of my research, and Deb Bergeron for shielding me from administrative headaches. I am also grateful for Cindy Loiselle's help to become a more careful writer—I will always strive to have accurately formatted references.

Among my graduate colleagues, I am especially indebted to my co-authors. Matt Rattigan took me under his wing as an incoming student and helped me enjoy early research success. Brian's camaraderie was essential to surviving graduate coursework and developing research ideas. Andy was always willing to discuss graduate school

and sports, Hüseyin made sure that I carefully considered every assumption, Katerina has been instrumental in fostering the theoretical rigor that appears in this thesis, and David's creativity and knowledge of the literature have helped me see new research connections.

Several professors were influential before my graduate tenure began. I extend my gratitude to Nathaniel Whitaker and Panayotis Kevrekedis for starting me on a successful research path while I finished my undergraduate degrees and Robert Moll for his advice during my transition from an undergraduate to a graduate student.

I would also like to thank those who have supported me, on a personal note, throughout the years. I appreciate that my friends and extended family, on both sides, have rarely questioned the length of my education and have been willing to trust that a research doctorate will lead to a fulfilling career. My grandparents, Ruth and Max, continually instilled the value of education in a way that I intend to pass on to my children. I miss you both tremendously. My parents, Gary and Nicole, have been and continue to be extremely supportive, and my father, in particular, always believed that research and a Ph.D. were in my future, even when I was initially skeptical. My brother, David, and I always seem to follow similar paths in life; I am grateful that he led me to discover such a promising field in computer science.

Finally, it is difficult to describe in words just how thankful I am for my family. I feel incredibly lucky to have such a supportive, loving, and joyous home in my wife, Bri, and our beautiful children, Ella and Max. I thank the three of you for your unending patience while I finish this chapter. I hope to provide the same support as you set and meet your goals throughout life.

ABSTRACT

CAUSAL DISCOVERY FOR RELATIONAL DOMAINS: REPRESENTATION, REASONING, AND LEARNING

SEPTEMBER 2014

MARC E. MAIER

B.S., UNIVERSITY OF MASSACHUSETTS AMHERST

M.S., UNIVERSITY OF MASSACHUSETTS AMHERST

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor David D. Jensen

Many domains are currently experiencing the growing trend to record and analyze massive, observational data sets with increasing complexity. A commonly made claim is that these data sets hold potential to transform their corresponding domains by providing previously unknown or unexpected explanations and enabling informed decision-making. However, only knowledge of the underlying causal generative process, as opposed to knowledge of associational patterns, can support such tasks.

Most methods for traditional causal discovery—the development of algorithms that learn causal structure from observational data—are restricted to representations that require limiting assumptions on the form of the data. Causal discovery has almost exclusively been applied to directed graphical models of propositional data that assume a single type of entity with independence among instances. However, most real-world domains are characterized by systems that involve complex interactions

among multiple types of entities. Many state-of-the-art methods in statistics and machine learning that address such complex systems focus on learning associational models, and they are oftentimes mistakenly interpreted as causal. The intersection between causal discovery and machine learning in complex systems is small.

The primary objective of this thesis is to extend causal discovery to such complex systems. Specifically, I formalize a *relational* representation and model that can express the causal and probabilistic dependencies among the attributes of interacting, heterogeneous entities. I show that the traditional method for reasoning about statistical independence from model structure fails to accurately derive conditional independence facts from relational models. I introduce a new theory—*relational d-separation*—and a novel, lifted representation—the *abstract ground graph*—that supports a sound, complete, and computationally efficient method for algorithmically deriving conditional independencies from probabilistic models of relational data. The abstract ground graph representation also presents causal implications that enable the detection of causal direction for bivariate relational dependencies without parametric assumptions. I leverage these implications and the theoretical framework of relational d-separation to develop a sound and complete algorithm—the *relational causal discovery* (RCD) algorithm—that learns causal structure from relational data.

TABLE OF CONTENTS

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

CHAPTER 1

INTRODUCTION

Advances in machine learning and data mining have introduced powerful tools for modeling observational data, but most of these algorithms merely identify statistical associations. While associational models are useful for predicting values of unobserved variables, they are limited in their utility. Informing actions and producing explanations require modeling *causal* dependence rather than statistical association.

Researchers in a wide range of disciplines study complex systems and pose questions that require causal answers. Examples include (1) social scientists determining if a particular reading program improves student test scores, (2) economists asking if changing financial regulation will reduce fraud, and (3) medical researchers investigating how prescription drugs induce adverse health effects. In many cases, it would be advantageous to answer these questions passively or retrospectively, without intervention, because experimentation may be infeasible due to ethical or logistical constraints. Even in domains in which experiments are feasible, leveraging observational data would allow experimental resources to be spent more effectively.

Over the past several decades, a growing community of researchers in computer science, statistics, philosophy, and social science has focused on methods for discovering causal dependencies from observational data. This work has uncovered a number of basic methods, including algorithms for learning the structure of causal models and fundamental principles necessary for valid causal reasoning. However, the vast majority of this work is limited to a single knowledge representation—directed graphical models of *propositional* data—that is insufficient to describe many real-world do-

mains. A *relational* representation is more expressive as it can describe systems of interacting, heterogeneous entities (as opposed to a single entity type), but little attention has been devoted to its connection with causality.

This thesis focuses on developing tools and algorithms to support the discovery of causal knowledge in systems of interacting entities (i.e., inherently relational domains). To do so, this research explores the intersection of causal inference, machine learning, and knowledge representation. Specifically, it extends the basic framework for traditional, propositional causal discovery by (1) formalizing a relational model representation, (2) deriving its probabilistic and statistical implications, and (3) developing an algorithm that exploits those implications to learn causal structure. This thesis describes three primary contributions that lead to independently useful capabilities:

(1) **Representation**: Drawing from prior work for representing causal dependencies with Bayesian networks [122, 172] and relational representations, such as probabilistic relational models [53], I formalize fundamental concepts of relational data and models that are sufficient to represent causality and reason about conditional independence.

    This formalization provides a language for modeling relational data. I provide a complete characterization of the space of relational variables and dependencies, which is useful for analyzing the joint space of some target relational domain. I also introduce precise semantics for instantiating relational variables with data, which informs practical implementations for data retrieval.

(2) **Reasoning**: I show that the theory of d-separation for deriving conditional independence facts from Bayesian network structure does not apply directly to the structure of relational models. As a result, I develop the theory of relational d-separation and introduce a lifted representation—the abstract ground

graph—that supports a sound and complete method for deriving the conditional independencies encoded by relational models.

Although the internal representation that supports relational d-separation is complex, it need not be exposed to end users. The inputs and outputs are relatively simple concepts: The dependence structure and schematic of a relational domain contain sufficient information to produce conditional independence facts that should hold in the modeled data. Such a tool can enable practitioners to posit a relational model, derive conditional independence implications, and check those implications on their data to support or refute their model.

(3) **Learning**: Leveraging the theory of relational d-separation and the abstract ground graph representation, I present a sound and complete algorithm—the relational causal discovery (RCD) algorithm—that learns causal models from relational data. The completeness result hinges on a new method for identifying the direction of causality for bivariate relational dependencies with no parametric assumptions. This contribution provides the relational analog to the foundational work of the sound and complete PC algorithm for learning causal Bayesian networks [172, 106].

The relational causal discovery algorithm is an automated approach to inferring the causal structure present in a relational data set. The framework behind RCD shows that the basic logic of PC can be extended to substantially more expressive data representations. Similar to how traditional d-separation supports the PC algorithm, the reasoning capabilities of relational d-separation enable RCD. Additionally, the method for orienting bivariate relational dependencies demonstrates that relational data, when combined with relational d-separation, enable powerful new ways to infer causal dependence. RCD should be viewed as a tool for relational data that can provide practitioners with a joint structure that

encodes all conditional independencies, which can be used to constrain hypotheses for experimental validation and to design interventions to produce change in their domain.

1.1 Motivation

The contributions outlined above are motivated by a variety of situations in which analysts wish to understand and manage complicated systems of interacting components. For example, consider a problem domain in which a company employs individuals, develops products, and is organized by different business units. For any company, it would be beneficial to determine which factors influence the productivity of employees, the viability of different products, and the overall revenue and budget flowing throughout the organization. With this type of knowledge, executives could make strategic decisions that affect the operation of their company. In other words, a correctly learned causal model would allow managers to understand the underlying dynamics of their company and guide actions that achieve desired outcomes. This example helps motivate three main dimensions of this thesis, described in the following three subsections.

1.1.1 Causal

A causal model can support a wider array of decisions than an associational model. Associational and causal knowledge differ in their utility and the tasks they can address. If two variables X and Y are statistically associated, then knowing the value of X provides information about the value of Y. However, if Y is *causally* dependent on X, then *changing* the value of X will result in a change in the distribution of Y. Associational knowledge can be used to predict the values of unobserved variables, assuming that data instances are drawn from the same distribution used to learn the model. In contrast, causal models can be used to predict the outcomes

of interventions—actions that may deviate from the probability distribution used to learn the model. Causality allows for reasoning about the consequences of actions, and causal models provide better capabilities for generating explanations of observed associations. If the executives wish to produce change in the company, then only a causal model could support their decision-making.

Identifying statistical associations is strictly less challenging than causal discovery because association *underdetermines* causation. This is due to the fact that an observed association can stem from several different potential causal structures. If X and Y are statistically associated, then it could be that X causes Y, Y causes X, a third set of variables, $\mathbf{Z}$, causes both X and Y, or even a third variable, Z, is a common effect of X and Y, and by conditioning on Z (or sampling certain values), dependence is induced between X and Y. Each causal structure implies different potential actions. This thesis focuses on reasoning about conditional independence to infer causal structure.

1.1.2 Relational

A relational representation can support learning models that are more accurate than those expressible in a non-relational representation. Many real-world domains, such as organizations, are complex, involving interactions among different classes of entities. This type of data can be expressed by relational representations, as opposed to propositional representations that only model a single entity class (e.g., employees). Modeling relational data can provide more effective and accurate structure learning than would be achieved by limiting the representation and jettisoning potentially valuable information for causal discovery. This thesis focuses on a highly expressive model representation: probabilistic models of relational data.

This thesis also discusses three key concepts that are rarely mentioned as separate components for propositional models but are critical for relational models: the

schema, the *model*, and the *ground graph*. The schema is a top-level description of the entities and relationships (i.e., classes of interactions among entities) that occur in a given domain, as well as the attributes that exist for those entities and relationships. For propositional domains, there is a single type of entity with no relationships, and consequently, the schema is typically implicit.

The model contains probabilistic dependencies among the attributes of entities and relationships, constrained by the structure of the schema. Models of relational data are much more expressive than their propositional counterparts because relational schemas are much less restrictive than propositional schemas. A model of propositional data is restricted to dependencies among the variables of a single type of entity. In contrast, a relational model can capture any dependence that follows the relationships among different entity types (e.g., from variables on individuals to variables on products they develop). This additional class of *relational* dependencies enables complex dependence structures that improve causal modeling beyond the more restrictive propositional representation. Section 5.5 provides empirical evidence that causal discovery in relational representations leads to more accurate and identifiable causal models than in propositional representations.

Both the schema and the model are *templates*. They describe the structure of the data and the dependencies that hold among variables in the distribution that generates the data. A third concept is the ground graph, which is an *instantiation* of the model template to instances of random variables that belong to the sets of entity and relationship instances in the data. The dependence structure that manifests in ground graphs is an important property that supports reasoning about conditional independence in relational models and will be revisited in Chapter 4.

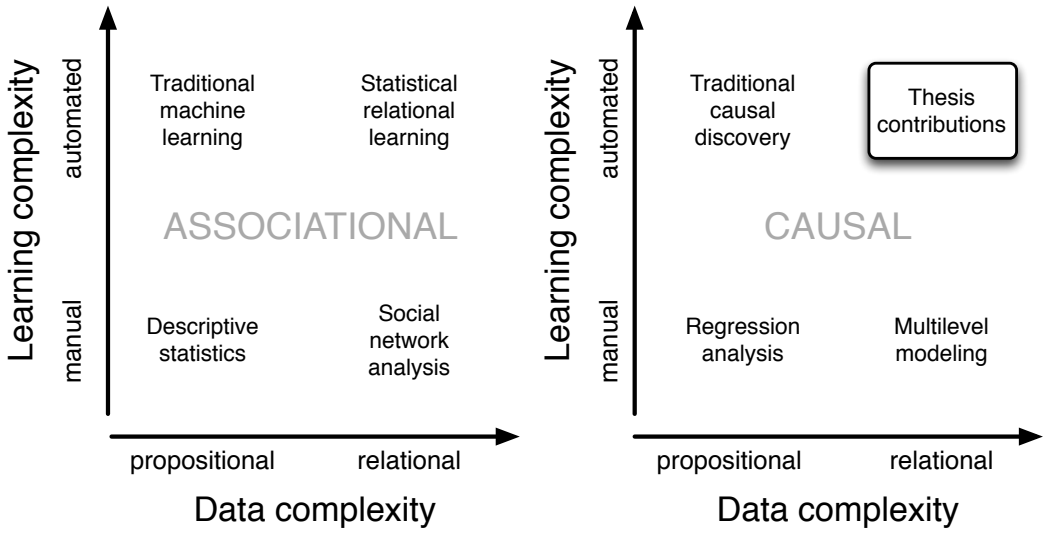


Figure 1.1: Dissertation landscape: The three dimensions addressed in this thesis (causal, relational, and automated learning) can be characterized as extreme points on three axes of complexity (model, data, and learning). The left plot depicts associational models, and the right plot depicts causal models. The contributions of this thesis are positioned on this chart with respect to current scientific practice.

1.1.3 Learning

A model that is learned from data can be more accurate than one that relies exclusively on the knowledge of domain experts. The underlying dynamics of many domains are unknown, especially when the system is non-deterministic and complex, such as in large organizations. Current practice is to (largely) solicit the knowledge of domain experts, but many systems are too complex or may operate in unexpected ways. This thesis focuses on learning the causal structure of domains from observational data.

1.2 Landscape

The three dimensions listed in the previous section (causal, relational, and learning) can be placed on three different axes of complexity that help describe where the contributions of this thesis lie with respect to the landscape of current scientific practice. The three axes are: *model complexity*, spanning associational and causal models;

data complexity, spanning propositional and relational data; and *learning complexity*, spanning manual and automated methods. Figure 1.1 relates these axes and provides an example class of techniques for each point in this complexity space.

The simplest position involves a manual approach for describing associations in propositional data. This could include a wide range of tools for descriptive statistics or exploratory data analysis, such as identifying correlations between columns in a data table loaded into statistical software. If the data include multiple linked tables, perhaps involving the relationships among individual instances, then manual approaches for discovering associations could include a range of social network analytic methods [157]. Analysis using descriptive statistics, such as degree distributions, node centrality, graph clustering, and measures of homophily or social contagion, are examples of this position, which covers a vast community.

Automated methods for learning associational models of propositional data have been a major task in machine learning and data mining for the past several decades. Algorithms for learning Markov random fields, Bayesian networks, and factor graphs all fall into this category [84]. These methods have been very successful and widely deployed in many real-world systems. Over the past 15 years, machine learning has expanded to richer data representations, such as relational, which has led to the subfields of statistical relational learning and inductive logic programming [57]. The goal in these fields is to develop new algorithms (or extend propositional algorithms) to capture the complexity of relational data. Notable representations and corresponding learning algorithms include probabilistic relational models [85] and Markov logic networks [143].

Beyond associations, causal knowledge has been the object of pursuit by many researchers in a variety of disciplines, including the social sciences, behavioral sciences, and life sciences. These researchers and practitioners often require causal knowledge because their interests center on influencing policy or gaining a deep understanding

of their domains. Many approaches used by these researchers are manual, typically addressing a single potential cause or effect. While the applications may make restrictive assumptions, the methods employed have a strong mathematical and statistical basis and can be quite powerful (e.g., regression analysis on a data table, interpreted causally [6]; multilevel modeling to incorporate a fixed, hierarchical relational structure [51]). Improving the capabilities of regression analysis is an active area of research, leading to some innovative methods, such as those involving Bayesian or nonparametric regression. These analyses can also require complex forethought (e.g., experimental and quasi-experimental designs [160]). However, the work in this complexity space is typically informal with respect to data and model representations, often relying on implicit data complexity.

As described above, traditional causal discovery almost exclusively focuses on automated learning of causal models from propositional data. The goal of this thesis is to develop formal theory and algorithms to accurately represent, reason about, and discover causal dependencies from *relational* data. This work can be regarded as providing the theoretical foundations for extending causal discovery research to more realistic settings.

There are several additional key distinctions concerning the objectives of this thesis with respect to other scientific endeavors. The overall goal is to learn causal *structure*, which is a component of a larger setting. Parameter estimation, given a model structure, fits appropriate parameters to each dependency. After fully specifying a model (structure and parameters), the task of inference applies the model to new or future data. Both parameter estimation and inference receive considerable attention in machine learning and statistics and are complementary tasks to structure learning.

An alternative specification for the model complexity dimension could span conditional and joint models. In this work, the intention is to represent, reason about,

and learn *joint* models. Conditional models are useful for predicting the value of a single variable, but joint models can produce any conditional model and can predict the state of an entire system of variables. However, almost all manual approaches and many automated methods are devoted to conditional models or even single dependencies. Joint models rely on conditional models (e.g., the joint distribution of a Bayesian network factors into a product of conditional distributions), but joint models facilitate causal discovery and causal reasoning. (Section 2.1.3 provides evidence.)

Our approach to structure learning employs the *constraint-based* paradigm, relying on local tests of conditional independence. (Section 2.3.1 provides details of a widely used constraint-based algorithm.) An alternative approach to structure learning is the search-and-score paradigm, a technique that selects the most likely model by searching across the space of possible models. Search-and-score algorithms accurately model joint probability distributions but are typically computationally intensive. By exploiting local computations, constraint-based algorithms can overcome some complexity concerns, and they are generally more well-suited for causal discovery because of the connection between conditional independence and causal structure.

Finally, the scope of this thesis is on purely *automated* techniques. Mixed-initiative systems—algorithms that interact with users—may be the most effective approach to causal discovery, similar to how computer-aided design systems can assist engineers. However, the research in this thesis provides a necessary foundation that can enable future interactive approaches to causal discovery.

1.3 Dissertation Structure

In the next chapter, we provide background material on traditional causal discovery. The contributions of this thesis build on much of that material, especially Bayesian network representation, the theory of d-separation, and the constraint-based PC algorithm for learning causal structure from propositional, observational data.

The organization of the background chapter mirrors the three main contributions of this thesis on representation, reasoning, and learning for causality.

Chapter 3 describes the formal representations used for relational data and models that are necessary to reason about conditional independence. Chapter 4 introduces relational d-separation and the abstract ground graph representation to support reasoning about conditional independence in relational models. Chapter 5 presents the implications of relational d-separation and abstract ground graphs for learning the causal structure of relational models. Finally, Chapter 6 concludes by providing an assessment of the broader implications of this work and offers potential high-impact areas of future research that build on this dissertation.

CHAPTER 2

BACKGROUND

Relational data and models are strictly more expressive than their propositional counterparts. The vast majority of prior and current work on causal discovery focuses on propositional domains but has led to a solid theoretical foundation on which the contributions presented in this dissertation build. Therefore, it is necessary to provide background on causal discovery from propositional data. This thesis can largely be viewed as introducing major technical and theoretical extensions in the *representation*, *reasoning*, and *learning* of what researchers in causal discovery have previously accomplished.

This background chapter parallels the three major chapters that present the contributions of this dissertation. First, Section 2.1 reviews the primary representation—Bayesian networks—used to encode causal knowledge, as well as alternative representations for causality used by many practitioners. Then, Section 2.2 defines *d*-separation, a useful theory that enables reasoning about the conditional independence and causal implications of Bayesian networks. Finally, Section 2.3 describes the constraint-based PC algorithm—one of the most influential causal discovery algorithms—and alternative strategies and paradigms for learning causal structure of propositional domains.

2.1 Representation

Propositional representations consist of a single entity type and its attributes. Strong assumptions are made about the underlying distribution, most notably that

sampled data instances are independent and identically distributed (IID). The first condition assumes that the variables on any given data instance are marginally independent of the variables of any other data instance. The second condition assumes that every data instance is drawn from the same underlying joint probability distribution. IID data—also referred to as propositional data[1]—are effectively represented as a single table, where rows correspond to the independent instances and columns are attributes of those instances.

These assumptions lead to a mathematical simplicity that has provided causal discovery researchers with a useful testbed for developing theoretical foundations, while simultaneously proving to be an effective modeling choice in practice. However, many domains are not well characterized by a propositional representation, and in the ensuing chapters, we remove these assumptions in an effort to model more complex, realistic domains.

Our approach is similar to the main line of causal discovery research, especially with respect to the underlying representation of directed graphical models. The following subsections define Bayesian networks and the assumptions that enable them to represent causality, and we describe alternative, non-graphical approaches to modeling causality that have been adopted by various communities.

2.1.1 Bayesian networks

Bayesian networks[2] are widely used probabilistic graphical models of propositional data that are capable of compactly representing a joint probability distribution [120]. Bayesian networks enable an array of useful tasks by supporting inference over a set of variables, and they have been successfully applied to model many domains, ranging

[1]IID data are typically referred to as *propositional* because the data can be equivalently expressed under propositional logic.

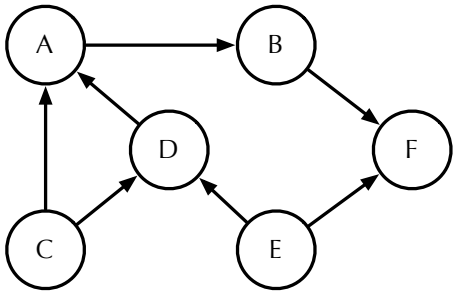


Figure 2.1: A simple example of a Bayesian network with six random variables (nodes) and seven dependencies (edges). The structure of the Bayesian network is paired with a set of conditional probability distributions for each node (not pictured).

from bioinformatics [20] and medicine [61] to computer vision [142] and information retrieval [45].

The structure of a Bayesian network is represented as a directed acyclic graph $\mathcal{G} = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V}$ is a set of nodes corresponding to random variables and $\mathbf{E} \subset \mathbf{V} \times \mathbf{V}$ is a set of edges encoding the probabilistic dependencies among the variables. If there is an edge $X \to Y$, we say that X is a *parent* of Y and Y is a *child* of X. The set of parents for some node V is denoted by $\mathit{parents}(V)$. The nodes that can reach V by a directed path are called *ancestors* and denoted by $\mathit{anc}(V)$, and the nodes that V can reach by a directed path are called *descendants* and denoted by $\mathit{desc}(V)$. The graph is acyclic when there are no directed paths from any node to itself.

Each random variable $V \in \mathbf{V}$ is also associated with a conditional probability distribution $P\big(V \mid \mathit{parents}(V)\big)$, where $\mathit{parents}(V) \subseteq \mathbf{V} \setminus \{V\}$. If the joint probability distribution $P(\mathbf{V})$ satisfies the Markov condition for $\mathcal{G}$, then the joint distribution can be factored as a product of the conditional distributions:

$$P(\mathbf{V}) = \prod_{V \in \mathbf{V}} P\big(V \mid \mathit{parents}(V)\big).$$

[2]The term "Bayesian network" is considered by some researchers to be a misnomer as the models themselves do not necessarily entail a Bayesian interpretation of probability. Consequently, they are also referred to as "belief networks" or simply "Bayes nets."

The Markov condition states that every variable $V \in \mathbf{V}$ is conditionally independent of its non-descendants given its parents. In other words,

$$P\big(V \mid \mathbf{V} \setminus \{V\}\big) = P\big(V \mid \mathit{parents}(V)\big).$$

The Markov condition can also be written as

$$V \perp\!\!\!\perp \mathbf{V} \setminus \mathit{desc}(V) \mid \mathit{parents}(V)$$

for all $V \in \mathbf{V}$, where the notation $\perp\!\!\!\perp$ denotes probabilistic independence (as introduced by Dawid [34]).

The fact that Bayesian networks are propositional, encoding IID instances, is implicit in that the joint probability of a set of instances is equal to the product of the joint probability of each instance. Additionally, as stated by Russell and Norvig (Chapter 14.6 [151]), "...Bayesian networks are essentially propositional: the set of random variables is fixed and finite, and each has a fixed domain of possible values. This fact limits the applicability of Bayesian networks." This is one of the main reasons for increasing the expressiveness of the underlying representation to relational (i.e., first-order logic), as we show in the ensuing chapters.

Naïvely specifying a joint distribution by hand would require an exponential number of states, but the Markov condition enables a Bayesian network to represent this distribution with many fewer parameters. Consider the example Bayesian network displayed in Figure 2.1, which consists of six nodes and seven edges. Using the chain rule from probability theory, one can write the joint distribution as

$$\begin{aligned} P(A, B, C, D, E, F) = \; & P(A) \cdot P(B|A) \cdot P(C|A, B) \cdot P(D|A, B, C) \cdot \\ & P(E|A, B, C, D) \cdot P(F|A, B, C, D, E) \end{aligned}$$

In this example, if all variables are binary (i.e., have domain $\{0, 1\}$), then the joint distribution would involve $2^6 = 64$ states and 63 parameters. However, if the Markov condition holds, then the Bayesian network can factor the joint distribution into a

product of smaller conditional probability distributions. For the example, the joint distribution could be written as

$$P(A, B, C, D, E, F) = P(C) \cdot P(E) \cdot P(D|C, E) \cdot P(A|C, D) \cdot P(B|A) \cdot P(F|B, E)$$

Factored in this way, only $2^0 + 2^0 + 2^2 + 2^2 + 2^1 + 2^2 = 16$ parameters would be necessary to fully specify the model.

The Markov condition ties the structure of the model $\mathcal{G}$ to the set of conditional independencies that hold over all compatible probability distributions $\mathcal{P}$. All conditional independence facts can be derived from the Markov condition and the structure of $\mathcal{G}$, but they may involve complex manipulations of the joint distribution and various probability axioms. In Section 2.2.1, we describe d-separation, a set of graphical rules that *algorithmically* derive conditional independence facts directly from the graphical structure of the model. These two approaches (the Markov condition and d-separation) have been shown to produce equivalent sets of conditional independence facts from Bayesian networks [191, 47, 115].

With respect to directed graphical models, this thesis focuses on reasoning about conditional independence given model structure and learning model structure from observational data. Many researchers work on two complementary tasks: parameter estimation and inference. If the structure of a Bayesian network is known, there are various methods to estimate its parameters. These methods generally involve maximum-likelihood estimation if the data set is complete, Bayesian estimation if starting with a prior distribution, or expectation-maximization if some data values are missing. The task of inference generally involves inferring the values of unobserved variables under different settings. Methods for inference can be grouped into two main classes: exact methods, such as variable elimination and clique tree propagation, and approximate methods, such as belief propagation, Monte Carlo simulations, and variational methods. We refer the reader to the accessible introductions to Bayesian

networks by Charniak [19] and Darwiche [32], as well as the lengthier overviews by Jensen [78] and Neapolitan [115].

2.1.2 Modeling causality with Bayesian networks

The definition of causality has been debated by philosophers since the time of Aristotle, and there remain different viewpoints for a precise account of causation. In this work, we champion Pearl's and Spirtes et al.'s view of causality, which is a combination of probabilistic and interventionist causation [122, 172]. Specifically, we say that X causes Y ($X \to Y$) if the conditional distribution of Y changes upon intervening on X. In Pearl's notation, X causes Y if for some y in the domain of Y and two different values x, x' in the domain of X, $P\big(Y = y \mid do(X = x)\big) \neq P\big(Y = y \mid do(X = x')\big)$, where the *do* operator corresponds to an intervention on X. Additionally, we make no restrictions on which variables can be causes. This definition of causality contrasts with deterministic causation (i.e., manipulating X necessarily results in a change in the value of Y as opposed to its conditional distribution) and a pure manipulation theory under which only manipulable variables can be causes [65].

This view of causality is compatible with using directed probabilistic graphical models to represent causal dependencies. Directed acyclic graphs can be an appropriate representation because causal dependencies are predominantly irreflexive (X does not cause itself), asymmetric (if X causes Y, then Y does not cause X), and transitive (if X causes Y and Y causes Z, then X is causal for Z). Under a few formal assumptions, Bayesian networks can be interpreted causally, with directed edges corresponding to direct causal dependencies rather than mere probabilistic dependencies. For the following definitions, let $\mathcal{G} = (\mathbf{V}, \mathbf{E})$ be a causal Bayesian network and let $\mathcal{P}$ be the joint probability distribution over $\mathbf{V}$.

Definition 2.1.1 (Causal sufficiency) $\mathcal{G}$ is *causally sufficient* if, for all pairs of variables $X, Y \in \mathbf{V}$, all common causes are observed, measured, and included in $\mathbf{V}$.

For structure learning, if there exists a latent common cause of two variables, then we may incorrectly conclude causal dependence between them instead of choosing the correct, albeit unrepresented, causal model. This does not preclude any variable from having a latent cause, only cases where two variables share the same latent cause. Causal sufficiency is a relatively strong assumption, and there are techniques and more complex representations that can relax or remove this assumption.

Causal sufficiency is also a necessary assumption for the causal Markov condition to hold. The causal Markov condition is identical to the Markov condition, replacing parents with direct causes and non-descendants with non-effects.

Definition 2.1.2 (Causal Markov condition) Given that $\mathbf{V}$ is causally sufficient, $\mathcal{P}$ is Markov to $\mathcal{G}$ if each variable $V \in \mathbf{V}$ is conditionally independent of its graphical non-effects given its graphical direct causes.

The causal Markov condition (and equivalently, d-separation) has been shown to provide the correct connection between causal structure and probability distributions by tying statistical (in)dependence with causal (in)dependence [154]. However, if this assumption is violated, then two variables without a direct causal dependence may remain statistically dependent even after conditioning on all their causes. For example, conditioning on all common causes may not entirely explain the statistical association between two variables that are derivatives of the same variable.

In addition, to connect the graphical causal dependencies with appropriate interventions, we also assume *causal minimality*. This assumption restricts $\mathcal{G}$ to be the simplest structure that is Markov to $\mathcal{P}$. The reason is that a probability distribution $\mathcal{P}$ that is (causally) Markov to a graph structure $\mathcal{G}$ is also (causally) Markov to any supergraph of $\mathcal{G}$. Without this assumption, it would be possible to infer interventions from $\mathcal{G}$ that have no effect on $\mathcal{P}$ [199]. This condition is entailed when assuming the faithfulness condition (defined in Section 2.3) for structure learning, but it is necessary for purely modeling causality.

2.1.3 Alternative approaches to causality

The most recent and comprehensive theory for causal inference is the structural causal theory developed by Pearl [121, 122, 123]. The cornerstone of this theory is the causal Bayesian network described above. This approach is agnostic about functional form and supports inference and counterfactual analysis (using the *do*-calculus for representing interventions). The contributions made in this thesis only require the principles found in the causal Bayesian network representation. Below, we review alternative approaches to reasoning about or detecting causality, which are all individually useful, but they can be represented, facilitated, and often improved by the graphical models approach.

Many researchers focus on estimating the effects of individual dependencies in observational data without reference to a structural, graphical model. For example, propensity scores are a widely used method that enables matching treatment and control pairs of instances that would otherwise have the same propensity for treatment [149]. Once matched, the effect of treatment on outcome can be measured given that treatment is now essentially randomized for each pair. Conventional wisdom is to use as many covariates to model treatment as possible; however, there exist situations under which propensity score matching can increase bias if inappropriate covariates are selected [125]. Propensity score matching is an effective, yet complex statistical method for testing conditional independence and estimating causal effects, but its application can be improved by using graphical models to identify relevant and admissible covariates [124, 166, 168]. This is due to the fact that joint graphical models render the dependencies among variables transparent, leading to more effective reasoning about causal effects. Representing assumptions within a graphical framework can provide information that may be necessary to reason over individual dependencies.

Approaches to estimating individual effects are broadly captured by the literature on experimental and quasi-experimental design developed by social scientists [18, 160]. These approaches do not explicitly represent causal dependencies, but instead focus on the techniques used to determine causality. Propensity score matching can be viewed as a quasi-experimental design. Another example is the widely used instrumental variable design [5]. Typically, a researcher identifies an instrumental variable with two conditions: (1) it must be a cause of treatment and (2) it can only affect the outcome through the treatment. This enables analysis of the effect of treatment on outcome, even in the presence of latent common causes connecting treatment and outcome. While instrumental variables present another effective way to measure causal effects, their application often requires extensive domain knowledge for identification. However, the graphical models approach facilitates their discovery and can actually generalize the conditions under which they hold [16]. With the transparent structural knowledge encoded by graphical models, the applicability and robustness of instrumental variables, and likely most quasi-experimental designs, can be improved.

Another approach is to perform a controlled, randomized experiment [39], long considered the gold standard for estimating causal effects. Unfortunately, many domains are not conducive to randomization or control because of ethical or logistical concerns. While the results of experiments could be incorporated into a joint model, one of the goals of this thesis is to learn a joint causal structure from *observational* data. Typically the space is too large to run the necessary experiments to identify every causal dependency, but structure learning could be viewed as a way to constrain the hypothesis space and use experiments to validate important dependencies. Additionally, recent work by Eberhardt has shown that without strong assumptions on functional forms and the ability to execute ideal interventions—often simultaneously on multiple variables—it may not be possible to rely on experiments to completely

identify causal structure [37]. These results stand in contrast to the interventionist view of causation, under which the very concept of causality is the ability to perform an intervention [65, 194]. Furthermore, determining which variables require experimentation and control is facilitated by graphical models and its accompanying framework for identifying interventions [122].

A widely adopted theory of randomized and non-randomized experiments is the representation alternately referred to as the potential-outcome framework [150], Rubin's model [65], or the Neyman-Rubin model (since Neyman originally proposed potential outcomes solely for randomized experiments [118]). This approach maintains that modeling the causal effect of a single experiment is of primary interest. The potential-outcome framework has a strong connection to statistics, and it has clear semantics for causal inference under certain assumptions. However, common usage of these models assumes that all treatment variables can be manipulated. In contrast, the graphical models approach enforces no restrictions on which variables can be a cause since any variable has the potential to alter conditional probability distributions [122, 172]. In addition, the potential-outcome framework does not facilitate reasoning about the effects of interventions in complex joint models whereas Pearl's *do*-calculus on graphical models does [122]. Finally, the potential-outcome approach is too restrictive for the relational setting in this thesis because of its reliance on the stable unit-treatment value assumption (SUTVA). SUTVA assumes that instances are independent, which is one of the primary assumptions lifted by relational representations. Although there have been some minor relaxations for interference models (see Section 3.8.2), the potential-outcome framework appears too rigid to handle the complexity of general relational data.

Most alternative approaches to modeling causality lack the transparency and algorithmic tools afforded by using causal Bayesian networks. In the following two sections we review one main capability for reasoning about independence in Bayesian

networks (Section 2.2) and algorithms that learn the causal structure of Bayesian networks (Section 2.3).

2.2 Reasoning

As noted in Section 2.1.2, the causal Markov condition provides a connection between the causal structure of a directed acyclic graph $\mathcal{G}$ and conditional independence occurring in represented probability distributions $\mathcal{P}$. However, deriving the set of conditional independencies from $\mathcal{G}$ based on the Markov condition is cumbersome, requiring complex combinations of probability axioms. Fortunately, d-separation, a set of graphical criteria, provides the foundation for algorithmic derivation of all conditional independencies in $\mathcal{G}$ and entails the exact same set of conditional independencies as the causal Markov condition [191, 47, 115]. In the following subsections, we detail how to reason about conditional independence from model structure, and we describe two useful tasks—identifying causal effects and causal structure learning—that are enabled by this capability.

2.2.1 Defining *d*-separation

In the late 1980s, Pearl and his students devised a graphical theory, termed d-separation, that was shown to induce the exact same set of conditional independencies from a directed acyclic graph as the Markov condition. They presented proofs of both soundness [191] and completeness [47], and they developed efficient algorithms for checking independence [48, 49]. The main idea behind d-separation is to connect probabilistic dependence with graphical connection and conditional independence with graphical separation. The "d" stands for "directional" since the precise conditions for connection and separation hinge on the direction of arrows along paths in the graph structure.

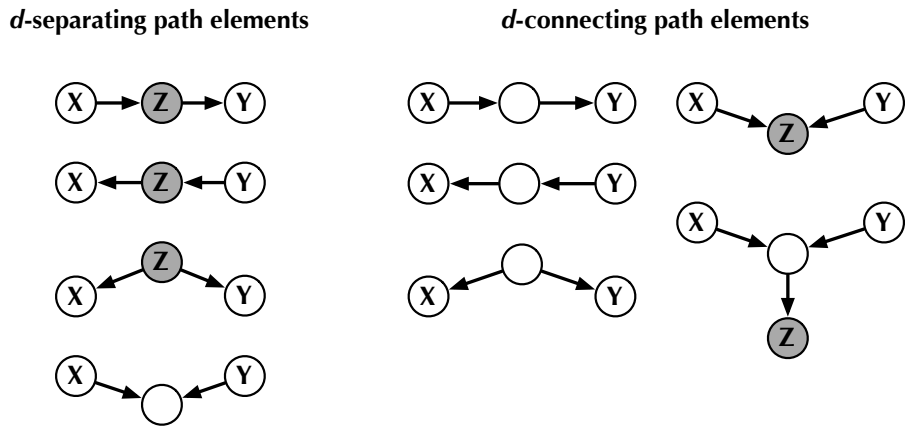


Figure 2.2: Graphical patterns of d-separating and d-connecting path elements among disjoint sets of variables $\mathbf{X}$ and $\mathbf{Y}$ given $\mathbf{Z}$. Paths for which there *exists* a non-collider in $\mathbf{Z}$ or a collider not in $\mathbf{Z}$ are d-separating. Paths for which *all* non-colliders are not in $\mathbf{Z}$ and *all* colliders (or a descendant of colliders) are in $\mathbf{Z}$ are d-connecting.

In the following definition, a path is a sequence of vertices following edges in either direction. We say that a variable V is a *collider* on a path p if the two arrowheads point at each other (collide) at V; otherwise, V is a *non-collider* on p.

Definition 2.2.1 (d-separation) Let $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ be disjoint sets of variables in directed acyclic graph $\mathcal{G}$. A path from some $X \in \mathbf{X}$ to some $Y \in \mathbf{Y}$ is *d-connected* given $\mathbf{Z}$ if and only if every collider W on the path, or a descendant of W, is a member of $\mathbf{Z}$ and there are no non-colliders in $\mathbf{Z}$. Then, we say that $\mathbf{X}$ and $\mathbf{Y}$ are *d-separated* by $\mathbf{Z}$ if and only if there are no d-connecting paths between $\mathbf{X}$ and $\mathbf{Y}$ given $\mathbf{Z}$.

Figure 2.2 depicts the graphical patterns found along paths that produce d-separation or d-connection based on Definition 2.2.1. It can be helpful to think about the flow of information (dependence) between the terminal nodes on each path. Conditioning on a common cause or intermediate variable blocks the flow of information along the path, removing any dependence. This matches the intuition behind the Markov condition: Conditioning on a parent renders a variable independent of non-descendants. This occurs in the first three d-separating path elements in Figure 2.2.

The d-separation condition is actually more general, with any non-collider on the path blocking dependence from flowing.

When conditioning on a collider or common effect, information is allowed to flow through that variable. This phenomenon, referred to as "explaining away" or Berkson's paradox [10], is best explained by Pearl's well-known example [120], paraphrased here. Assume that a car's battery (B) and fuel level (F) are marginally independent events. Both B and F are causes of an engine's ability to start (S), that is, we have the directed graph $B \rightarrow S \leftarrow F$. Imagine that you recently filled your gas tank, but your car did not start. Since we observed the value of S (car did not start), knowledge of F provides evidence of the state of the car's battery (it's most likely dead if we assume there are no other causes of S). If we did not observe whether the engine started, then we would have no reason to believe that the distribution of B is anything but its marginal. More formally, $P(B \mid F) = P(B)$, but $P(B \mid F, S) \neq P(B)$.

Applying d-separation to the example in Figure 2.1, we can identify more complex patterns. For example, B and C are not marginally independent ($B \not\perp\!\!\!\perp C$) because of the d-connecting paths $C \rightarrow A \rightarrow B$ and $C \rightarrow D \rightarrow A \rightarrow B$. However, conditioning on A renders them independent ($B \perp\!\!\!\perp C \mid A$) by blocking those two paths, and the remaining path $C \rightarrow D \leftarrow E \rightarrow F \leftarrow B$ contains colliders that are not in the conditioning set. As stated, the graphical rules also extend to sets of variables. For example, $\{C, D\} \not\perp\!\!\!\perp \{B, F\} \mid \emptyset$, but $\{C, D\} \perp\!\!\!\perp \{B, F\} \mid \{A, E\}$.

At first glance, identifying conditional independence facts using the rules of d-separation appears computationally intensive, testing a potentially exponential number of paths. However, Geiger et al. provide a linear-time algorithm based on breadth-first search and reachability on $\mathcal{G}$ [49]. An alternative method, known as the "Bayes-Ball" algorithm, provides a different intuition for how d-separation works, modeling the passing of a ball (information) among the nodes in the graph [159]. A lesser known

approach formulates the rules of d-separation into a logical language and provides an equivalent linear-time algorithm based on model checking [180].

The work in Chapter 4 presents a major extension of d-separation to directed graphical models of relational data. Since its inception, there have been several other extensions of d-separation. It was originally formulated to handle deterministic dependencies via a slight modification to the rules for colliders [49, 172]. Pearl and Dechter noted that, as defined, d-separation could also apply to directed graphs with cycles or feedback, as long as the variables were finite and discrete [126], but Neal showed that it does not generalize to arbitrary functional systems with feedback [114]. Recently, Winn worked out details for d-separation under context-specific independence by adding factors, called gates, to directed graphs and modifying the rules to account for the presence of these factors [193]. However, the most significant and useful extension to d-separation has been its generalization to m-separation on mixed graphical models, such as ancestral graphs [144]. A recent paper by Sadeghi and Lauritzen further generalized m-separation to a hierarchy of ribbonless and loopless graphs, for which directed, undirected, and mixed graphs are all special cases [153]. The work on m-separation appears to be a promising direction for extending relational d-separation to more general model classes, but new relational representations would need to be developed first.

2.2.2 Why *d*-separation is a useful theory

The conditional independence facts dervied by d-separation are guaranteed to hold in every joint distribution the model represents and consequently, in any data instance sampled from those distributions. The semantics of holding across all distributions is the main reason why d-separation is useful, enabling two large classes of applications:

(1) *Identification of causal effects*: The theory of d-separation connects the causal structure encoded by a Bayesian network to the set of probability distributions it can

represent. On this basis, many researchers have developed accompanying theory that describes the conditions under which certain causal effects are identifiable (uniquely known) and algorithms for deriving those quantities from the joint distribution. This work enables sound and complete identification of causal effects, not only with respect to conditioning, but also under counterfactuals and interventions—via the *do*-calculus introduced by Pearl [122]—and in the presence of latent variables [183, 69, 164].

(2) *Constraint-based causal discovery algorithms*: The theory of d-separation can be leveraged to constrain the hypothesis space by eliminating models that are inconsistent with observed conditional independence facts. While many distributions do not lead to uniquely identifiable models, this approach (under simple assumptions) frequently discovers useful causal knowledge for domains that can be represented as a Bayesian network. This approach to learning causal structure is referred to as the *constraint-based* paradigm, and many algorithms that follow this approach have been developed over the past 20 years. In Section 2.3, we review the PC algorithm [172] and alternative methods for causal discovery of propositional data. In Chapters 4 and 5, we formalize the theory of *relational* d-separation and introduce a sound and complete constraint-based algorithm—the relational causal discovery (RCD) algorithm [99]—that learns causal models from relational data.

2.3 Learning

The goal of traditional causal discovery, as well as the objective of this thesis, is to learn causal models from observational data. Specifically, the intention is to learn a joint causal model because reasoning about single dependencies or conditional models—the aim of many of the alternative methods described in Section 2.1.3—are more effective given the structural knowledge encoded by a joint model. Additionally, the goal in this thesis is to learn the causal *structure* of the model rather than its

parameters, an important related task. Parameter estimation can also be improved with accurate model structure.

Given this objective and the framework of causal Bayesian networks, we can now describe the general approach for learning causal structure followed in this thesis. Recall that the causal Markov condition and d-separation on a model structure $\mathcal{G}$ entail which conditional independencies should appear in distributions $\mathcal{P}$ represented by $\mathcal{G}$. However, to infer causal structure by examining independencies that hold in $\mathcal{P}$ (or data sampled from $\mathcal{P}$), we need an assumption that relates those independencies back to $\mathcal{G}$. Many causal discovery algorithms rely on the following assumption:

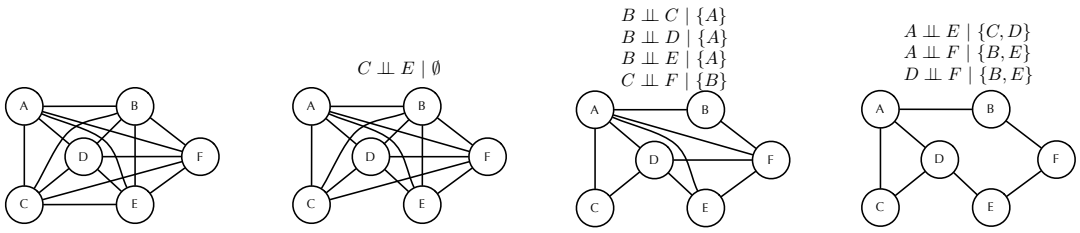
Definition 2.3.1 (Faithfulness) $\mathcal{P}$ is faithful[3] to $\mathcal{G}$ if there exist no conditional independencies in $\mathcal{P}$ that are not entailed by the causal Markov condition on $\mathcal{G}$.

The faithfulness assumption is essentially the converse of the causal Markov condition. If it does not hold, then causal dependencies in $\mathcal{G}$ may not manifest as statistical dependence in $\mathcal{P}$. For example, if the effects of two dependencies exactly cancel each other out, then those dependencies would not be identifiable in the probability distribution.

In the following section, we describe the PC algorithm, which identifies the edges in $\mathcal{G}$ that are consistent with observed conditional independencies in $\mathcal{P}$ and can determine the direction of causality for certain edges [172]. If $\mathcal{P}$ is assumed to be faithful to $\mathcal{G}$ (and causally sufficient), then PC is known to be sound and complete, learning the Markov, or likelihood, equivalent set of causal models. The relational causal discovery algorithm described in Chapter 5 provides the equivalent result for relational models. The final section presents alternative algorithms to PC for learning the structure of Bayesian networks.

---

[3]This assumption is occasionally referred to as stability [122].



(a) Start of PC (b) After $d = 0$ (c) After $d = 1$ (d) After $d = 2$

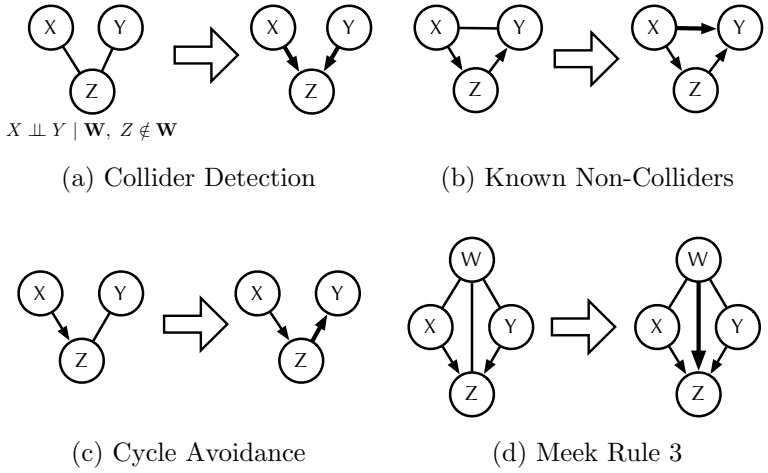
Figure 2.3: Example causal skeleton after different stages of Phase I of the PC algorithm for the model in Figure 2.1.

2.3.1 The PC algorithm

The PC algorithm[4] is arguably the most well-known constraint-based causal discovery algorithm for propositional data [172]. The algorithm was devised 20 years ago, and there are continual updates to TETRAD (`http://www.phil.cmu.edu/projects/tetrad/`), and the R package pcalg [79], two open-source software packages that implement the PC algorithm and various other tools. The execution of PC is separated into two distinct phases and produces a partially directed acyclic graph (PDAG) that corresponds to the Markov equivalence class of statistically indistinguishable causal models. The causal structure is guaranteed to be sound and complete under the three assumptions of causal sufficiency, the causal Markov condition, and faithfulness [106]. The algorithm also assumes perfect tests of conditional independence, such as a d-separation oracle on $\mathcal{P}$.

The first phase, *skeleton identification*, determines the undirected graphical structure that encodes the set of conditional independencies present in the data. An edge between two variables indicates statistical dependence, whereas the absence of an edge corresponds to marginal or conditional independence. PC begins with a fully connected graph and iteratively tests all pairs of variables X and Y for marginal independence followed by conditional independence over all possible sets of conditioning

[4]PC stands for Peter Clark, the first names of its original authors.



(a) Collider Detection (b) Known Non-Colliders

(c) Cycle Avoidance (d) Meek Rule 3

Figure 2.4: The four edge orientation rules used in Phase II of the PC algorithm.

variables of increasing size d. If $X \perp\!\!\!\perp Y \mid \mathbf{S}$ for some set of conditioning variables $\mathbf{S}$, then PC removes the edge between X and Y. Figure 2.3 depicts an example of the different stages of Phase I as applied to the true underlying model in Figure 2.1. Only variables that remain connected are considered as possible conditioning variables—as edges are removed, the set of potential conditioning variables becomes smaller. Phase I finishes when all possible conditional tests of independence have been exhausted (typically restricted to some maximum conditioning set size *depth*), and the final undirected skeleton encodes the observed set of conditional independencies. Additionally, for each pair of variables, the algorithm records the separating set (*sepset*) that includes the variables that, when conditioned on, render the pair independent. These separating sets are then used in the second phase of the algorithm.

The second phase, referred to as *edge orientation*, applies a set of rules that uniquely determines the correct causal structure consistent with the conditional independencies in the skeleton. PC uses the following four rules for orienting edges [106, 172]. These rules exploit the theory of d-separation and the assumption of model acyclicity. The rules are also displayed pictorially in Figure 2.4.

Definition 2.3.2 (Collider Detection) If $X - Z - Y$ and $Z \notin sepset(X, Y)$, then orient as $X \rightarrow Z \leftarrow Y$.

This rule exploits a concept used in d-separation (and the underlying phenomenon in Berkson's paradox)—two variables become dependent conditional on a common effect. If a third variable Z does not render X and Y conditionally independent yet exhibits association with both of them, it must be a collider. Collider detection is exhaustively run before the remaining edge orientation rules.

Collider detection enables additional orientations. If $\langle X, Z, Y \rangle$ is not oriented as a collider, but X is a known cause of Z, then only a single causal model can explain the association between Z and Y (namely, Z causes Y).

Definition 2.3.3 (Known Non-Colliders) If $X \rightarrow Z - Y$ and $\langle X, Z, Y \rangle$ is not a collider, then orient as $X \rightarrow Z \rightarrow Y$.

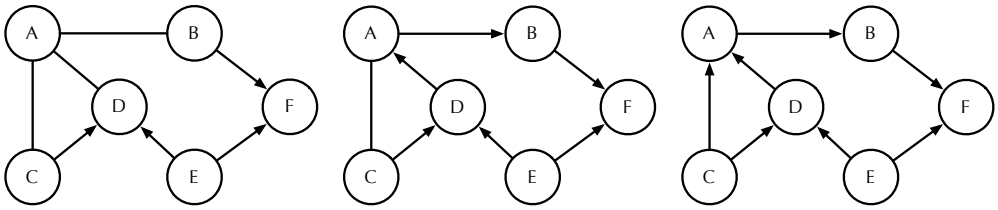
The third rule stems from assuming the data are atemporal and that causality is transitive. Orienting the dependency in the reverse direction would lead to a model cycle.

Definition 2.3.4 (Cycle Avoidance) If $X - Y$ and $X \rightarrow Z \rightarrow Y$, then orient as $X \rightarrow Y$.

The final rule, which does not have a particularly simple intuition, was introduced by Meek to prove completeness of the orientation rules [106].

Definition 2.3.5 (Meek Rule 3) If $W - X \rightarrow Z$, $W - Y \rightarrow Z$, $W - Z$, and $\langle X, Y \rangle$ are not neighbors, then orient as $W \rightarrow Z$.

At the end of Phase II, PC produces a partially directed model because not all edge orientations may be identifiable. However, given the assumptions, PC is guaranteed to have identified the Markov equivalence class. In Figure 2.5, we show the resulting model after applying these orientation rules. In this particular example, the skeleton can be completely oriented and Meek Rule 3 does not activate.



(a) After collider detection (b) After known non-colliders (c) After cycle avoidance

Figure 2.5: Resulting orientations after applying edge orientation rules during Phase II of the PC algorithm. For this example, the skeleton can be completely oriented because the Markov equivalence class consists of a single model.

2.3.2 Alternatives to the PC algorithm

The PC algorithm works remarkably well in practice, but simple implementations can lead to structural errors on finite samples. As a result, researchers have proposed many variants of PC that attempt to improve on its performance on real data. If the assumptions made by the underlying conditional independence test used by PC (e.g., linear, Gaussian data for partial correlation tests) do not match the true functional form of the generative distribution, then PC may introduce both false positives and false negatives in its learned skeleton. Combining PC with more accurate statistical tests is a common approach, such as leveraging Gaussian copulas [60] and kernel conditional independence tests [201]. These variants do not attempt to modify the general strategy employed by PC.

Aside from more accurate finite-sample tests, several researchers have proposed modifications to how PC reasons about conditional independence decisions and patterns in the skeleton. One often-cited error is based on the consistency of the skeleton and separating sets. It is possible for PC to delete an edge between X and Y via a separating set for which members may not appear on any path between X and Y. This occurrence violates the implications of d-separation on the learned structured. Steck and Tresp [175], Cheng et al. [21], and Abellán et al. [1] all use variations of the so-called *necessary path condition* to remedy these inconsistencies. The Edge-Opt al-

gorithm by Fast attempts to resolve conflicts in the resulting partially directed graph by maximizing the number of satisfied d-separation constraints [38]. Abellán et al. also propose a method for revising the skeleton by removing the weakest edge in a triple of mutually and marginally dependent variables [1].

The skeleton learned after Phase I of PC is also known to be dependent on the order of the variables tested. A simple modification introduced by Colombo and Maathhuis leads to a stable version of PC [26]. Instead of removing edges immediately following the discovery of a separating set, they show that delaying the removal until all tests of a given size are performed yields an order-independent skeleton. They also propose similar modifications for edge orientation.

The modifications described above all retain the basic strategy of PC, but other researchers have developed novel constraint-based methods that can learn more accurate structures for finite samples. These algorithms typically alter how neighbors are selected for consideration in conditioning sets or the order in which they are processed. Some examples include grow-shrink [104], total conditioning [127], recursive learning algorithms [196, 197], and light mutual min [97].

Beyond constraint-based algorithms, the *search-and-score* based paradigm has been widely used to learn Bayesian network structure. These algorithms evaluate the structure simultaneously with its parameters in some pre-defined model space. The first prominent search-and-score algorithm, called K2, requires prior knowledge of the order of variables in the network in order to learn network structure [28]. Algorithms such as K2 use likelihood measures, typically penalized by structural complexity, to heuristically navigate the search because learning the optimal Bayesian network is an NP-complete problem [22]. Typically, search-and-score algorithms use greedy hill-climbing search, but other optimizations to avoid local minima have also been explored. Search spaces other than completely oriented Bayesian networks include the space of Markov equivalence classes [23, 4, 30], a combination of skeletons and oriented

models [174], and node orderings [181]. Search-and-score algorithms that learn a Markov equivalence class, such as greedy equivalence search [23], are particularly well-suited for causal discovery.

A third approach, referred to as *hybrid* algorithms, combines elements from the constraint-based and search-and-score paradigms to improve efficiency and accuracy. Examples include CB [167], sparse candidate (SC) [44], BENEDICT [3], max-min hill climbing (MMHC) [188], and constrained optimal search (COS) [130].

Finally, many researchers have explored options for relaxing or removing the assumptions leveraged by PC. The work by Ramsey et al. decomposes the faithfulness assumption into violations with respect to the adjacency structure and orientations [137]. Assuming a faithful skeleton, the conservative PC (CPC) algorithm [137] learns a causal structure without assuming faithful orientations. Detecting violations of adjacency faithfulness was proposed in the very conservative PC (VCPC) algorithm [94], leading to networks with many edges marked as ambiguous. The generalization of Bayesian networks and d-separation to maximal ancestral graphs and m-separation has led to algorithms which remove the causal sufficiency assumption. The analog to PC for this representation is the FCI algorithm, which has been shown to be sound and complete in the presence of latent confounders [172, 198]. Finite sample improvements of FCI have been proposed in the MBCS* [128], RFCI [27], and BCCD [25] algorithms.

The RCD algorithm in Chapter 5 relies on the same assumptions as PC to prove soundness and completeness, and this dissertation leaves finite sample strategies for future work. The search-and-score and hybrid paradigms also offer interesting possibilities for an empirical comparison on real relational data sets. Extensions to RCD based on relaxing or removing assumptions is another clear direction for future research. Although not included in this dissertation, we are currently working to incorporate relational blocking [141, 139] as a new operator for structure learning

that can relax the causal sufficiency assumption (see Section 6.2 for more details). A complete extension based on m-separation would require further development of our relational representation, as well as a relational m-separation theory. In the following chapter, we formally describe the representation that enables relational d-separation and relational causal discovery.

2.4 Concluding Remarks

This chapter described three main concepts that reflect the three main contributions of this thesis. First, Bayesian networks and the assumptions leading to their causal interpretation are important because directed graphical models and those assumptions inform the relational model representation presented in Chapter 3. Second, reasoning about conditional independence facts that hold in distributions represented by a Bayesian network enable the approach taken by constraint-based algorithms that learn causal structure. To that end, the theory of d-separation is critical, and we extend its semantics and graphical criteria to the theory of relational d-separation in Chapter 4. Finally, details of the PC algorithm, and constraint-based methods in general, are necessary because our approach to structure learning follows the same strategy of identifying conditional independencies in data (or underlying distributions) and inferring causal structure from those observed independencies. Specifically, Chapter 5 introduces the RCD algorithm, which follows the two-phase procedure employed by PC and presents analogous soundness and completeness results under similar assumptions.

CHAPTER 3

REPRESENTATION

Propositional representations, such as Bayesian networks, describe domains from the perspective of a single entity type and assume that data instances are independent and identically distributed. However, many real-world systems involve multiple types of interacting entities with probabilistic dependencies that cross the boundaries of those entities. Many researchers have focused on modeling such domains, which are generally characterized as *relational*. Relational models are strictly more expressive than propositional models because they capture a large class of dependencies that are necessarily excluded from propositional models. In this chapter, we formalize concepts of relational data and models that support the theory and methods underlying the larger set of contributions in this thesis (e.g., reasoning about conditional independence in relational models and constraint-based causal structure learning from relational data).[1]

Over the past 15 years, researchers in statistics and computer science have devised expressive classes of models to capture interactions among entity types and remove the assumptions of independent and identically distributed instances [57]. Because our primary task concerns causality, we focus on directed, acyclic graphical models of relational data that encode conditional independence. Representations that model *marginal* dependencies, such as relational dependency networks [117], and *undirected* models, such as Markov logic networks (MLNs) [143], cannot have causal semantics

[1]Portions of this chapter are drawn from Maier et al. [100] with contributions from Katerina Marazopoulou.

by definition. The formal concepts of relational data and models described in this chapter are most similar to probabilistic relational models (PRMs) [85] and directed acyclic probabilistic entity-relationship (DAPER) models [63].

The main contributions presented in this chapter are:

- A sound and complete characterization of relational paths with precise semantics of their instantiation to terminal sets (Section 3.3)
- A proof of the conditions under which two distinct relational paths may have a non-empty intersection between their terminal sets (Section 3.4)
- A formalization of relational variables and dependencies as fundamental concepts to support reasoning about conditional independence and constraint-based structure learning (Section 3.5).

Beyond these contributions, we describe how the semantics of "bridge burning" for terminal sets leads to a more expressive class of models than would otherwise be induced in its absence (Section 3.7). We also provide background on four related classes of models: Bayesian networks, those that are effectively subsumed by our model class, those that are similar in expressive power, and an approach, referred to as propositionalization, that reuses existing technology (Section 3.8). Finally, we describe six limitations of our model class that present clear directions for future extensions of this thesis (Section 3.9).

3.1 Example Relational Domains and Applications

The increasing complexity, availability, quantitative nature, and sheer volume of data demand new methods for analysis. While it may be the de facto approach for causal discovery to restrict model representation to propositional domains, we argue that most real-world systems have underlying relational structure. Any domain that

involves interacting, heterogeneous entities can be characterized as relational. Some examples include:

- *Scholarly publishing* is composed of researchers at various institutions that collaborate to write articles that cite other articles and are published at different venues.
- *Epidemiology* studies individuals, their interactions, and different types of contagions and treatments.
- *Sports* include athletes, their teams and coaches, referees, and competitive interactions among players and teams.
- *Social networks* record the personal and professional interactions among individuals, companies, and events.
- *Education* research commonly investigates school districts, which encompass different grade levels, classes, teachers, and students.
- *Movie industry* data consist of interactions among movies, actors, directors, producers, studios, and critics that provide reviews.
- *Organizations* typically employ individuals, are divided into departments and business units, and develop various products.
- *Neuroscience* studies the nervous system at various levels, such as the molecular, cellular, system, and cognitive levels, where each may be composed of parts that interact within and across those levels.

All these domains are large systems that involve modular, interacting components. There are additional aspects beyond relational structure (e.g., temporal dynamics, ontological categories of entities), but we concentrate on the main difference from propositional domains: multiple entity types and the relationships among them.

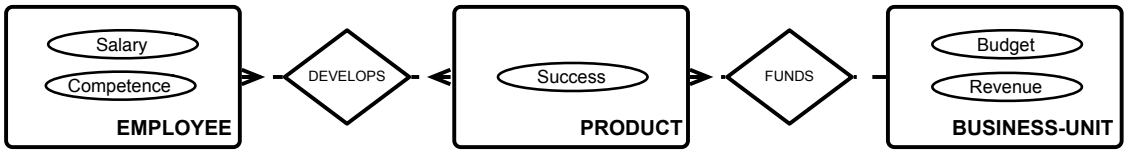


Figure 3.1: Example relational schema for an organization consisting of employees working on products, which are funded by specific business units within a corporation.

Many practical applications have also benefited from learning and reasoning with relational models. Because this thesis focuses on directed graphical models of relational data, we identify several example applications that have recently used PRMs.[2]

Probabilistic relational models have been used successfully to analyze gene regulatory interactions [158], scholarly citations [178], ecosystems [31], biological cellular networks [42], epidemiology [56], and security in information systems [170]. The structure and parameters of these models can be learned directly from a relational data set. The model is typically used either to predict values of certain attributes (e.g., topics of papers) or the structure is examined directly (e.g., to determine predictors of disease spread). A major goal in many of these applications is to promote understanding of a domain or to determine causes of various outcomes. However, as with Bayesian networks, to effectively interpret and reason about relational models *causally*, it is necessary to understand their conditional independence implications. To that effect, this chapter presents a formalization of relational concepts that support the larger objectives of reasoning about independence and learning causal structure.

3.2 Relational Schemas and Skeletons

Relational data and models are typically defined at two levels: templates and instantiations of those templates. This is an approach that we adapt for every rela-

[2]To date, PRMs are the most widely used general framework for representing directed graphical models of relational domains. While more expressive, DAPER has only been presented as a language for modeling relational data, with no practical learning algorithms developed.

tional concept. A relational schema is a top-level description of what data exist in a particular domain. Specifically (adapted from Heckerman et al. [62]):

Definition 3.2.1 (Relational schema) A *relational schema* $\mathcal{S} = (\mathcal{E}, \mathcal{R}, \mathcal{A}, card)$ consists of a set of entity classes $\mathcal{E} = \{E_1, \ldots, E_m\}$; a set of relationship classes $\mathcal{R} = \{R_1, \ldots, R_n\}$, where each $R_i = \langle E_1^i, \ldots, E_{a_i}^i \rangle$, with $E_j^i \in \mathcal{E}$ and a_i is the arity for R_i; a set of attribute classes $\mathcal{A}(I)$ for each item class $I \in \mathcal{E} \cup \mathcal{R}$; and a cardinality function $card : \mathcal{R} \times \mathcal{E} \rightarrow \{\text{ONE, MANY}\}$.

A relational schema can be represented graphically with an entity-relationship (ER) diagram. We adopt a slightly modified ER diagram using Barker's notation [8], where entity classes are rectangular boxes, relationship classes are diamonds with dashed lines connecting their associated entity classes, attribute classes are ovals residing on entity and relationship classes, and cardinalities are represented with crow's foot notation.

Example 3.2.1 The relational schema $\mathcal{S}$ for the organization domain depicted in Figure 3.1 consists of entities $\mathcal{E} = \{$EMPLOYEE, PRODUCT, BUSINESS-UNIT$\}$; relationships $\mathcal{R} = \{$DEVELOPS, FUNDS$\}$, where DEVELOPS $= \langle$EMPLOYEE, PRODUCT$\rangle$, FUNDS $= \langle$BUSINESS-UNIT, PRODUCT$\rangle$ and having cardinalities *card*(DEVELOPS, EMPLOYEE)=MANY, *card*(DEVELOPS, PRODUCT)=MANY, *card*(FUNDS, BUSINESS-UNIT)=MANY, and *card*(FUNDS, PRODUCT)=ONE; and attributes $\mathcal{A}$(EMPLOYEE)= {*Competence*, *Salary*}, $\mathcal{A}$(PRODUCT)={*Success*}, and $\mathcal{A}$(BUSINESS-UNIT)={*Budget*, *Revenue*}. □

A relational schema is a template for a relational skeleton (also referred to as a data graph by Neville and Jensen [117]), an instantiation of entity and relationship classes. Specifically (adapted from Heckerman et al. [62]):

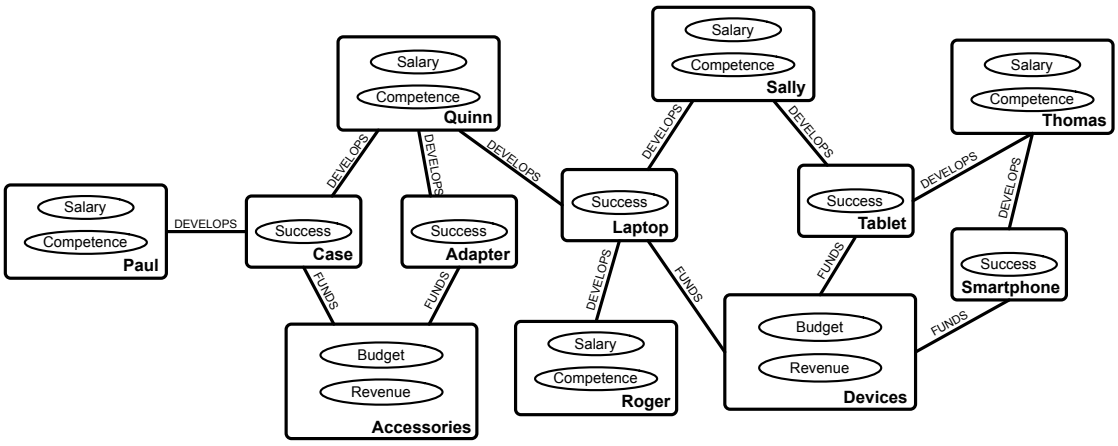


Figure 3.2: Example fragment of a relational skeleton. Roger and Sally are employees, both of whom develop the Laptop product, but of the two, only Sally works on product Tablet. Both products Laptop and Tablet are funded by business unit Devices. For convenience, we depict attribute placeholders on each entity instance.

Definition 3.2.2 (Relational skeleton) A *relational skeleton* σ for schema $\mathcal{S} = (\mathcal{E}, \mathcal{R}, \mathcal{A}, \mathit{card})$ specifies a set of entity instances $\sigma(E)$ for each $E \in \mathcal{E}$ and relationship instances $\sigma(R)$ for each $R \in \mathcal{R}$. Relationship instances adhere to the cardinality constraints of $\mathcal{S}$: If $\mathit{card}(R, E) = \text{ONE}$, then for each $e \in \sigma(E)$ there is at most one $r \in \sigma(R)$ such that e participates in r.

For convenience, we use the notation $E \in R$ if entity class E is a component of relationship class R, and, similarly, $e \in r$ if entity instance e is a component of the relationship instance r. We also denote the set of all skeletons for schema $\mathcal{S}$ as $\Sigma_{\mathcal{S}}$.

Example 3.2.2 The relational skeleton σ for the organization example is depicted in Figure 3.2. The sets of entity instances are $\sigma(\text{EMPLOYEE}) = \{$Paul, Quinn, Roger, Sally, Thomas$\}$, $\sigma(\text{PRODUCT}) = \{$Case, Adapter, Laptop, Tablet, Smartphone$\}$, and $\sigma(\text{BUSINESS-UNIT}) = \{$Accessories, Devices$\}$. The sets of relationship instances are $\sigma(\text{DEVELOPS}) = \{\langle$Paul, Case$\rangle$, $\langle$Quinn, Case$\rangle$, ..., $\langle$Thomas, Smartphone$\rangle\}$ and $\sigma(\text{FUNDS}) = \{\langle$Accessories, Case$\rangle$, $\langle$Accessories, Adapter$\rangle$, ..., $\langle$Devices, Smartphone$\rangle\}$. The relationship instances adhere to their cardinality constraints (e.g., FUNDS is a

ONE-TO-MANY relationship—within σ(FUNDS), every product has a single business unit, and every business unit may have multiple products). □

3.3 Relational Paths and Terminal Sets

In order to specify a model over a relational domain, we must define a space of possible variables and dependencies. Consider the example dependency [PRODUCT, DEVELOPS, EMPLOYEE].*Competence* → [PRODUCT].*Success* from the model in Figure 3.4, expressing that the competence of employees developing a product affects the success of that product. For relational data, the variable space includes not only intrinsic entity and relationship attributes (e.g., success of a product), but also the attributes on other entity and relationship classes that are reachable by paths along the relational schema (e.g., the competence of employees that develop a product). We define relational paths to formalize the notion of which item classes are reachable on the schema from a given item class.[3]

Definition 3.3.1 (Relational path) A *relational path* $[I_j, \ldots, I_k]$ for schema $\mathcal{S}$ is an alternating sequence of entity and relationship classes $I_j, \ldots, I_k \in \mathcal{E} \cup \mathcal{R}$ such that:

(1) For every pair of consecutive item classes $[E, R]$ or $[R, E]$ in the path, $E \in R$.

(2) For every triple of consecutive item classes $[E, R, E']$, $E \neq E'$.[4]

(3) For every triple of consecutive item classes $[R, E, R']$, if $R = R'$,
then $card(R, E) = \text{MANY}$.

I_j is called the *base item*, or *perspective*, of the relational path.

Condition (1) enforces that entity classes participate in adjacent relationship classes in the path. Conditions (2) and (3) remove any paths that would invariably reach an empty terminal set (see Definition 3.3.2 and Lemma 3.3.1 below). This

[3]Because the term "path" is also commonly used to describe chains of dependencies in graphical models, we will explicitly qualify each reference to avoid ambiguity.

definition of relational paths is similar to "meta-paths" and "relevance paths" in similarity search and information retrieval in heterogeneous networks [176, 162]. Relational paths also extend the notion of "slot chains" from the PRM framework [53] by including cardinality constraints and formally describing the semantics under which repeated item classes may appear on a path. Relational paths are also a specialization of the first-order constraints on arc classes imposed on DAPER models [62].

Example 3.3.1 Consider the relational schema in Figure 3.1. Some example relational paths from the EMPLOYEE perspective (with an intuitive meaning of what the paths describe) include the following: [EMPLOYEE] (an employee), [EMPLOYEE, DEVELOPS, PRODUCT] (products developed by an employee), [EMPLOYEE, DEVELOPS, PRODUCT, FUNDS, BUSINESS-UNIT] (business units of the products developed by an employee), and [EMPLOYEE, DEVELOPS, PRODUCT, DEVELOPS, EMPLOYEE] (coworkers developing the same products). Invalid relational paths include [EMPLOYEE, DEVELOPS, EMPLOYEE] (because EMPLOYEE=EMPLOYEE and DEVELOPS $\in \mathcal{R}$) and [BUSINESS-UNIT, FUNDS, PRODUCT, FUNDS, BUSINESS-UNIT] (because PRODUCT $\in \mathcal{E}$ and *card*(FUNDS, PRODUCT) = ONE). $\square$

Relational paths are defined at the level of relational schemas, and as such are templates for paths in a relational skeleton. An instantiated relational path produces a set of traversals on a relational skeleton. However, the quantity of interest is not the traversals, but the set of reachable item instances (i.e., entity or relationship instances). These reachable instances are the fundamental elements that support model instantiations (i.e., ground graphs).

Definition 3.3.2 (Terminal set) For skeleton $\sigma \in \Sigma_{\mathcal{S}}$ and $i_j \in \sigma(I_j)$, the *terminal set* $P|_{i_j}$ for relational path $P = [I_j, \ldots, I_k]$ of length n is defined inductively as

[4]This condition suggests at first glance that self-relationships (e.g., employees manage other employees, individuals in social networks maintain friendships, scholarly articles cite other articles) are prohibited. We discuss this and other model assumptions in Section 3.9.

$$P^1|_{i_j} = [I_j]|_{i_j} = \{i_j\}$$
$$\vdots$$
$$P^n|_{i_j} = [I_j, \ldots, I_k]|_{i_j} = \bigcup_{i_m \in P^{n-1}|_{i_j}} \Big\{ i_k \mid \big((i_m \in i_k \text{ if } I_k \in \mathcal{R}) \vee (i_k \in i_m \text{ if } I_k \in \mathcal{E})\big) \wedge i_k \notin \bigcup_{l=1}^{n-1} P^l|_{i_j} \Big\}$$

A terminal set of a relational path $P = [I_j, \ldots, I_k]$ consists of instances of class I_k, the terminal item on the path. Conceptually, a terminal set is produced by traversing a skeleton beginning at a single instance of the base item class, $i_j \in \sigma(I_j)$, following instances of the item classes in the relational path, and reaching a set of instances of class I_k. The term $i_k \notin \bigcup_{l=1}^{n-1} P^l|_{i_j}$ in the definition implies a "bridge burning" semantics under which no item instances are revisited (i_k does not appear in the terminal set of any prefix of P).[5] The notion of terminal sets is a necessary concept for grounding any relational model and has been described in previous work—e.g., for PRMs [53] and MLNs [143]—but has not been explicitly named. We emphasize their importance because terminal sets are also critical for defining relational d-separation, and we formalize the semantics for bridge burning.

Example 3.3.2 We can generate terminal sets by pairing the set of relational paths for the schema in Figure 3.1 with the relational skeleton in Figure 3.2. Let Quinn be our base item instance. Then [Employee]$|_{\text{Quinn}}$ = {Quinn}, [Employee, Develops, Product]$|_{\text{Quinn}}$ = {Case, Adapter, Laptop}, [Employee, Develops, Product, Funds, Business-Unit]$|_{\text{Quinn}}$ = {Accessories, Devices}, and [Employee, Develops, Product, Develops, Employee]$|_{\text{Quinn}}$ = {Paul, Roger, Sally}. The bridge burning semantics enforce that Quinn is not also included in this last terminal set. □

[5]The bridge burning semantics yield terminal sets that are necessarily subsets of terminal sets that would otherwise be produced without bridge burning. Although this appears to be limiting, it actually enables a strictly more expressive class of relational models. See Section 3.7 for more details and an example.

In the following lemma, we prove that Definition 3.3.1 for relational paths is sound and complete with respect to producing non-empty terminal sets for at least one relational skeleton. This effectively justifies the stated schematic conditions on relational paths.

Lemma 3.3.1 *Let $\mathcal{S}$ be a relational schema and $[I_j, \ldots, I_k]$ be a sequence of alternating entity and relationship classes of $\mathcal{S}$ that satisfy participation constraints (condition (1) of Definition 3.3.1). The relational path $[I_j, \ldots, I_k]$ satisfies conditions (2) and (3) of Definition 3.3.1 if and only if there exists a relational skeleton $\sigma \in \Sigma_{\mathcal{S}}$ and an item instance $i_j \in \sigma(I_j)$ such that $[I_j, \ldots, I_k]|_{i_j} \neq \emptyset$. More formally,*

$$\exists \sigma \in \Sigma_{\mathcal{S}}\ \exists i_j \in \sigma(I_j)\ \left([I_j, \ldots, I_k]|_{i_j} \neq \emptyset\right)$$
$$\Leftrightarrow$$
$$\left([ERE] \notin [I_j, \ldots, I_k]\right) \wedge \left([RER] \in [I_j, \ldots, I_k] \rightarrow \mathit{card}(R, E) = \text{MANY}\right)$$

Proof. Left-to-right $\Rightarrow$: Assume that there exists a skeleton $\sigma \in \Sigma_{\mathcal{S}}$ and item instance $i_j \in \sigma(I_j)$ such that $[I_j, \ldots, I_k]|_{i_j} \neq \emptyset$. We must show that $[I_j, \ldots, I_k]$ obeys conditions (2) and (3), i.e., $[I_j, \ldots, I_k]$ does not contain any $[ERE]$ patterns, and if it contains an $[RER]$ pattern, then $\mathit{card}(R, E) = \text{MANY}$.

- Assume for contradiction that $[I_j, \ldots, I_k]$ contains a pattern of the form $[ERE]$. From Definition 3.3.2 for terminal sets, it follows that if the terminal set of a path is not empty, then the terminal set of every prefix of that path is not empty:

  $$[I_j, \ldots, I_k]|_{i_j} \neq \emptyset \Rightarrow [I_j, \ldots, I_m]|_{i_j} \neq \emptyset \text{ for all } [I_j, \ldots, I_m] \leq [I_j, \ldots, I_k]$$

  By assumption, $[I_j, \ldots, I_k]|_{i_j} \neq \emptyset$; therefore, the prefix $[I_j, \ldots, I_m]$ that ends in the ERE pattern also has a non-empty terminal set:

$$[I_j, \ldots, I_k]|_{i_j} \neq \emptyset \Rightarrow [I_j, \ldots, E, R, E]|_{i_j} \neq \emptyset$$
$$[I_j, \ldots, I_k]|_{i_j} \neq \emptyset \Rightarrow [I_j, \ldots, E, R]|_{i_j} \neq \emptyset$$
$$[I_j, \ldots, I_k]|_{i_j} \neq \emptyset \Rightarrow [I_j, \ldots, E]|_{i_j} \neq \emptyset$$

Let $e \in \sigma(E)$ be an entity instance in the terminal set $[I_j, \ldots, E]|_{i_j}$. Since the terminal set $[I_j, \ldots, E, R]|_{i_j}$ is not empty, it follows that there exists a relationship instance $r = \langle \ldots, e, \ldots \rangle$ such that $r \in [I_j, \ldots, E, R]|_{i_j}$. However, $[I_j, \ldots, E, R, E]|_{i_j}$ is also not empty; thus, there exists some $e' \in \sigma(E)$ such that $e' \in [I_j, \ldots, E, R, E]|_{i_j}$, where $e' \neq e$ and $e' \in r$. It follows that both e and e' participate in the relationship instance r, which is a contradiction.

- Assume for contradiction that $[I_j, \ldots, I_k]$ contains a pattern of the form $[R, E, R]$ and $card(R, E) = \text{ONE}$.

$$[I_j, \ldots, R]|_{i_j} \neq \emptyset \Rightarrow \exists r = \langle e, \ldots \rangle \in [I_j, \ldots, R]|_{i_j}$$
$$[I_j, \ldots, R, E]|_{i_j} \neq \emptyset \Rightarrow \exists e \in [I_j, \ldots, R, E]|_{i_j} \text{ and } e \in r$$
$$[I_j, \ldots, R, E, R]|_{i_j} \neq \emptyset \Rightarrow \exists r' = \langle e, \ldots \rangle \text{ such that } r' \in [I_j, \ldots, R, E, R]|_{i_j}$$
$$\text{and } r' \neq r \text{ (bridge burning semantics)}$$

From the first and third lines above, it follows that e participates in two instances of R; therefore, $card(R, E)$ must be MANY, which is a contradiction.

Right-to-left $\Leftarrow$: Assume that $[I_j, \ldots, I_k]$ adheres to Definition 3.3.1 for relational paths. We must show that $\exists \sigma \in \Sigma_{\mathcal{S}}\ \exists i_j \in \sigma(I_j)\ \big([I_j, \ldots, I_k]|_{i_j} \neq \emptyset\big)$. We can construct such a skeleton σ according to the following procedure: For each entity class E on the path, add a unique entity instance e to $\sigma(E)$. Then, for each relationship class R on the path, add a unique relationship instance r connecting the previously created unique entity instances that participate in R, and add unique entity instances

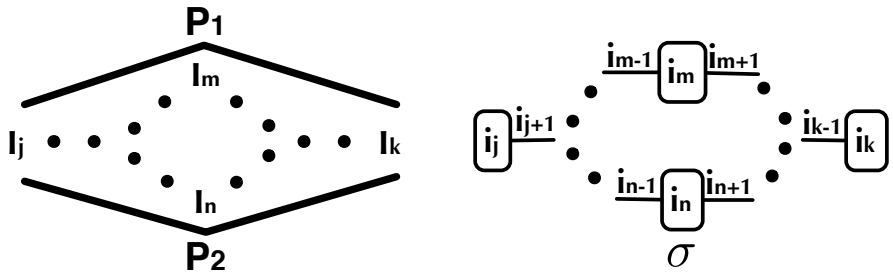


Figure 3.3: Schematic of two relational paths P_1 and P_2 for which Lemma 3.4.1 guarantees that some skeleton σ yields a non-empty intersection of their terminal sets. The example depicts a possible constructed skeleton based on the procedure used in the proof of Lemma 3.4.1.

for classes $E \in R$ not appearing on the path. This process constructs an admissible skeleton—all instances are unique and this process assumes no cardinality constraints aside from those required by Definition 3.3.1. By construction, there exists an item instance $i_j \in \sigma(I_j)$ such that $[I_j, \ldots, I_k]|_{i_j} \neq \emptyset$. ■

3.4 Intersection of Relational Paths

For a given base item class, it is common (depending on the schema) for distinct relational paths to reach the same terminal item class. The following lemma states that if two relational paths with the same base item and the same terminal item differ at some point in the path, then for some relational skeleton and some base item instance, their terminal sets will have a non-empty intersection. This property is important to consider for relational d-separation.

Lemma 3.4.1 *For two relational paths of arbitrary length from I_j to I_k that differ in at least one item class, $P_1 = [I_j, \ldots, I_m, \ldots, I_k]$ and $P_2 = [I_j, \ldots, I_n, \ldots, I_k]$ with $I_m \neq I_n$, there exists a skeleton $\sigma \in \Sigma_{\mathcal{S}}$ such that $P_1|_{i_j} \cap P_2|_{i_j} \neq \emptyset$ for some $i_j \in \sigma(I_j)$.*

Proof. Proof by construction. Let $\mathcal{S}$ be an arbitrary schema with two arbitrary relational paths $P_1 = [I_j, \ldots, I_m, \ldots, I_k]$ and $P_2 = [I_j, \ldots, I_n, \ldots, I_k]$ where $I_m \neq I_n$.

We will construct a skeleton $\sigma \in \Sigma_{\mathcal{S}}$ such that the terminal sets for item $i_j \in \sigma(I_j)$ along P_1 and P_2 have a non-empty intersection, that is, an item $i_k \in P_1|_{i_j} \cap P_2|_{i_j} \neq \emptyset$ (roughly depicted in Figure 3.3). We use the following procedure to build σ:

1. Simultaneously traverse P_1 and P_2 from I_j until the paths diverge. For each entity class $E \in \mathcal{E}$ reached, add a unique entity instance e to $\sigma(E)$.

2. Simultaneously traverse P_1 and P_2 backwards from I_k until the paths diverge. For each entity class $E \in \mathcal{E}$ reached, add a unique entity instance e to $\sigma(E)$.

3. For the divergent subpaths of both P_1 and P_2, add unique entity instances for each entity class $E \in \mathcal{E}$.

4. Repeat 1–3 for relationship classes. For each $R \in \mathcal{R}$ reached, add a unique relationship instance r connecting the entity instances from classes on P_1 and P_2, and add unique entity instances for classes $E \in R$ not appearing on P_1 and P_2.

This process constructs an admissible skeleton—all instances are unique and this process assumes no cardinality constraints aside from those required by Definition 3.3.1. By construction, there exists an item $i_j \in \sigma(I_j)$ such that $P_1|_{i_j} \cap P_2|_{i_j} = \{i_k\} \neq \emptyset$. ■

Example 3.4.1 Let P_1 = [Employee, Develops, Product, Develops, Employee, Develops, Product], the terminal sets for which yield other products developed by collaborating employees. Let P_2 = [Employee, Develops, Product, Funds, Business-Unit, Funds, Product], the terminal sets for which consist of other products funded by the business units funding products developed by a given employee. Intersection among terminal sets for these paths occurs even in the small example skeleton. In fact, the intersection of the terminal sets for P_1 and P_2 is non-empty for all employees. For example, Paul: $P_1|_{\text{Paul}}$ = {Adapter, Laptop} and $P_2|_{\text{Paul}}$ = {Adapter}; Quinn: $P_1|_{\text{Quinn}}$ = {Tablet} and $P_2|_{\text{Quinn}}$ = {Tablet, Smartphone}. □

3.5 Relational Variables, Dependencies, and Models

Given the definition for relational paths, it is simple to define relational variables and their instances.

Definition 3.5.1 (Relational variable) A *relational variable* $[I_j, \ldots, I_k].X$ consists of a relational path $[I_j, \ldots, I_k]$ and an attribute class $X \in \mathcal{A}(I_k)$.

As with relational paths, we refer to I_j as the perspective of the relational variable. Relational variables are templates for sets of random variables (see Definition 3.5.2). Sets of relational variables are the basis of relational d-separation queries, and consequently they are also the nodes of the abstract representation that answers those queries. There is an equivalent formulation in the PRM framework, although not explicitly named (they are simply denoted as attribute classes of **K**-related item classes via slot chain **K**). As they are critical to relational d-separation, we provide this concept with an explicit designation.

Example 3.5.1 Relational variables for the relational paths in Example 3.3.1 include intrinsic attributes, such as [Employee].*Competence* and [Employee].*Salary*, and also attributes on related entity classes, such as

- [Employee, Develops, Product].*Success*,
- [Employee, Develops, Product, Funds, Business-Unit].*Revenue*, and
- [Employee, Develops, Product, Develops, Employee].*Salary*. □

Definition 3.5.2 (Relational variable instance) For skeleton $\sigma \in \Sigma_{\mathcal{S}}$ and $i_j \in \sigma(I_j)$, a *relational variable instance* $[I_j, \ldots, I_k].X|_{i_j}$ for relational variable $[I_j, \ldots, I_k].X$ is the set of random variables $\{i_k.X \mid X \in \mathcal{A}(I_k) \ \wedge \ i_k \in [I_j, \ldots, I_k]|_{i_j} \ \wedge \ i_k \in \sigma(I_k)\}$.

To instantiate a relational variable $[I_j, \ldots, I_k].X$ for a specific base item instance i_j, we first find the terminal set of the underlying relational path $[I_j, \ldots, I_k]|_{i_j}$ and

then take the X attributes of the I_k item instances in that terminal set. This produces a set of random variables $i_k.X$. These random variables also correspond to nodes in the ground graph. As a notational convenience, if $\mathbf{X}$ is a set of relational variables, all from a common perspective I_j, then we say that $\mathbf{X}|_{i_j}$ for some item $i_j \in \sigma(I_j)$ is the union of all instantiations, $\{x \mid x \in X|_{i_j} \ \wedge \ X \in \mathbf{X}\}$.

Example 3.5.2 Instantiating the relational variables from Example 3.5.1 with base item instance Sally yields:

- [EMPLOYEE].*Competence*$|_{\text{Sally}}$ = {Sally.*Competence*},
- [EMPLOYEE, DEVELOPS, PRODUCT].*Success*$|_{\text{Sally}}$ = {Laptop.*Success*, Tablet.*Success*},
- [EMPLOYEE, DEVELOPS, PRODUCT, FUNDS, BUSINESS-UNIT].*Revenue*$|_{\text{Sally}}$ = {Devices.*Revenue*}, and
- [EMPLOYEE, DEVELOPS, PRODUCT, DEVELOPS, EMPLOYEE].*Salary*$|_{\text{Sally}}$ = {Quinn.*Salary*, Thomas.*Salary*}. □

Given the definitions for relational variables, we can now define relational dependencies.

Definition 3.5.3 (Relational dependency) A *relational dependency* $[I_j, \ldots, I_k].Y \rightarrow [I_j].X$ is a directed probabilistic dependence from attribute class Y to X through the relational path $[I_j, \ldots, I_k]$.

Depending on the context, $[I_j, \ldots, I_k].Y$ and $[I_j].X$ can be referred to as *treatment* and *outcome*, *cause* and *effect*, or *parent* and *child*. A relational dependency consists of two relational variables having a common perspective. The relational path of the child is restricted to a single item class, ensuring that the terminal sets consist of a single value. This is consistent with PRMs, except that we explicitly delineate dependencies

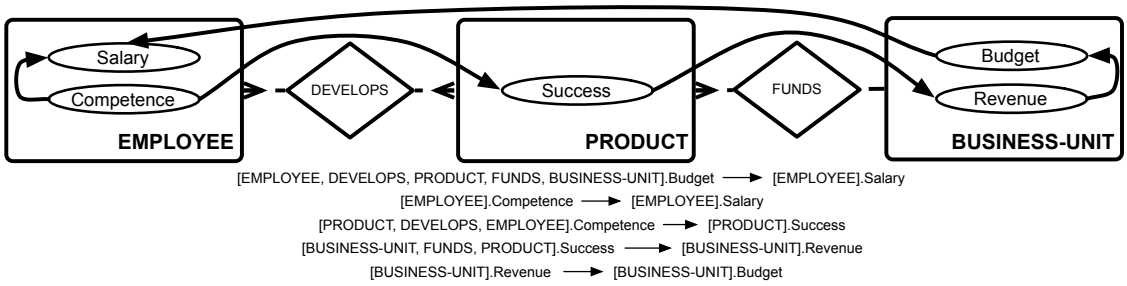


Figure 3.4: Example relational model. Employee competence causes the success of products they develop, which in turn influences the revenue received by the business unit funding the product. Additional dependencies involve the budget of business units and employee salaries. The dependencies are specified by relational paths, listed below the graphical model.

rather than define parent sets of relational variables. Note that relational variables are not nodes in a relational model, but they form the space of parent variables for relational dependencies. The relational path specification (before the attribute class of the parent) is equivalent to a slot chain, as in PRMs, or the logical constraint on a dependency, as in DAPER models.

Example 3.5.3 The dependencies in the relational model displayed in Figure 3.4 are: [Product, Develops, Employee].*Competence* → [Product].*Success* (product success is influenced by the competence of the employees developing the product), [Employee].*Competence* → [Employee].*Salary* (an employee's competence affects his or her salary), [Business-Unit, Funds, Product].*Success* → [Business-Unit].*Revenue* (the success of the products funded by a business unit influences the revenue of that unit), [Employee, Develops, Product, Funds, Business-Unit].*Budget* → [Employee].*Salary* (employee salary is governed by the budget of business units for which they develop products), and [Business-Unit].*Revenue* → [Business-Unit].*Budget* (the revenue of a business unit influences its budget). □

We now have sufficient information to define relational models.

Definition 3.5.4 (Relational model) A *relational model* $\mathcal{M}_\Theta$ has two parts:

1. The *structure* $\mathcal{M} = (\mathcal{S}, \mathcal{D})$: a schema $\mathcal{S}$ paired with a set of relational dependencies $\mathcal{D}$ defined over $\mathcal{S}$.

2. The *parameters* Θ: a conditional probability distribution

$$P\big([I_j].X \mid \mathit{parents}([I_j].X)\big)$$

for each relational variable of the form $[I_j].X$, where $I_j \in \mathcal{E} \cup \mathcal{R}$, $X \in \mathcal{A}(I_j)$ and $\mathit{parents}\big([I_j].X\big) = \big\{[I_j, \ldots, I_k].Y \mid [I_j, \ldots, I_k].Y \to [I_j].X \in \mathcal{D}\big\}$ is the set of parent relational variables.

The structure of a relational model can be represented graphically by superimposing dependencies on the ER diagram of a relational schema (see Figure 3.4 for an example). A relational dependency of the form $[I_j, \ldots, I_k].Y \to [I_j].X$ is depicted as a directed arrow from attribute class Y to X with the specification listed separately. Note that the subset of relational variables with singleton paths $[I].X$ in the definition corresponds to the set of attribute classes in the schema.

A common technique in relational learning is to use *aggregation functions* to transform parent multi-sets to single values within the conditional probability distributions. Typically, aggregation functions are simple, such as `mean` or `mode`, but they can be complex, such as those based on vector distance or object identifiers, as in the ACORA system [129]. Complementary to aggregation, we can also define *combining rules*, such as `noisy-or`, as a method to combine multiple dependencies into a single conditional probability distribution [113]. We omit both aggregation functions and combining rules from our model specification by assuming they are internal to the definition of conditional probability distributions.

This definition of relational models is consistent with and yields structures expressible as DAPER models [62]. These relational models are also equivalent to

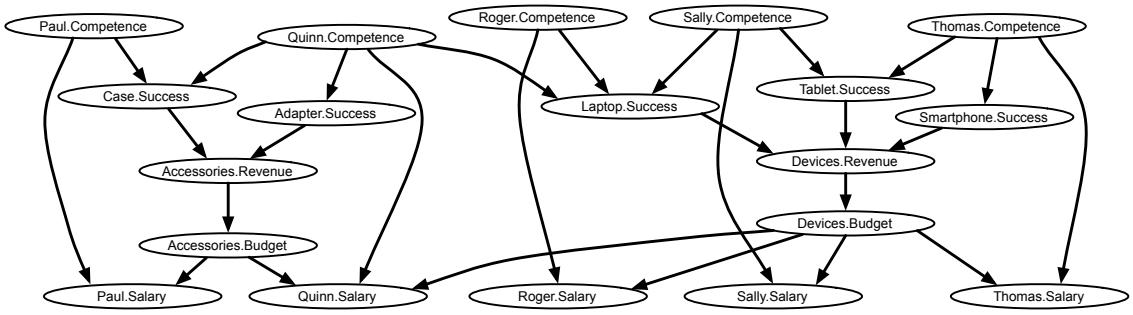


Figure 3.5: Example fragment of a ground graph. The success of product Laptop is influenced by the competence of Roger, Sally, and Quinn. The revenue of business unit Devices is caused by the success of all its funded products—Laptop, Tablet, and Smartphone.

PRMs, but we extend slot chains as relational paths and provide a formal semantics for their instantiation. These models are also more general than plate models because dependencies can be specified with arbitrary relational paths as opposed to simple intersections among plates [17, 58].

3.6 Ground Graphs

Just as the relational schema is a template for skeletons, the structure of a relational model can be viewed as a template for ground graphs: dependencies applied to skeletons.

Definition 3.6.1 (Ground graph) The *ground graph* $GG_{\mathcal{M}\sigma} = (V, E)$ for relational model structure $\mathcal{M} = (\mathcal{S}, \mathcal{D})$ and skeleton $\sigma \in \Sigma_{\mathcal{S}}$ is a directed graph with nodes $V = \{i.X \mid I \in \mathcal{E} \cup \mathcal{R} \ \wedge \ X \in \mathcal{A}(I) \ \wedge \ i \in \sigma(I)\}$ and edges $E = \{i_k.Y \rightarrow i_j.X \mid i_k.Y, i_j.X \in V \ \wedge \ i_k.Y \in [I_j, \ldots, I_k].Y|_{i_j} \ \wedge \ [I_j, \ldots, I_k].Y \rightarrow [I_j].X \in \mathcal{D}\}$.

A ground graph is a directed graph with (1) a node (random variable) for each attribute of every entity and relationship instance in a skeleton and (2) an edge from $i_k.Y$ to $i_j.X$ if they belong to the parent and child relational variable instances, respectively, of some dependency in the model. The concept of a ground graph

appears for any type of relational model, graphical or logic-based. For example, PRMs produce "ground Bayesian networks" that are structurally equivalent to ground graphs, and Markov logic networks yield ground Markov networks by applying all formulas to a set of constants [143]. The example ground graph shown in Figure 3.5 is the result of applying the dependencies in the relational model shown in Figure 3.4 to the skeleton in Figure 3.2.

Similar to Bayesian networks, given the parameters of a relational model, a *parameterized ground graph* can express a joint distribution that factors as a product of the conditional distributions:

$$P(GG_{\mathcal{M}_\Theta\sigma}) = \prod_{I\in\mathcal{E}\cup\mathcal{R}} \prod_{X\in\mathcal{A}(I)} \prod_{i\in\sigma(I)} P\big(i.X \mid \mathit{parents}(i.X)\big)$$

where each $i.X$ is assigned the conditional distribution defined for $[I].X$ (a process referred to as parameter-tying).

Relational models only define coherent joint probability distributions if they produce acyclic ground graphs. A useful construct for checking model acyclicity is the class dependency graph [53], defined as:

Definition 3.6.2 (Class dependency graph) The class dependency graph $G_\mathcal{M} = (V, E)$ for relational model structure $\mathcal{M} = (\mathcal{S}, \mathcal{D})$ is a directed graph with a node for each attribute of every item class $V = \big\{I.X \mid I \in \mathcal{E}\cup\mathcal{R} \ \wedge \ X \in \mathcal{A}(I)\big\}$ and edges between pairs of attributes supported by relational dependencies in the model $E = \big\{I_k.Y \to I_j.X \mid [I_j,\ldots,I_k].Y \to [I_j].X \in \mathcal{D}\big\}$.

If the relational dependencies form an acyclic class dependency graph, then every possible ground graph of that model is acyclic as well [53]. Given an acyclic relational model, the ground graph has the same semantics as a Bayesian network [52, 62]. All future references to acyclic relational models refer to relational models whose structure forms acyclic class dependency graphs.

By Lemma 3.4.1 and Definition 3.6.1, one relational dependency may imply dependence between the instances of many relational variables. If there is an edge from $i_k.Y$ to $i_j.X$ in the ground graph, then there is an implied dependency between all relational variables for which $i_k.Y$ and $i_j.X$ are elements of their instances.

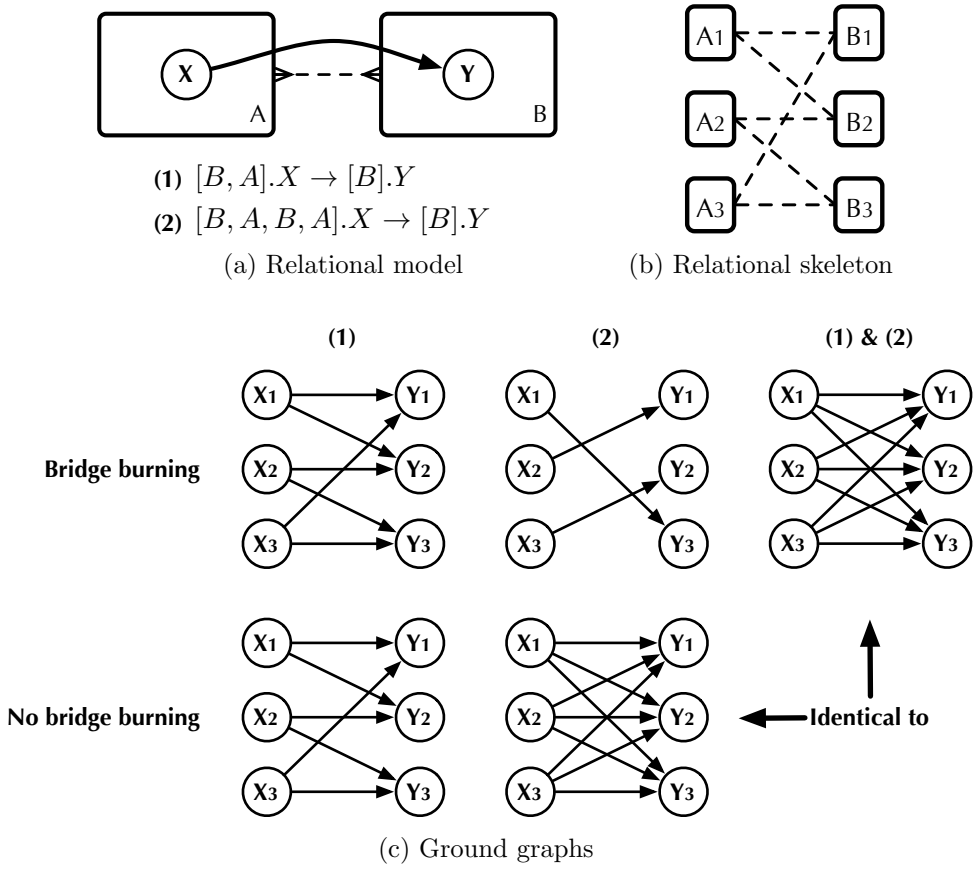
Example 3.6.1 The dependency [EMPLOYEE].*Competence* → [EMPLOYEE].*Salary* yields Roger.*Competence* → Roger.*Salary* in the ground graph of Figure 3.5 because Roger.*Competence* ∈ [EMPLOYEE].$Competence|_{\text{Roger}}$. However, Roger.*Competence* ∈ [EMPLOYEE, DEVELOPS, PRODUCT, DEVELOPS, EMPLOYEE].$Competence|_{\text{Sally}}$ (as is Roger.*Salary*, replacing *Competence* with *Salary*). Consequently, the relational dependency *implies* dependence among the random variables in the instances of [EMPLOYEE, DEVELOPS, PRODUCT, DEVELOPS, EMPLOYEE].*Competence* and [EMPLOYEE, DEVELOPS, PRODUCT, DEVELOPS, EMPLOYEE].*Salary*. □

These implied dependencies form the crux of the challenge of identifying independence in relational models. Additionally, the intersection between the terminal sets of two relational paths is crucial for reasoning about independence because a random variable can belong to the instances of more than one relational variable. Since d-separation only guarantees independence when there are no d-connecting paths, we must consider all possible paths between pairs of random variables, either of which may be a member of multiple relational variable instances. In Chapter 4, we define relational d-separation and provide an appropriate representation, the abstract ground graph, that enables straightforward reasoning about d-separation.

3.7 Bridge Burning Semantics

In this section, we provide an example to show that the bridge burning semantics for terminal sets of relational paths yields a strictly more expressive class of relational models than semantics without bridge burning. The bridge burning semantics

produces terminal sets that are necessarily *subsets* of terminal sets which would otherwise be produced without bridge burning. Paradoxically, this enables a *superset* of relational models.



(a) Relational model (b) Relational skeleton

(c) Ground graphs

Figure 3.6: Example demonstrating that bridge burning semantics yields a more expressive class of models than semantics without bridge burning. (a) Relational model over a schema with two entity classes and two attributes with two possible relational dependencies (relationship class omitted for simplicity). (b) Simple relational skeleton with three A and three B instances. (c) Bridge burning semantics yields three possible ground graphs with combinations of dependencies (1) and (2), whereas no bridge burning yields two possible ground graphs. The bridge burning ground graphs subsume the ground graphs without bridge burning.

Recall Definition 3.3.2 for the terminal set for a relational path. The final condition in the inductive definition ($i_k \notin [I_1, \ldots, I_j]|_{i_1}$ for $j = 1$ to $k - 1$) encodes bridge burning. The item i_k is only added to the terminal set if it is not a member of

the terminal set of any previous subpath. For example, let P be the relational path [Employee, Develops, Product, Develops, Employee]. This relational path produces terminal sets that include the employees that work on the same products (that is, co-workers). Instantiating this path with the employee Quinn, $P|_{\text{Quinn}}$, produces the terminal set {Paul, Roger, Sally}. Since Quinn $\in$ [Employee]$|_{\text{Quinn}}$, the bridge burning semantics excludes Quinn from this set. This makes intuitive sense as well—Quinn should not be considered her own colleague.

A relational model is simply a collection of relational dependencies. Each relational dependency is primarily described by the relational path of the parent relational variable (because, for canonically specified dependencies, the relational path of the child consists of a single item class). The relational path specification is used in the construction of ground graphs, connecting variable instances that appear in the terminal sets of the parent and child relational variables.

To characterize the expressiveness of relational models, we can inspect the space of representable ground graphs by choosing an arbitrary relational skeleton and a small set of relational dependencies. We show with a simple example that the bridge burning semantics for a model over a two-entity, bivariate schema yields more possible ground graphs than without bridge burning. (We omit the relationship class for simplicity.) In Figure 3.6(a), we present such a model with two possible relational dependencies labeled (1) and (2). Figure 3.6(b) provides a simple relational skeleton involving three A and three B instances (relationship instances are represented as dashed lines for simplicity). As shown in Figure 3.6(c), the bridge burning semantics leads to three possible ground graphs, one for each combination of the dependencies (1), (2), and both (1) and (2) together. Without bridge burning, only two ground graphs are possible because dependency (2) completely subsumes dependency (1) with those semantics.

This example generalizes to arbitrary dependencies. The terminal sets of relational paths that repeat item classes subsume subpaths under the semantics without bridge burning. This leads to fewer possible relational models, which justifies our choice of semantics for terminal sets of relational paths.

3.8 Related Representations

To compare the class of relational models considered in this chapter, we examine four sets of related representations. First, we revisit Bayesian networks and describe them as a proper subset of relational models. The second set of model classes consists of those that are effectively subsumed in their expressive power. The third set of model classes consists of those belonging to the statistical relational learning community, with similar expressive power to our model class. Finally, we describe an approach that reduces the expressiveness of a relational representation down to a propositional setting via a process referred to as flattening or propositionalization.

3.8.1 Bayesian networks

Relational representations are strictly more expressive than the propositional representation used in Bayesian network modeling [120].[6] As described in Section 2.1, Bayesian networks form the basis for much of the fundamental research on causal reasoning and discovery [122, 172]. Bayesian networks describe domains from the perspective of a single entity class; thus, they can only produce schemas with $|\mathcal{E}| = 1$ (one entity class) and $|\mathcal{R}| = 0$ (no relationship classes). The variables on the entity, however, can include propositionalized variables from the perspective of the given entity, as we describe below in Section 3.8.4.

[6]This reflects traditional Bayesian networks that model propositional data. Other generalizations of Bayesian networks, such as dynamic Bayesian networks, can model sequential and temporal data for which the IID assumption does not hold (e.g., hidden Markov networks and two-slice temporal Bayesian networks) [112].

The relational skeleton of a Bayesian network consists of a set of disconnected entity instances, all drawn from the same entity class. Consequently, the skeleton has a simple one-to-one mapping with the representation as a table: Each entity instance corresponds to a single row, and each variable is a column. In the organization example with $\mathcal{E} = \{\text{EMPLOYEE}\}$, each employee would be an entity instance, and no instances of other entity types or relationships would appear in the skeleton. Because all variables in a Bayesian network are defined for a single entity class with no relationships, the relational path specification becomes trivial and, hence, implicit. All relational paths, relational variables, and relational dependencies are defined from a single perspective with singleton paths (e.g., [EMPLOYEE]). The ground graph of a Bayesian network, similar to the skeleton, has a very regular structure. The ground graph consists of a set of identical copies of the model structure, one for each instance in the skeleton. For a Bayesian network, d-separation can be applied directly to the model structure because there is no variability in its ground graphs (in contrast to relational models, as we show in Chapter 4).

3.8.2 Subsumed model representations

This thesis focuses on a broad class of models that generalizes various other approaches, mostly drawn from statistics, to modeling conditional and joint distributions over data. The expressive power of these other models is mostly subsumed by the relational models we consider.

The primary advantage of multilevel, hierarchical, and random effects models is to relax this assumption of a single entity class [51, 67, 169]. Most applications of multilevel models are usually restricted to simple relational models that involve a hierarchy of types. For example, an education domain may consist of districts, schools, classrooms, and students. A multilevel model can include variables of entities higher in the hierarchy that may have effects on variables of entities lower down (e.g., school-

level variables, such as teacher quality and academic climate, correlate with student-level variables, such as achievement and demographics, as presented in the study by Lee and Bryk [93]). In contrast to Bayesian networks and relational models, the focus of multilevel models is to model a *conditional* probability distribution rather than a joint distribution. These models are commonly restricted to linear parameterizations, but they can be extended to non-linear functional forms.

There has also been growing attention to models that relax the stable unit-treatment value assumption (SUTVA), incorporating interference or spillover effects [148, 70, 179, 15, 103, 186]. These models relax the assumption of instance independence in domains where it is known that the treatment of one instance may affect the outcome of another—a scenario typically prohibited in the potential-outcome framework—to estimate an unbiased average treatment effect [150, 65]. For example, in epidemiology studies, vaccinating one individual may have indirect effects on unvaccinated individuals that come in contact with the vaccinated individual. [190] Many applications involving interference models focus on the conditional distribution of outcome given treatment, covariates, and the treatment on other instances that may affect a particular unit. These types of models are relational in that they consist of a single entity class and, usually, a single self-relationship (e.g., social networks of individuals).

Finally, there are explicit models of network data, such as the p_1 model [66] and p^* or exponential random graph models (ERGMs) [41, 147]. These are generally models of a unipartite graph with a single entity type and one relationship type. The central aim of these models is to capture the evolution of network structure, modeling the existence of links between nodes. ERGMs are useful for describing models that can produce certain network properties, such as degree distributions and clustering coefficients. However, they are now commonly extended to model directed graphs, bipartite networks, homophily of node attributes, and various dependence structures.

The causal interpretation of existence is still in flux (see Section 3.9), which is the primary reason why our class of models does not currently include dependencies involving existence.

Each of these models assumes a fixed relational schema, whereas for relational models in general, the goal is to represent arbitrary probabilistic dependencies among variables of heterogeneous, interacting entities. In the following section, we discuss various alternatives that provide similar expressive power.

3.8.3 Similar model representations

Many researchers have focused on modeling relational domains, and consequently, there are numerous alternative representations. This thesis focuses on directed, acyclic, graphical models for two main reasons: (1) to facilitate an extension to the graphical criterion of d-separation in order to reason about conditional independence and (2) to represent and learn causal structure. We have already stated that our model class is most similar to PRMs and DAPER models. In this section, we describe models with similar expressive power.

Probabilistic first-order logic models can express the model class we described in the preceding sections. However, we found it simpler to define and prove relevant theoretical properties for relational d-separation in a representation most similar to Bayesian networks (see Chapter 4). Examples of first-order logic models include Bayesian logic programs [81], parametrized Bayesian networks [134], Bayesian logic [109], multi-entity Bayesian networks [89], and relational probability models [151]. The results in this thesis could feasibly generalize to these representations.

Although undirected models can encode the same set of conditional independencies as a directed model (through a process known as "moralization"), they are unable to represent causal dependencies, which are directed by defintion. Therefore, we exclude models such as relational Markov networks [177] and Markov logic networks [143] from

our consideration. The results on relational d-separation could be transferred to these representations, but algorithms for learning causal structure are not applicable.

There are certainly useful aspects to these representations (otherwise there would not be sustained research in this area). As we describe in Section 3.9, reasonable future directions include adapting some of these features for reasoning and learning. For a more complete survey that contrasts the subtle differences among these relational representations, see the taxonomy presented by Milch and Russell [110].

3.8.4 Propositionalizing relational data

Most techniques in classical statistics and machine learning operate over a single data table, often referred to as *propositional* data. There have been two main, competing approaches to model the relational data described in this chapter: (1) develop new algorithms to handle the increased representational complexity or (2) transform the data into a single table and rely on existing algorithms. The latter approach involves a process called *propositionalization*, which flattens relational data into a propositional representation [87].

Propositionalization is the process of projecting a set of tables (typically one for each entity and relationship type) down to a single data table. This procedure is defined with respect to a single *perspective*—one of the original entity or relationship types—from which the other tables are summarized. The additional constructed features are meant to capture relational aspects of the data, placing the complexity on transformation rather than on learning. Propositionalization creates all variables ahead of time, decoupling feature construction from learning, as opposed to dynamically generating variables as in most relational learning algorithms.

There are many different approaches and systems for propositionalization, but they can be generally divided into three classes: (1) those constructing Boolean clauses over related concepts as typically used in the inductive logic programming

```
SELECT  t1.id, t1.salary, t2.success, t3.revenue
FROM     (  SELECT  E.id, E.salary
            FROM    Employee E) t1,
         (  SELECT  E.id, P.success
            FROM    Employee E, Develops D, Product P
            WHERE   E.id = D.e_id AND D.p_id = P.id) t2,
         (  SELECT  E.id, B.revenue
            FROM    Employee E, Develops D, Product P
                    Funds F, Business-Unit B
            WHERE   E.id = D.e_id AND D.p_id = P.id AND
                    P.id = F.p_id AND F.b_id = B.id) t3
WHERE   t1.id = t2.id AND t2.id = t3.id
```

Figure 3.7: Sketch of a relational database query that joins the instances of three relational variables having the common perspective EMPLOYEE used to produce the data instances shown in Table 3.1. The three relational variables are (1) [EMPLOYEE].*Salary*, (2) [EMPLOYEE, DEVELOPS, PRODUCT].*Success*, and (3) [EMPLOYEE, DEVELOPS, PRODUCT, FUNDS, BUSINESS-UNIT].*Revenue*.

(ILP) community (e.g., LINUS [91], RSD [92]), (2) those using database aggregations over sets of related values as used in the knowledge discovery in databases (KDD) community (e.g., ACORA [129]), and (3) those characterized by both (e.g., RELAGGS [88]). Rattigan [139] formalizes propositionalization as a graph sampling procedure operating over ground graphs.

For the organization domain example, consider data about only employees ($\mathcal{E}$ = {EMPLOYEE}). Variables would include intrinsic attributes, such as salary, but could also include variables describing other related entities, all from the employee perspective. That is, we could construct a single table for employees that includes columns for the success of developed products, the revenue of all business units they work under, etc. In Figure 3.7, we show an example SQL-like query that would produce such data, and the resulting data set applied to the example in Figure 3.2 is shown in Table 3.1.[7]

[7]Modeling propositionalized data with Bayesian networks still requires the IID assumption, which is often violated since variables of one instance can influence variables of another. For example, ac-

Employee	[Employee].*Salary*	[Employee, Develops, Product].*Success*	[Employee, Develops, Product, Funds, Business-Unit].*Revenue*
Paul	{Paul.*Salary*}	{Case.*Success*}	{Accessories.*Revenue*}
Quinn	{Quinn.*Salary*}	{Case.*Success*, Adapter.*Success*, Laptop.*Success*}	{Accessories.*Revenue*, Devices.*Revenue*}
Roger	{Roger.*Salary*}	{Laptop.*Success*}	{Devices.*Revenue*}
Sally	{Sally.*Salary*}	{Laptop.*Success*, Tablet.*Success*}	{Devices.*Revenue*}
Thomas	{Thomas.*Salary*}	{Tablet.*Success*, Smartphone.*Success*}	{Devices.*Revenue*}

Table 3.1: Propositional table consisting of employees, their salaries, the success of products they develop, and the revenue of the business units they operate under. Producing this table requires joining the instances of three relational variables, all from a common perspective—Employee.

Historically, there has been a debate surrounding the efficacy of propositionalization. The disadvantages focus on statistical concerns, including the prevalence of autocorrelation [76, 80] and degree disparity bias [77], both of which can increase Type I errors. (Alternative tests that avoid such bias are discussed by Rattigan and Jensen [140].) The advantages center around not needing to develop new algorithms given the vast number of deployed propositional learning algorithms. Flattening data invariably leads to some loss of the relational information, but it is generally assumed that predictive accuracy—a common measure for tasks involving propositionalization—achieves equivalent levels of performance. However, the accuracy varies depending on the propositionalization approach.

While statistical issues can also lead to serious causal concerns (e.g., statistical conclusion validity [160]), there are other potential problems with propositionalization for causal discovery rather than prediction. In a prior paper, we outline four such

cording to the model in Figure 3.4, the competence of collaborating employees influences the success of products, which affects the revenue of business units, which affects its budget, thereby influencing an employee's salary. As a result, modeling relational data with a propositional representation may unnecessarily lose valuable information, especially in the context of causal reasoning and accurate estimation of causal effects.

issues that can occur when reasoning about causality in the absence of statistical errors [98]. These problems include (1) unnecessarily creating latent variables and violating causal sufficiency, (2) inducing selection bias via relationship existence, (3) violating the causal Markov condition by deriving multiple variables from the same underlying relational variable, and (4) not specifying certain acyclicity constraints.

3.9 Model Assumptions and Related Work

The class of relational models considered in this thesis, while strictly more expressive than Bayesian networks, has limitations in its current formalization. In this section, we highlight these assumptions and discuss how related and future work could address them.

3.9.1 Self-relationships

Self-relationships are relationship classes that involve the same entity class more than once. Relational schemas, as defined in Definition 3.2.1, can express these types of relationships. Only the definition of relational paths—which govern the space of variables and dependencies—requires unique entity class names within [E,R,E] triples (see condition (2) of Definition 3.3.1). However, a common procedure in entity-relationship modeling is to map entity names to unique *role indicators* within the context of a self-relationship, such as manager/subordinate, friend1/friend2, or citing-paper/cited-paper [136]. This approach does not duplicate entity instances in the skeleton or ground graph; it only modifies their reference names within the relational path, requiring extended semantics for terminal sets. Incorporating self-relationships is a straightforward extension, but for simplicity, we omit this additional layer of complexity.

3.9.2 Relational autocorrelation

In contrast to self-relationships, relational autocorrelation is a statistical dependency among the values of the same attribute class frequently found in relational data sets [76]. Various models and learning algorithms have been developed to capture these types of dependencies, such as RDNs [117], PBNs with an extended normal form [156], and PRMs with dependencies that follow guaranteed acyclic relationships [53]. Our formalism, and equivalently PRMs (without guaranteed acyclic relationships), can represent a class of models for apparent autocorrelation. Any relational dependency that yields a common cause for grounded variables of the same attribute class—essentially any dependency that crosses a MANY cardinality—produces relational autocorrelation. The only autocorrelations not accounted for involve latent causes or those produced by temporal processes (e.g., feedback).

3.9.3 Context-specific independence

Context-specific independence (CSI) introduces independence of some variable and its parents, depending on the values of other variables. This can be achieved within the specification of conditional probability distributions as if-then-else statements of logical conditions, such as in DAPER models [62] or RPMs [151], encoded as regularities in conditional probability tables [14], or represented with the recent graphical convention of gates [111]. However, this introduces a notion of independence that cannot be inferred from model structure via traditional d-separation. Thus far, there have been two extensions to d-separation that support reasoning about CSIs: (1) Boutilier et al. [14] use d-separation on a manipulated Bayesian network by deleting certain dependencies given some context, and (2) Winn [193] adds rules to d-separation to reason over additional paths that are introduced by gates. An alternative and more general approach to encoding CSIs is to develop an *ontology* for which (in)dependencies hold depending on the type of entity or relationship. For

example, PRMs with class hierarchies allow a hierarchy of entity types where the dependency structure can vary depending on the type [55]. Rules of inheritance derived from object-oriented programming are used to define a coherent joint probability distribution. This aligns with our formalism, as relational schemas can be viewed as an ontology defined at a particular level. However, the semantics of d-separation under inheritance has not been developed and is a profitable direction of future research.

3.9.4 Causes of entity and relationship existence

Without a generative model of relational skeletons, the relational models are not truly generative as the skeleton must be generated prior to the attributes. However, the same issue occurs for Bayesian networks: Relational skeletons consist of disconnected entity instances, but the model does not specify how many instances to create. Some relational model classes do attempt to learn and represent unknown numbers of entity instances, such as BLOG [109], or uncertain relationship instances, such as PRMs with existence uncertainty [54]. However, reasoning about the connection between conditional independence and existence is an open problem. For relationship existence, selection bias (conditioning) occurs when testing marginal dependence between variables across a particular relationship [102]. For entity existence, some researchers argue that existence cannot be represented as a variable or predicate [135], while others represent them as predicates [89]. Therefore, we currently choose simple processes for generating skeletons, allowing us to focus on relational models of attributes and leaving structural causes and effects as future work.

3.9.5 Causal sufficiency

The relational models we consider assume that all common causes of observed variables are also observed and included in the model—an assumption commonly referred to as causal sufficiency. Many researchers have developed methods for learning and inference by explicitly modeling unobserved variables—typically termed latent

variable models [11]—or inferring the presence of latent entity classes—for example, latent group models [116]. However, only ancestral graphs, acyclic directed mixed graphs (ADMGs), and summary graphs do so in order to preserve an underlying conditional independence structure [144, 146, 192]. These models are paired with the theory of m-separation, which is a generalization of d-separation for Bayesian networks. The generalization of ancestral graphs, ADMGs, or summary graphs to relational models requires extensive theoretical exploration; therefore, we leave this as an important direction for future work. Given that a primary motivation for d-separation is to support constraint-based causal discovery, any relational extension to algorithms that learn causal models without assuming causal sufficiency, such as FCI [173, 198] and its variants [24, 27], MBCS* [128], and BCCD [25], would require such an extension to m-separation.

3.9.6 Temporal and cyclic models

Currently, the relational model is assumed to be acyclic (with respect to the class dependency graph), and consequently, atemporal. Model-level cycles typically result from temporal processes for which grounding across time would yield an acyclic ground graph, such as in dynamic Bayesian networks [36, 112]. However, cycles can also be due to temporal processes where the interaction occurs at a faster rate than measurement. As a result, there has been considerable attention devoted to models that explicitly encode cyclic dependencies, such as the work by Spirtes [171], Pearl and Dechter [126], Richardson [145], Dash [33], Schmidt and Murphy [155], and Hyttinen et al. [72]. Our formalism currently prohibits any relational dependency that has a common attribute class for the cause and effect, regardless of the relational path constraint. Relaxing this assumption would require either explicitly modeling temporal dynamics or enabling feedback loops. We reserve temporal dynamics and feedback as another important avenue for future research.

3.10 Concluding Remarks

Despite the limitations described in Section 3.9, we characterized a highly expressive class of models in this chapter. The relational models we consider strictly generalize Bayesian networks to capture multiple types of entities, the relationships among them, and the probabilistic dependencies among their attributes. This model class is suitable for representing relational causal dependencies under the same assumptions used to interpret Bayesian networks causally.

While this relational model class is very similar to PRMs, we formalized certain aspects that will be useful in the subsequent chapters on reasoning and learning. Aside from modifying the syntax of slot chains to relational paths (and proving their sound and complete characterization), we developed a formal semantics for instantiating relational paths and relational variables. We introduced new terms for their instantiated sets, as these sets are critical for reasoning about instance-level paths of dependence and for proving theoretical properties of our approach in the following chapters. We also showed the conditions under which intersections may occur between pairs of relational variables and showed that our bridge burning semantics introduces a more expressive space of models than its absence.

The next chapter presents a graphical and algorithmic approach for deriving conditional independencies from relational models analogous to traditional d-separation for Bayesian networks. Even though the current chapter is classified as "representation," our approach to solving relational d-separation in the next chapter relies on a new representation called the abstract ground graph.

CHAPTER 4

REASONING

The ability to reason about which conditional independencies hold for a given model is fundamental to identifying causal effects and supporting constraint-based causal discovery algorithms. As described in Section 2.2.2, the theory of d-separation provides graphical rules that can algorithmically derive all conditional independence facts that hold in distributions represented by a Bayesian network. In this chapter, we show that d-separation may not correctly infer conditional independence when applied directly to the graphical structure of a relational model.[1]

We introduce the notion of *relational d-separation*—a graphical criterion for deriving conditional independence facts from relational models—and define its semantics to be consistent with traditional d-separation (i.e., it claims independence only when it is guaranteed to hold for all model instantiations). We present an alternative, lifted representation—the *abstract ground graph*—that enables an algorithm for deriving conditional independence facts from relational models. We show that this approach is sound, complete, and computationally efficient, and we provide an empirical demonstration of effectiveness across synthetic causal structures of relational domains.

The main contributions presented in this chapter are:

- A formal definition of d-separation for relational models that is analogous to d-separation for Bayesian networks (Section 4.3)

[1]Portions of this chapter are drawn from Maier et al. [100] with contributions from Katerina Marazopoulou.

- The abstract ground graph—a lifted representation that *abstracts* all possible ground graphs of a given relational model structure, as well as proofs of the soundness and completeness of this abstraction (Section 4.4)

- Proofs of soundness and completeness for a method that answers relational d-separation queries in the limit (Section 4.5), as well as for finite bounds on the length of relational paths (Section 4.6)

We also provide an empirical comparison of relational d-separation to traditional d-separation applied directly to relational model structure, showing that, not only would most queries be undefined, but those that can be represented yield an incorrect judgment of conditional independence up to 50% of the time (Section 4.7). Finally, we offer additional empirical results on synthetic data that demonstrate the effectiveness of relational d-separation with respect to complexity and consistency (Section 4.8). In Chapter 5, we introduce a sound and complete algorithm for learning causal structure from relational data that is supported entirely by the theoretical foundation presented herein.

4.1 Example

Consider a corporate analyst who was hired to identify which employees are effective and productive for some organization. If the company is structured as a pure project-based organization (for which company personnel are structured around projects, not departments), the analyst may collect data as described by the relational schema underlying the model in Figure 4.1. The schema denotes that employees can collaborate and work on multiple products, each of which is funded by a specific business unit. The analyst has also obtained attributes on each entity—competence of employees, the success of each product, and the revenue of business units. The relational skeleton in Figure 3.2, which describes an organization with five employees, five products, and two business units, is consistent with this schema.

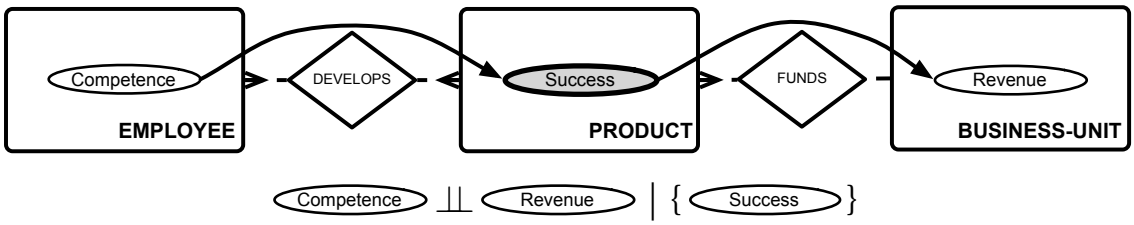


Figure 4.1: A simple relational model for the organization domain that produces an incorrect judgment about conditional independence when d-separation is applied to the model structure. This model naïvely claims that employee competence is independent of business unit revenue conditioned on product success.

Assume that the organization operates under the model depicted in Figure 4.1 (a subset of the dependencies of the model in Figure 3.4). Under this model, the success of a product depends on the competence of employees that develop it, and the revenue of a business unit is influenced by the success of products that it funds. If this were known by the analyst (who happens to have experience in graphical models), then it would be conceivable to spot-check the model and test whether some of the conditional independencies encoded by the model are reflected in the data. The analyst thus naïvely applies d-separation to the model structure in an attempt to derive conditional independencies to test. However, applying d-separation directly to the structure of relational models may not correctly derive conditional independencies, violating the Markov condition. If the analyst were to discover significant and substantive effects, he may believe the model structure is incorrect and needlessly search for alternative dependencies to explain these effects.

Naïvely applying d-separation to the model in Figure 4.1 suggests that employee competence is conditionally independent of the revenue of business units given the success of products:

$$\textsc{Employee}.\mathit{Competence} \perp\!\!\!\perp \textsc{Business-Unit}.\mathit{Revenue} \mid \textsc{Product}.\text{Success}$$

To see why this approach is flawed, we must consider the *ground graph*. A necessary precondition for inference is to apply a model to a data instantiation, which yields

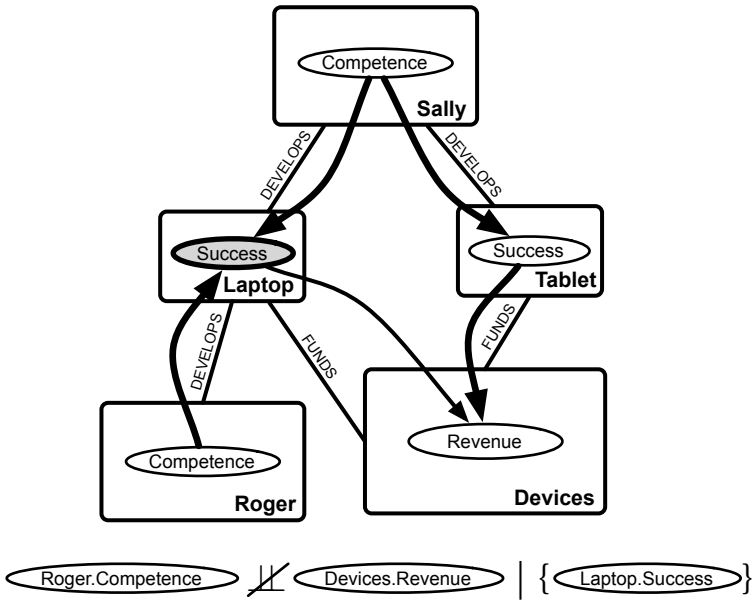


Figure 4.2: A fragment of the ground graph that illustrates a relationally *d*-connecting path (highlighted with bold arrows). When conditioning on Laptop.*Success*, Roger.*Competence* remains dependent on Devices.*Revenue* because of a path through Sally.*Competence* and Tablet.*Success*.

a ground graph to which *d*-separation can be applied (ground graphs were described in Section 3.6). For a Bayesian network, a ground graph consists of replicates of the model structure for each data instance. In contrast, a relational model defines a template for how dependencies apply to a data instantiation, resulting in a ground graph with varying structure.

Using the relevant subset of the ground graph from Figure 3.5, we can see that, for a single employee, simply conditioning on the success of developed products can activate a path through the competence of other employees who develop the same products—we call this a *relationally d-connecting path.*[2] For example, Roger's competence remains *d*-connected with Device's revenue when conditioning solely on the

[2]The indirect effect attributed to a relationally *d*-connecting path is often referred to as interference, a spillover effect, or a violation of the stable unit treatment value assumption (SUTVA) because the treatment of one instance (employee competence) affects the outcome of another (the revenue of another employee's business unit).

success of Laptops (see Figure 4.2). Checking d-separation on the ground graph indicates that to d-separate an employee's competence from the revenue of funding business units, we should not only condition on the success of developed products, but also on the competence of other employees who work on those products (e.g., Roger.$Competence \perp\!\!\!\perp$ Devices.$Revenue$ | {Laptop.$Success$, Sally.$Competence$}).

This example also demonstrates that the Markov condition can be violated when d-separation is directly applied to the structure of a relational model. In this case, the Markov condition according to the model structure in Figure 4.1 implies that

$$P(Competence, Revenue \mid Success) = P(Competence \mid Success)P(Revenue \mid Success),$$

that revenue is independent of its non-descendants (competence) given its parents (success). However, the ground graph shows the opposite, for example,

$$P(\text{Roger.}Competence, \text{Devices.}Revenue \mid \text{Laptop.}Success) \neq P(\text{Roger.}Competence \mid \text{Laptop.}Success)\ P(\text{Devices.}Revenue \mid \text{Laptop.}Success).$$

In fact, for this model, d-separation produces many other incorrect judgments of conditional independence. Through simulation, we found that only 25% of the pairs of variables can even be described by direct inspection of this model structure, and of those (such as the above example), 75% yield an incorrect conclusion. This is a single data point of a larger empirical evaluation presented in Section 4.7. Those results provide quantitative details of how often to expect traditional d-separation to fail when applied to the structure of relational models.

4.2 Semantics and Alternatives

The example in Section 4.1 provides a useful basis to describe the semantics imposed by relational d-separation and the characteristics of our approach. There are two primary concepts:

(1) *All-ground-graphs semantics*: It might appear that, since the standard rules of d-separation apply to Bayesian networks and the ground graphs of relational models are also Bayesian networks, that applying d-separation to relational models is a non-issue. However, applying d-separation to a single ground graph may result in potentially unbounded runtime if the instantiation is large (i.e., since relational databases can be arbitrarily large). Further, and more importantly, the semantics of d-separation require that conditional independencies hold across *all possible* model instantiations. Although d-separation can apply directly to a ground graph, these semantics prohibit reasoning about a single ground graph.

The conditional independence facts derived from d-separation hold for all distributions represented by a Bayesian network. Analogously, the implications of *relational* d-separation should hold for all distributions represented by a relational model. It is simple to show that these implications hold for all ground graphs of a Bayesian network—every ground graph consists of a set of disconnected subgraphs, each of which has a structure that is identical to that of the model. However, the set of distributions represented by a relational model depends on both the relational skeleton (constrained by the schema) and the model parameters. That is, the ground graphs of relational models vary with the structure of the underlying relational skeleton (e.g., different products are developed by varying numbers of employees). As a result, answering relational d-separation queries requires reasoning without respect to ground graphs.

(2) *Perspective-based analysis*: Relational models make explicit one implicit choice underlying nearly any form of data analysis. This choice—what we refer to here as a *perspective*—concerns the selection of a particular unit or subject of analysis. For example, in the social sciences, a commonly used acronym is *UTOS*, for framing an analysis by choosing a unit, treatment, outcome, and setting. Any method, such as Bayesian network modeling, that assumes IID data makes the implicit assumption

that the attributes on data instances correspond to attributes of a single unit or perspective. In the example, we targeted a specific conditional independence regarding employee instances (as opposed to products or business units).

The concept of perspectives is not new, but it is central to statistical relational learning because relational data sets may be heterogeneous, involving instances that refer to multiple, distinct perspectives. The inductive logic programming community has discussed individual-centered representations [40], and many approaches to propositionalizing relational data have been developed to enforce a *single* perspective to rely on existing propositional learning algorithms (see Section 3.8.4). An alternative strategy is to explicitly acknowledge the presence of multiple perspectives and learn jointly among them. This approach underlies many algorithms that learn the types of probabilistic models of relational data applicable in this work, e.g., learning the structure of probabilistic relational models [43], relational dependency networks [117], or parametrized Bayesian networks [156].

Often, data sets are derivative, leading to little or no choice about which perspectives to analyze. However, for relational domains, from which these data sets are derived, it is assumed that there are multiple perspectives, and we can dynamically analyze different perspectives. In the example, we chose the employee perspective, and the analysis focused on the dependence between an employee's competence and the revenue of business units that fund developed products. However, if the question were posed from the perspective of business units, then we could conceivably condition on the success of products for each business unit. In this scenario, reasoning about d-separation at the model level *would* lead to a correct conditional independence statement. Some (though fairly infrequent) d-separation queries produce accurate conditional independence facts when applied to relational model structure (see Section 4.7). However, the model is often unknown, a perspective may be chosen a priori, and a theory that is occasionally correct is clearly undesirable. Additionally, to sup-

port constraint-based learning algorithms, it is important to reason about conditional independence implications from different perspectives.

One plausible alternative approach would be to answer d-separation queries by ignoring perspectives and considering just the attribute classes (i.e., reason about *Competence* and *Revenue* given *Success*). However, it remains to define explicit semantics for grounding and evaluating the query based on the relational skeleton. There are at least three options:

- *Construct three sets of variables, including all instances of competence, revenue, and success variables*: Although the ground graph has the semantics of a Bayesian network, there is only a *single* ground graph—one data sample [195]. Consequently, this analysis would be statistically meaningless. Having a single data sample is the primary reason why relational learning algorithms dynamically *generate* propositional data for each instance of a given perspective.

- *Test the Cartesian product of competence and revenue variables, conditioned on all success variables*: Testing all pairs invariably leads to independence. Moreover, these semantics are incoherent; only reachable pairs of variables should be compared. For propositional data, variable pairs are constructed by choosing attribute values, e.g., height and weight, *within* an individual. The same is true for relational data: Only choose the success of products for employees that actually develop them, following the underlying relational connections.

- *Test relationally connected pairs of competence and revenue variables, conditioned on all success variables*: Again, this appears plausible based on traditional d-separation, but every instance in the table conditions on the *same* set of success values. Therefore, this is akin to not conditioning because the conditioning variable is a constant.

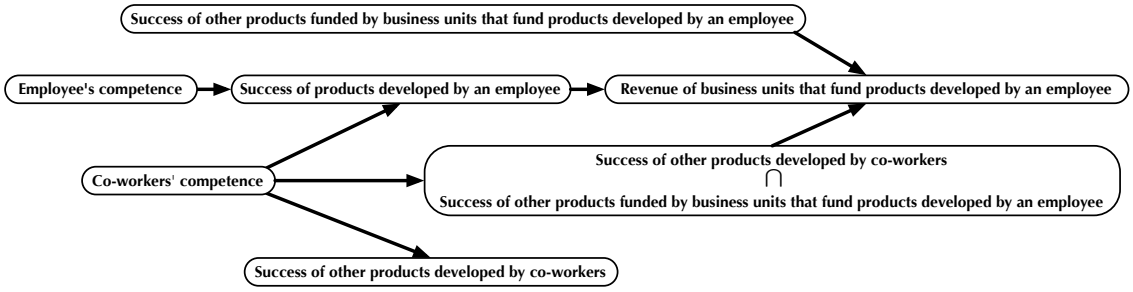


Figure 4.3: Example abstract ground graph from the perspective of employees. Nodes are labeled with their intuitive meaning.

We argue that the desired semantics are essentially the explicit semantics of perspective-based queries. Therefore, we advocate perspective-based analysis as the *only* statistically and semantically meaningful approach for relational data and models.

Our approach to answering relational d-separation queries incorporates the two aforementioned semantics. In Section 4.4, we describe a new, lifted representation—the abstract ground graph—that is provably sound and complete in its abstraction of all ground graphs for a given relational model. As their name suggests, abstract ground graphs *abstract* all ground graphs of a relational model, representing any potential relationally d-connecting path (recall the example d-connecting path that only manifests in the ground graph). A relational model has a corresponding *set* of abstract ground graphs, one for each perspective (i.e., entity or relationship class in its underlying schema), and can be used to reason about relational d-separation with respect to any given perspective. Figure 4.3 shows a fragment of an abstract ground graph from the employee perspective for the model in Figure 4.1. For this example, the nodes are depicted with their intuitive meaning rather than their actual syntax. Representational details and accompanying theory are presented in Section 4.4.

4.3 Relational *d*-separation

Conditional independence facts are correctly entailed by the rules of d-separation, but only when applied to the graphical structure of Bayesian networks. Every ground graph of a Bayesian network consists of a set of identical copies of the model structure. Thus, the implications of d-separation on Bayesian networks hold for all instances in every ground graph. In contrast, the structure of a relational model is a template for ground graphs, and the structure of a ground graph varies with the underlying skeleton (which is typically more complex than a set of disconnected instances). Conditional independence facts are only useful when they hold across all ground graphs that are consistent with the model, which leads to the following definition:

Definition 4.3.1 (Relational *d*-separation) Let $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ be three distinct sets of relational variables with the same perspective $B \in \mathcal{E} \cup \mathcal{R}$ defined over relational schema $\mathcal{S}$. Then, for relational model structure $\mathcal{M}$, $\mathbf{X}$ and $\mathbf{Y}$ are d-separated by $\mathbf{Z}$ if and only if, for all skeletons $\sigma \in \Sigma_{\mathcal{S}}$, $\mathbf{X}|_b$ and $\mathbf{Y}|_b$ are d-separated by $\mathbf{Z}|_b$ in ground graph $GG_{\mathcal{M}\sigma}$ for all $b \in \sigma(B)$.

For any relational d-separation query, it is necessary that all relational variables in $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ have the same perspective (otherwise, the query would be incoherent).[3] For $\mathbf{X}$ and $\mathbf{Y}$ to be d-separated by $\mathbf{Z}$ in relational model structure $\mathcal{M}$, d-separation must hold for all instantiations of those relational variables for all possible skeletons. This is a conservative definition, but it is consistent with the semantics of d-separation on Bayesian networks—it guarantees independence, but it does not guarantee dependence. If there exists even one skeleton and faithful distribution represented by the relational model for which $\mathbf{X} \not\perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z}$, then $\mathbf{X}$ and $\mathbf{Y}$ are not d-separated by $\mathbf{Z}$.

[3] This trivially holds for d-separation in Bayesian networks as all "propositional" variables have the same implicit perspective.

Given the semantics specified in Definition 4.3.1, answering relational d-separation queries is challenging for several reasons:

- *D-separation must hold over all ground graphs*: The implications of d-separation on Bayesian networks hold for all possible ground graphs. However, the ground graphs of a Bayesian network consist of identical copies of the structure of the model, and it is sufficient to reason about d-separation on a single subgraph. Although it is possible to verify d-separation on a single ground graph of a relational model, the conclusion may not generalize, and ground graphs can be arbitrarily large.

- *Relational models are templates*: The structure of a relational model is a directed acyclic graph, but the dependencies are actually templates for constructing ground graphs. The rules of d-separation do not directly apply to relational models, only to their ground graphs. Applying the rules of d-separation to a relational model frequently leads to incorrect conclusions because of unrepresented d-connecting paths that are only manifest in ground graphs.

- *Instances of relational variables may intersect*: The instances of two different relational variables may have non-empty intersections, as described by Lemma 3.4.1. These intersections may be involved in relationally d-connecting paths, as in the example in Section 4.1. As a result, a sound and complete approach to answering relational d-separation queries must account for these paths.

- *Relational models may be specified from multiple perspectives*: Relational models are defined by relational dependencies, each specified from a single perspective. However, variables in a ground graph may contribute to *multiple* relational variable instances, each defined from a *different* perspective. Thus, reasoning about implied dependencies between arbitrary relational variables, such as the

one described in Example 3.6.1, requires a method to translate dependencies across perspectives.

4.4 Abstract Ground Graphs

The definition of relational d-separation and its challenges suggest a solution that abstracts over all possible ground graphs and explicitly represents the potential intersection between pairs of relational variable instances. We introduce a new lifted representation, called the *abstract ground graph*, that captures all dependencies among arbitrary relational variables for all ground graphs, using knowledge of only the schema and the model. To represent all dependencies, the construction of an abstract ground graph uses the *extend* method, which maps a relational dependency to a set of implied dependencies for different perspectives. Each abstract ground graph of a relational model is defined with respect to a given perspective and can be used to reason about relational d-separation queries for that perspective.

Definition 4.4.1 (Abstract ground graph) An *abstract ground graph* $AGG_{\mathcal{M}B} = (V, E)$ for relational model structure $\mathcal{M} = (\mathcal{S}, \mathcal{D})$ and perspective $B \in \mathcal{E} \cup \mathcal{R}$ is a directed graph that abstracts the dependencies $\mathcal{D}$ for all ground graphs $GG_{\mathcal{M}\sigma}$, where $\sigma \in \Sigma_{\mathcal{S}}$.

The set of nodes in $AGG_{\mathcal{M}B}$ is $V = RV \cup IV$, where

- RV is the set of all relational variables of the form $[B, \ldots, I_j].X$
- IV is the set of all pairs of relational variables that could have non-empty intersections (referred to as intersection variables):

$$\big\{RV_1 \cap RV_2 \mid RV_1, RV_2 \in RV \ \wedge \ RV_1 = [B, \ldots, I_k, \ldots, I_j].X \\ \wedge \ RV_2 = [B, \ldots, I_l, \ldots, I_j].X \ \wedge \ I_k \neq I_l\big\}$$

The set of edges in $AGG_{\mathcal{M}B}$ is $E = RVE \cup IVE$, where

- $RVE \subset RV \times RV$ is the set of edges between pairs of relational variables:

$$RVE = \big\{[B, \ldots, I_k].Y \rightarrow [B, \ldots, I_j].X \mid [I_j, \ldots, I_k].Y \rightarrow [I_j].X \in \mathcal{D} \ \wedge \\ [B, \ldots, I_k] \in extend([B, \ldots, I_j], [I_j, \ldots, I_k])\big\}$$

- $IVE \subset IV \times RV \cup RV \times IV$ is the set of edges inherited from both relational variables involved in every intersection variable in IV:

$$IVE = \big\{\hat{Y} \rightarrow [B, \ldots, I_j].X \mid \hat{Y} = P_1.Y \cap P_2.Y \in IV \ \wedge \\ (P_1.Y \rightarrow [B, \ldots, I_j].X \in RVE \ \vee \\ P_2.Y \rightarrow [B, \ldots, I_j].X \in RVE)\big\}$$

$$\bigcup$$

$$\big\{[B, \ldots, I_k].Y \rightarrow \hat{X} \mid \hat{X} = P_1.X \cap P_2.X \in IV \ \wedge \\ ([B, \ldots, I_k].Y \rightarrow P_1.X \in RVE \ \vee \\ [B, \ldots, I_k].Y \rightarrow P_2.X \in RVE)\big\}$$

The *extend* method is described in Section 4.4.1. Essentially, the construction of an abstract ground graph for relational model structure $\mathcal{M}$ and perspective $B \in \mathcal{E} \cup \mathcal{R}$ follows three simple steps: (1) Add a node for all relational variables from perspective B.[4] (2) Insert edges for the direct causes of every relational variable by translating the dependencies in $\mathcal{D}$ using *extend*. (3) For each pair of potentially intersecting relational variables, create a new node that inherits the direct causes and effects from both participating relational variables in the intersection. Then, to answer queries of the form "Are $\mathbf{X}$ and $\mathbf{Y}$ d-separated by $\mathbf{Z}$?", simply (1) augment $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ with

their corresponding intersection variables that they subsume and (2) apply the rules of d-separation on the abstract ground graph for the common perspective of $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$. Since abstract ground graphs are defined from a specific perspective, every relational model produces a *set of abstract ground graphs*, one for each perspective in its underlying schema.

Example 4.4.1 Figure 4.4 shows the abstract ground graph $AGG_{\mathcal{M},\,\text{EMPLOYEE}}$ for the organization example from the EMPLOYEE perspective with hop threshold $h = 6$. As in Section 4.1, we derive an appropriate conditioning set $\mathbf{Z}$ in order to d-separate individual employee competence ($\mathbf{X} = \{[\text{EMPLOYEE}].Competence\}$) from the revenue of the employee's funding business units ($\mathbf{Y} = \{[\text{EMPLOYEE, DEVELOPS, PRODUCT, FUNDS, BUSINESS-UNIT}].Revenue\}$). Applying the rules of d-separation to the abstract ground graph, we see that it is necessary to condition on both product success ([EMPLOYEE, DEVELOPS, PRODUCT].*Success*) and the competence of other employees developing the same products ([EMPLOYEE, DEVELOPS, PRODUCT, DEVELOPS, EMPLOYEE].*Competence*). For $h = 6$, augmenting $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ with their corresponding intersection variables does not result in any changes. For $h = 8$, the abstract ground graph includes a node for relational variable [EMPLOYEE, DEVELOPS, PRODUCT, DEVELOPS, EMPLOYEE, DEVELOPS, PRODUCT, FUNDS, BUSINESS-UNIT].*Revenue* (the revenue of the business units funding the other products of collaborating employees) which, by Lemma 3.4.1, could have a non-empty intersection with [EMPLOYEE, DEVELOPS, PRODUCT, FUNDS, BUSINESS-UNIT].*Revenue*. Therefore, $\mathbf{Y}$ would also include the intersection with this other relational variable. However, for this query, the conditioning set $\mathbf{Z}$ for $h = 6$ happens to also d-separate at $h = 8$ (and any $h \in \mathbb{N}^0$). □

[4]In theory, abstract ground graphs can have an infinite number of nodes as relational paths may have no bound. In practice, a *hop threshold* $h \in \mathbb{N}^0$ is enforced to limit the length of these paths. Hops are defined as the number of times the path "hops" between item classes in the schema, or one less than the length of the path.

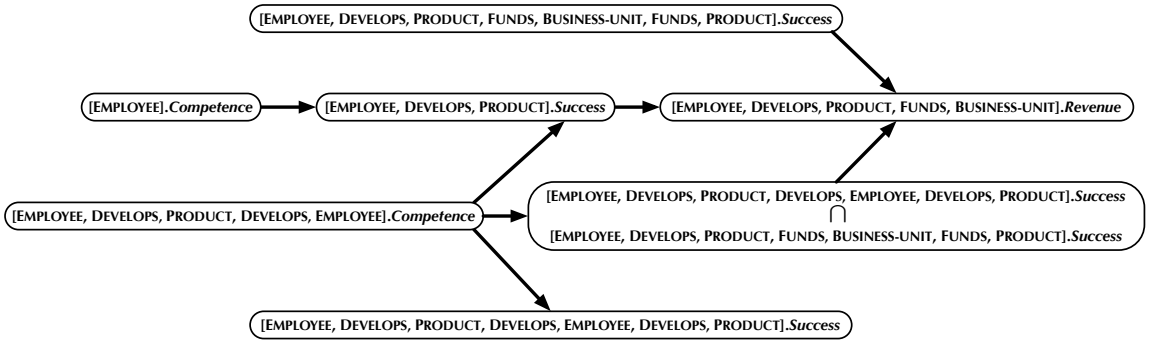


Figure 4.4: The abstract ground graph for the organization domain model in Figure 4.1 from the EMPLOYEE perspective with hop threshold $h = 6$. This abstract ground graph includes one intersection node.

Using the algorithm devised by Geiger et al. [49], relational d-separation queries can be answered in $O(|E|)$ time with respect to the number of edges in the abstract ground graph. In practice, the size of an abstract ground graph depends on properties of the relational schema and model (e.g., the number of entity classes, the types of cardinalities, the number of dependencies—see the experiment in Section 4.8.1), as well as the hop threshold limiting the length of relational paths. For the example in Figure 4.4, the abstract ground graph has 7 nodes and 7 edges (including 1 intersection node with 2 edges); for $h = 8$, it has 13 nodes and 21 edges (including 4 intersection nodes with 13 edges). Abstract ground graphs are invariant to the size of ground graphs, even though ground graphs can be arbitrarily large—that is, relational databases have no maximum size.

4.4.1 Inserting edges in abstract ground graphs: The *extend* method

Internal to the construction of abstract ground graphs is the *extend* method, which we now formally define. At a high level, this method translates dependencies specified in a relational model into dependencies in its abstract ground graphs.

Definition 4.4.2 (Extending relational paths) Let P_{orig} and P_{ext} be two relational paths for schema $\mathcal{S}$. The following three functions extend P_{orig} with P_{ext}:

$$extend(P_{orig}, P_{ext}) = \{P = P_{orig}^{1,n_o-i+1} + P_{ext}^{i+1,n_e} \mid i \in pivots(reverse(P_{orig}), P_{ext}) \wedge isValid(P)\}$$

$$pivots(P_1, P_2) = \{i \mid P_1^{1,i} = P_2^{1,i}\}$$

$$isValid(P) = \begin{cases} \text{True} & \text{if } P \text{ does not violate Definition 3.3.1} \\ \text{False} & \text{otherwise} \end{cases}$$

where n_o is the length of P_{orig}; n_e is the length of P_{ext}; $P^{i,j}$ corresponds to 1-based i-inclusive, j-inclusive subpath indexing; + is concatenation of paths; and *reverse* is a method that reverses the order of the path.

The *extend* method constructs a set of valid relational paths from two input relational paths. It first finds the indices (called pivots) of the item classes for which the input paths ($reverse(P_{orig})$ and P_{ext}) have a common starting subpath. Then, it concatenates the two input paths at each pivot, removing one of the duplicated subpaths (see Example 4.4.2). Since d-separation requires blocking all paths of dependence between two sets of variables, the *extend* method is critical to ensure the soundness and completeness of our approach. The abstract ground graph must capture all paths of dependence among the random variables in the relational variable instances for all represented ground graphs. However, relational model structures are specified by relational dependencies, each from a given perspective and with outcomes that have singleton relational paths. The *extend* method is called repeatedly during the creation of an abstract ground graph, with P_{orig} set to some relational path and P_{ext} drawn from the relational path of the treatment in some relational dependency.

Example 4.4.2 During construction of the abstract ground graph $AGG_{\mathcal{M},\,\text{EMPLOYEE}}$ depicted in Figure 4.4, the *extend* method is called several times. First, all relational variables from the EMPLOYEE perspective are added as nodes. Next, *extend* inserts

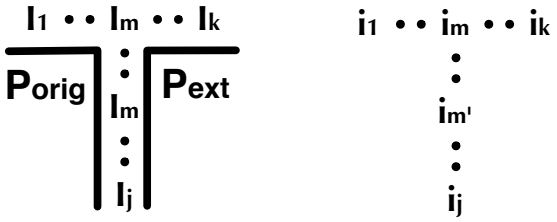


Figure 4.5: Example construction of a relational skeleton for two relational paths $P_{orig} = [I_1, \ldots, I_m \ldots, I_j]$ and $P_{ext} = [I_j, \ldots, I_m \ldots, I_k]$, where item class I_m is repeated between I_m and I_j. This construction is used within the proof of Lemma 4.4.1.

edges corresponding to direct causes. Consider the node for [EMPLOYEE, DEVELOPS, PRODUCT].*Success*. The construction of $AGG_{\mathcal{M},\,\text{EMPLOYEE}}$ calls $extend(P_{orig}, P_{ext})$ with P_{orig} = [EMPLOYEE, DEVELOPS, PRODUCT] and P_{ext} = [PRODUCT, DEVELOPS, EMPLOYEE] because [PRODUCT, DEVELOPS, EMPLOYEE].*Competence* $\rightarrow$ [PRODUCT].*Success* $\in \mathcal{D}$. Here, $extend(P_{orig}, P_{ext})$ = {[EMPLOYEE], [EMPLOYEE, DEVELOPS, PRODUCT, DEVELOPS, EMPLOYEE]}, which leads to the insertion of two edges in the abstract ground graph. Note that $pivots(reverse(P_{orig}), P_{ext}) = \{1, 2, 3\}$, and the pivot at $i = 2$ yields the invalid relational path [EMPLOYEE, DEVELOPS, EMPLOYEE]. □

4.4.2 Theoretical properties of the *extend* method

We also describe two important properties of the *extend* method with the following two lemmas. The first lemma states that every relational path produced by *extend* yields a terminal set for some skeleton such that there is an item instance also reachable by the two original paths. This lemma is useful for proving the soundness of our abstraction: All edges inserted in an abstract ground graph correspond to edges in some ground graph.

Lemma 4.4.1 *Let* $P_{orig} = [I_1, \ldots, I_j]$ *and* $P_{ext} = [I_j, \ldots, I_k]$ *be two relational paths with* $\mathbf{P} = extend(P_{orig}, P_{ext})$. *Then,* $\forall P \in \mathbf{P}$ *there exists a relational skeleton* $\sigma \in \Sigma_{\mathcal{S}}$ *such that* $\exists i_1 \in \sigma(I_1)$ *such that* $\exists i_k \in P|_{i_1}$ *and* $\exists i_j \in P_{orig}|_{i_1}$ *such that* $i_k \in P_{ext}|_{i_j}$.

Proof. Let $P \in \mathbf{P}$ be an arbitrary valid relational path, where $P = P_{orig}^{1,n_o-c+1} + P_{ext}^{c+1,n_e}$ for pivot c. There are two subcases:

(a) $c = 1$ and $P = [I_1, \ldots, I_j, \ldots, I_k]$. This subcase holds generally for any skeleton. Proof by contradiction. Let σ be an arbitrary skeleton, choose $i_1 \in \sigma(I_1)$ arbitrarily, and choose $i_k \in P|_{i_1}$ arbitrarily. Assume for contradiction that there is no i_j in the terminal set $P_{orig}|_{i_1}$ such that i_k would be in the terminal set $P_{ext}|_{i_j}$; that is, $\forall i_j \in P_{orig}|_{i_1}\ i_k \notin P_{ext}|_{i_j}$. Since $P = [I_1, \ldots, I_j, \ldots, I_k]$, we know that i_k is reached by traversing σ from i_1 via some i_j to i_k. But the traversal from i_1 to i_j implies that $i_j \in [I_1, \ldots, I_j]|_{i_1} = P_{orig}|_{i_1}$, and the traversal from i_j to i_k implies that $i_k \in [I_j, \ldots, I_k]|_{i_j} = P_{ext}|_{i_j}$. Therefore, there must exist an $i_j \in P_{orig}|_{i_1}$ such that $i_k \in P_{ext}|_{i_j}$.

(b) $c > 1$ and $P = [I_1, \ldots, I_m, \ldots, I_k]$. Proof by construction. We build a relational skeleton σ following the same procedure as outlined in the proof of Lemma 3.4.1. Add instances to σ for every item class that appears on P_{orig} and P_{ext}. Since $P = [I_1, \ldots, I_m, \ldots, I_k]$, we know that i_k is reached by traversing σ from i_1 via some i_m to i_k. By case (a), $\exists i_m \in [I_1, \ldots, I_m]|_{i_1}$ such that $i_k \in [I_m, \ldots, I_k]|_{i_m}$. We then must show that there exists an $i_j \in [I_m, \ldots, I_j]|_{i_m}$ with $i_m \in [I_j, \ldots, I_m]|_{i_j}$. But constructing the skeleton with unique item instances for every appearance of an item class on the relational paths provides this and does not violate any cardinality constraints. If any item class appears more than once, then the bridge burning semantics are induced. However, adding an additional item instance for every reappearance of an item class enables the traversal from i_j to i_m and vice versa. An example of this construction is displayed in Figure 4.5. This is also a valid relational skeleton because P_{orig} and P_{ext} are valid relational paths, and by definition, the cardinality constraints of the schema permit multiple instances in the skeleton of any repeated item class. By this procedure, we show that there exists a skeleton σ such that there exists an $i_j \in P_{orig}|_{i_1}$ such that $i_k \in P_{ext}|_{i_j}$. ■

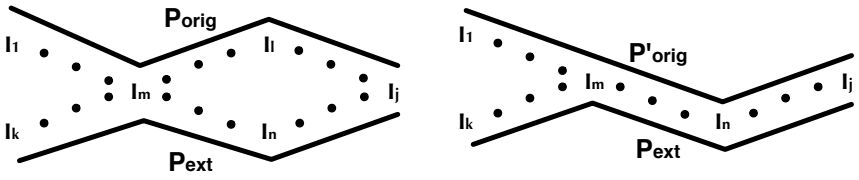


Figure 4.6: Schematic of the relational paths expected in Lemma 4.4.2. If item i_k is unreachable via $extend(P_{orig}, P_{ext})$, then there must exist a P'_{orig} of the form $[I_1, \ldots, I_m, \ldots, I_n, \ldots, I_j]$.

Example 4.4.3 Let σ be the skeleton shown in Figure 3.2, let P_{orig} = [Employee, Develops, Product], let P_{ext} = [Product, Develops, Employee], and let i_1 = Sally $\in \sigma$(Employee). From Example 4.4.2, we know that $\mathbf{P} = extend(P_{orig}, P_{ext})$ = {[Employee], [Employee, Develops, Product, Develops, Employee]}. We also have [Employee]$|_{\text{Sally}}$ = {Sally} and [Employee, Develops, Product, Develops, Employee]$|_{\text{Sally}}$ = {Quinn, Roger, Thomas}. By Lemma 4.4.1, there should exist an $i_j \in P_{orig}|_{i_1}$ such that Sally and at least one of Quinn, Roger, and Thomas are in the terminal set $P_{ext}|_{i_j}$. We have $P_{orig}|_{\text{Sally}}$ = {Laptop, Tablet}, and $P_{ext}|_{\text{Laptop}}$ = {Quinn, Roger, Sally} and $P_{ext}|_{\text{Tablet}}$ = {Sally, Thomas}. So, the lemma clearly holds for this example. □

Lemma 4.4.1 guarantees that, for some relational skeleton, there exists an item instance in the terminal sets produced by *extend* that also appears in the terminal set of P_{ext} via some instance in the terminal set of P_{orig}. It is also possible (although infrequent) that there exist items reachable by P_{orig} and P_{ext} that are not in the terminal set of any path produced with $extend(P_{orig}, P_{ext})$. The following lemma describes this unreachable set of items, stating that there must exist an alternative relational path P'_{orig} that intersects with P_{orig} and, when using *extend*, catches those remaining items. This lemma is important for proving the completeness of our abstraction: All edges in all ground graphs are represented in the abstract ground graph.

Lemma 4.4.2 *Let $\sigma \in \Sigma_{\mathcal{S}}$ be a relational skeleton, and let $P_{orig} = [I_1, \ldots, I_j]$ and $P_{ext} = [I_j, \ldots, I_k]$ be two relational paths with $\mathbf{P} = extend(P_{orig}, P_{ext})$. Then, $\forall i_1 \in \sigma(I_1)$ $\forall i_j \in P_{orig}|_{i_1}$ $\forall i_k \in P_{ext}|_{i_j}$ if $\forall P \in \mathbf{P}$ $i_k \notin P|_{i_1}$, then $\exists P'_{orig}$ such that $P_{orig}|_{i_1} \cap P'_{orig}|_{i_1} \neq \emptyset$ and $i_k \in P'|_{i_1}$ for some $P' \in extend(P'_{orig}, P_{ext})$.*

Proof. Proof by construction. Let $i_1 \in \sigma(I_1)$, $i_j \in P_{orig}|_{i_1}$, and $i_k \in P_{ext}|_{i_j}$ be arbitrary instances such that $i_k \notin P|_{i_1}$ for all $P \in \mathbf{P}$.

Since $i_j \in P_{orig}|_{i_1}$ and $i_k \in P_{ext}|_{i_j}$, but $i_k \notin P|_{i_1}$, there exists no pivot that yields a common subsequence in P_{orig} and P_{ext} that produces a path in *extend* that can reach i_k. Let the first divergent item class along the reverse of P_{orig} be I_l and along P_{ext} be I_n. The two paths must not only diverge, but they also necessarily reconverge at least once. If P_{orig} and P_{ext} do not reconverge, then there are no reoccurrences of an item class along any $P \in \mathbf{P}$ that would restrict the inclusion of i_k in some terminal set $P|_{i_1}$. The sole reason that $i_k \notin P|_{i_1}$ for all $P \in \mathbf{P}$ is the bridge burning semantics specified in Definition 3.3.2.

Without loss of generality, assume P_{orig} and P_{ext} reconverge once, at item class I_m. So, $P_{orig} = [I_1, \ldots, I_m, \ldots, I_l, \ldots, I_j]$ and $P_{ext} = [I_j, \ldots, I_n, \ldots, I_m, \ldots, I_k]$ with $I_l \neq I_n$, as depicted in Figure 4.6. Let $P'_{orig} = [I_1, \ldots, I_m, \ldots, I_n, \ldots, I_j]$, which is a valid relational path because $[I_1, \ldots, I_m]$ is a subpath of P_{orig} and $[I_m, \ldots, I_n, \ldots, I_j]$ is a subpath of P_{ext}.

By construction, i_j is in the intersection: $i_j \in P_{orig}|_{i_1} \cap P'_{orig}|_{i_1}$. If $P' = [I_1, \ldots, I_m, \ldots, I_k] \in extend(P'_{orig}, P_{ext})$ with pivot at I_m, then $i_k \in P'|_{i_1}$. ■

Example 4.4.4 Although Lemma 4.4.2 does not apply to the organization domain as currently represented, it could apply if either (1) there were cycles in the relational schema or (2) the path specifications on the relational dependencies included a cycle. Consider additional relationships between employees and products. If employees could be involved with products at various stages (e.g., research, development, testing, marketing), then there would be alternative relational paths for which the lemma

might apply. The proof of the lemma provides abstract conditions describing when the lemma applies. □

4.5 Soundness and Completeness of Relational *d*-Separation

The correctness of our approach to relational d-separation relies on several facts: (1) d-separation is valid for directed acyclic graphs; (2) ground graphs are directed acyclic graphs; and (3) abstract ground graphs are directed acyclic graphs that represent exactly the edges that could appear in all possible ground graphs. It follows that d-separation on abstract ground graphs, augmented by intersection variables, is sound and complete for all ground graphs.[5] Additionally, we show that since relational d-separation is sound and complete, it is also equivalent to the Markov condition for relational models. Using the previous definitions and lemmas, the following sequence of results proves the correctness of our approach to identifying independence in relational models.

Theorem 4.5.1 *The rules of d-separation are sound and complete for directed acyclic graphs.*

Proof. Due to Verma and Pearl [191] for soundness and Geiger and Pearl [47] for completeness. ■

Theorem 4.5.1 implies that (1) all conditional independence facts derived by d-separation on a Bayesian network structure hold in any distribution represented by that model (soundness) and (2) all conditional independence facts that hold in a faithful distribution can be inferred from d-separation applied to the Bayesian network that encodes the distribution (completeness).

[5] In Section 4.6, we provide proofs of soundness and completeness for abstract ground graphs and relational d-separation, both of which are limited by practical hop threshold bounds.

Lemma 4.5.1 *For every acyclic relational model structure $\mathcal{M}$ and skeleton $\sigma \in \Sigma_{\mathcal{S}}$, the ground graph $GG_{\mathcal{M}\sigma}$ is a directed acyclic graph.*

Proof. Due to both Heckerman et al. [62] for DAPER models and Getoor [52] for PRMs. ■

By Theorem 4.5.1 and Lemma 4.5.1, d-separation is sound and complete when applied to a ground graph. However, Definition 4.3.1 states that relational d-separation must hold across *all possible* ground graphs, which is the reason for constructing the abstract ground graph representation.

Theorem 4.5.2 *For every acyclic relational model structure $\mathcal{M}$ and perspective $B \in \mathcal{E} \cup \mathcal{R}$, the abstract ground graph $AGG_{\mathcal{M}B}$ is sound and complete for all ground graphs $GG_{\mathcal{M}\sigma}$ with skeleton $\sigma \in \Sigma_{\mathcal{S}}$.*

Proof. Let $\mathcal{M} = (\mathcal{S}, \mathcal{D})$ be an arbitrary acyclic relational model structure and let $B \in \mathcal{E} \cup \mathcal{R}$ be an arbitrary perspective.

Soundness: To prove that $AGG_{\mathcal{M}B}$ is sound, we must show that for every edge $P_k.X \rightarrow P_j.Y$ in $AGG_{\mathcal{M}B}$, there exists a corresponding edge $i_k.X \rightarrow i_j.Y$ in the ground graph $GG_{\mathcal{M}\sigma}$ for some skeleton $\sigma \in \Sigma_{\mathcal{S}}$, where $i_k \in P_k|_b$ and $i_j \in P_j|_b$ for some $b \in \sigma(B)$. There are three subcases, one for each type of edge in an abstract ground graph:

(a) Let $[B, \ldots, I_k].X \rightarrow [B, \ldots, I_j].Y \in RVE$ be an arbitrary edge in $AGG_{\mathcal{M}B}$ between a pair of relational variables. Assume for contradiction that there exists no edge $i_k.X \rightarrow i_j.Y$ in any ground graph:

$$\forall \sigma \in \Sigma_{\mathcal{S}}\ \forall b \in \sigma(B)\ \forall i_k \in [B, \ldots, I_k]|_b\ \forall i_j \in [B, \ldots, I_j]|_b\ \left(i_k.X \rightarrow i_j.Y \notin GG_{\mathcal{M}\sigma}\right)$$

By Definition 4.4.1 for abstract ground graphs, if $[B, \ldots, I_k].X \rightarrow [B, \ldots, I_j].Y \in RVE$, then the model must have dependency $[I_j, \ldots, I_k].X \rightarrow [I_j].Y \in \mathcal{D}$ such that $[B, \ldots, I_k] \in extend([B, \ldots, I_j], [I_j, \ldots, I_k])$. So, by Definition 3.6.1 for ground

graphs, there is an edge from every $i_k.X$ to every $i_j.Y$, where i_k is in the terminal set for i_j along $[I_j, \ldots, I_k]$:

$$\forall \sigma \in \Sigma_{\mathcal{S}}\ \forall i_j \in \sigma(I_j)\ \forall i_k \in [I_j, \ldots, I_k]|_{i_j}\ \left(i_k.X \rightarrow i_j.Y \in GG_{\mathcal{M}\sigma}\right)$$

Since $[B, \ldots, I_k] \in \mathit{extend}([B, \ldots, I_j], [I_j, \ldots, I_k])$, by Lemma 4.4.1 we know that

$$\exists \sigma \in \Sigma_{\mathcal{S}}\ \exists b \in \sigma(B)\ \exists i_k \in [B, \ldots, I_k]|_b\ \exists i_j \in [B, \ldots, I_j]|_b\ \left(i_k \in [I_j, \ldots, I_k]|_{i_j}\right)$$

Therefore, there exists a ground graph $GG_{\mathcal{M}\sigma}$ such that $i_k.X \rightarrow i_j.Y \in GG_{\mathcal{M}\sigma}$, which contradicts the assumption.

(b) Let $P_1.X \cap P_2.X \rightarrow [B, \ldots, I_j].Y \in \mathit{IVE}$ be an arbitrary edge in $AGG_{\mathcal{M}B}$ between an intersection variable and a relational variable, where $P_1{=}[B, \ldots, I_m, \ldots, I_k]$ and $P_2 = [B, \ldots, I_n, \ldots, I_k]$ with $I_m \neq I_n$. By Lemma 3.4.1, there exists a skeleton $\sigma \in \Sigma_{\mathcal{S}}$ and $b \in \sigma(B)$ such that $P_1|_b \cap P_2|_b \neq \emptyset$. Let $i_k \in P_1|_b \cap P_2|_b$ and assume for contradiction that for all $i_j \in [B, \ldots, I_j]|_b$ there is no edge $i_k.X \rightarrow i_j.Y$ in the ground graph $GG_{\mathcal{M}\sigma}$. By Definition 4.4.1, if the abstract ground graph has edge $P_1.X \cap P_2.X \rightarrow [B, \ldots, I_j].Y \in \mathit{IVE}$, then either $P_1.X \rightarrow [B, \ldots, I_j].Y \in \mathit{RVE}$ or $P_2.X \rightarrow [B, \ldots, I_j].Y \in \mathit{RVE}$. Then, as shown in case (a), there exists an $i_j \in [B, \ldots, I_j]|_b$ such that $i_k.X \rightarrow i_j.Y \in GG_{\mathcal{M}\sigma}$, which contradicts the assumption.

(c) Let $[B, \ldots, I_k].X \rightarrow P_1.Y \cap P_2.Y \in \mathit{IVE}$ be an arbitrary edge in $AGG_{\mathcal{M}B}$ between a relational variable and an intersection variable, where $P_1{=}[B, \ldots, I_m, \ldots, I_j]$ and $P_2 = [B, \ldots, I_n, \ldots, I_j]$ with $I_m \neq I_n$. The proof follows case (b) to show that there exists a skeleton $\sigma \in \Sigma_{\mathcal{S}}$ and $b \in \sigma(B)$ such that for all $i_k \in [B, \ldots, I_k]|_b$ there exists an $i_j \in P_1 \cap P_2|_b$ such that $i_k.X \rightarrow i_j.Y \in GG_{\mathcal{M}\sigma}$.

Completeness: To prove that the abstract ground graph $AGG_{\mathcal{M}B}$ is complete, we show that for every edge $i_k.X \rightarrow i_j.Y$ in every ground graph $GG_{\mathcal{M}\sigma}$ where $\sigma \in \Sigma_{\mathcal{S}}$, there is a set of corresponding edges in $AGG_{\mathcal{M}B}$. Specifically, the edge $i_k.X \rightarrow i_j.Y$ yields two sets of relational variables for some $b \in \sigma(B)$, namely $\mathbf{P_k}.\mathbf{X} = \{P_k.X \mid i_k \in$

$P_k|_b\}$ and $\mathbf{P_j}.\mathbf{Y} = \{P_j.Y \mid i_j \in P_j|_b\}$. Note that all relational variables in both $\mathbf{P_k}.\mathbf{X}$ and $\mathbf{P_j}.\mathbf{Y}$ are nodes in $AGG_{\mathcal{M}B}$, as are all pairwise intersection variables: $\forall P_k.X, P'_k.X \in \mathbf{P_k}.\mathbf{X}$ $\big(P_k.X \cap P'_k.X \in AGG_{\mathcal{M}B}\big)$ and $\forall P_j.Y, P'_j.Y \in \mathbf{P_j}.\mathbf{Y}$ $\big(P_j.Y \cap P'_j.Y \in AGG_{\mathcal{M}B}\big)$. We show that for all $P_k.X \in \mathbf{P_k}.\mathbf{X}$ and for all $P_j.Y \in \mathbf{P_j}.\mathbf{Y}$ either (a) $P_k.X \to P_j.Y \in AGG_{\mathcal{M}B}$, (b) $P_k.X \cap P'_k.X \to P_j.Y \in AGG_{\mathcal{M}B}$, where $P'_k.X \in \mathbf{P_k}.\mathbf{X}$, or (c) $P_k.X \to P_j.Y \cap P'_j.Y \in AGG_{\mathcal{M}B}$, where $P'_j.Y \in \mathbf{P_j}.\mathbf{Y}$.

Let $\sigma \in \Sigma_{\mathcal{S}}$ be an arbitrary skeleton, let $i_k.X \to i_j.Y \in GG_{\mathcal{M}\sigma}$ be an arbitrary edge drawn from $[I_j, \ldots, I_k].X \to [I_j].Y \in \mathcal{D}$, and let $P_k.X \in \mathbf{P_k}.\mathbf{X}, P_j.Y \in \mathbf{P_j}.\mathbf{Y}$ be an arbitrary pair of relational variables.

(a) If $P_k \in \mathit{extend}(P_j, [I_j, \ldots, I_k])$, then $P_k.X \to P_j.Y \in AGG_{\mathcal{M}B}$ by Definition 4.4.1.

(b) If $P_k \notin \mathit{extend}(P_j, [I_j, \ldots, I_k])$, but $\exists P'_k \in \mathit{extend}(P_j, [I_j, \ldots, I_k])$ such that $P'_k.X \in \mathbf{P_k}.\mathbf{X}$, then $P'_k.X \to P_j.Y \in AGG_{\mathcal{M}B}$, and $P_k.X \cap P'_k.X \to P_j.Y \in AGG_{\mathcal{M}B}$ by Definition 4.4.1.

(c) If $\forall P \in \mathit{extend}(P_j, [I_j, \ldots, I_k])$ $\big(P.X \notin \mathbf{P_k}.\mathbf{X}\big)$, then by Lemma 4.4.2, $\exists P'_j$ such that $i_j \in P'_j|_b$ and $P_k \in \mathit{extend}(P'_j, [I_j, \ldots, I_k])$. Therefore, $P'_j.Y \in \mathbf{P_j}.\mathbf{Y}$, $P_k.X \to P'_j.Y \in AGG_{\mathcal{M}B}$, and $P_k.X \to P'_j.Y \cap P_j.Y \in AGG_{\mathcal{M}B}$ by Definition 4.4.1. ■

Theorem 4.5.2 guarantees that, for a given perspective, an abstract ground graph captures all possible paths of dependence between any pair of variables in any ground graph. The details of the proof provide the reasons why explicitly representing intersection variables is necessary for ensuring a sound and complete abstraction.

Theorem 4.5.3 *For every acyclic relational model structure $\mathcal{M}$ and perspective $B \in \mathcal{E} \cup \mathcal{R}$, the abstract ground graph $AGG_{\mathcal{M}B}$ is directed and acyclic.*

Proof. Let $\mathcal{M}$ be an arbitrary acyclic relational model structure, and let $B \in \mathcal{E} \cup \mathcal{R}$ be an arbitrary perspective. It is clear by Definition 4.4.1 that every edge in the abstract ground graph $AGG_{\mathcal{M}B}$ is directed by construction. All edges inserted in

any abstract ground graph are drawn from the directed dependencies in a relational model. Since $\mathcal{M}$ is acyclic, the class dependency graph $G_{\mathcal{M}}$ is also acyclic by Definition 3.6.2. Assume for contradiction that there exists a cycle of length n in $AGG_{\mathcal{M}B}$ that contains both relational variables and intersection variables. By Definition 4.4.1, all edges inserted in $AGG_{\mathcal{M}B}$ are drawn from some dependency in $\mathcal{M}$, even for nodes corresponding to intersection variables. Retaining only the final item class in each relational path for every node in the cycle will yield a cycle in $G_{\mathcal{M}}$ by Definition 3.6.2. Therefore, $\mathcal{M}$ could not have been acyclic, which contradicts the assumption. ■

Theorem 4.5.3 ensures that the standard rules of d-separation can apply directly to abstract ground graphs because they are acyclic given an acyclic model. We now have sufficient supporting theory to prove that d-separation on abstract ground graphs is sound and complete. In the following theorem, we define $\bar{\mathbf{W}}$ as the set of nodes augmented with their corresponding intersection nodes for the set of relational variables $\mathbf{W}$: $\bar{\mathbf{W}} = \mathbf{W} \cup \bigcup_{W \in \mathbf{W}} \{W \cap W' \mid W \cap W' \text{ is an intersection node in } AGG_{\mathcal{M}B}\}$.

Theorem 4.5.4 *Relational d-separation is sound and complete for abstract ground graphs. Let $\mathcal{M}$ be an acyclic relational model structure, and let $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ be three distinct sets of relational variables for perspective $B \in \mathcal{E} \cup \mathcal{R}$ defined over relational schema $\mathcal{S}$. Then, $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ are d-separated by $\bar{\mathbf{Z}}$ on the abstract ground graph $AGG_{\mathcal{M}B}$ if and only if for all skeletons $\sigma \in \Sigma_{\mathcal{S}}$ and for all $b \in \sigma(B)$, $\mathbf{X}|_b$ and $\mathbf{Y}|_b$ are d-separated by $\mathbf{Z}|_b$ in ground graph $GG_{\mathcal{M}\sigma}$.*

Proof. We must show that d-separation on an abstract ground graph implies d-separation on all ground graphs it represents (soundness) and that d-separation facts that hold across all ground graphs are also entailed by d-separation on the abstract ground graph (completeness).

Soundness: Assume that $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ are d-separated by $\bar{\mathbf{Z}}$ on $AGG_{\mathcal{M}B}$. Assume for contradiction that there exists an item instance $b \in \sigma(B)$ such that $\mathbf{X}|_b$ and $\mathbf{Y}|_b$

are *not* d-separated by $\mathbf{Z}|_b$ in the ground graph $GG_{\mathcal{M}\sigma}$ for some arbitrary skeleton σ. Then, there must exist a d-connecting path p from some $x \in \mathbf{X}|_b$ to some $y \in \mathbf{Y}|_b$ given all $z \in \mathbf{Z}|_b$. By Theorem 4.5.2, $AGG_{\mathcal{M}B}$ is complete, so all edges in $GG_{\mathcal{M}\sigma}$ are captured by edges in $AGG_{\mathcal{M}B}$. So, path p must be represented from some node in $\{N_x \mid x \in N_x|_b\}$ to some node in $\{N_y \mid y \in N_y|_b\}$, where N_x, N_y are nodes in $AGG_{\mathcal{M}B}$. If p is d-connecting in $GG_{\mathcal{M}\sigma}$, then it is d-connecting in $AGG_{\mathcal{M}B}$, implying that $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ are *not* d-separated by $\bar{\mathbf{Z}}$. So, $\mathbf{X}|_b$ and $\mathbf{Y}|_b$ must be d-separated by $\mathbf{Z}|_b$.

Completeness: Assume that $\mathbf{X}|_b$ and $\mathbf{Y}|_b$ are d-separated by $\mathbf{Z}|_b$ in the ground graph $GG_{\mathcal{M}\sigma}$ for all skeletons σ for all $b \in \sigma(B)$. Assume for contradiction that $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ are *not* d-separated by $\bar{\mathbf{Z}}$ on $AGG_{\mathcal{M}B}$. Then, there must exist a d-connecting path p for some relational variable $X \in \bar{\mathbf{X}}$ to some $Y \in \bar{\mathbf{Y}}$ given all $Z \in \bar{\mathbf{Z}}$. By Theorem 4.5.2, $AGG_{\mathcal{M}B}$ is sound, so every edge in $AGG_{\mathcal{M}B}$ must correspond to some pair of variables in some ground graph. So, if p is d-connecting in $AGG_{\mathcal{M}B}$, then there must exist some skeleton σ such that p is d-connecting in $GG_{\mathcal{M}\sigma}$ for some $b \in \sigma(B)$, implying that d-separation does not hold for that ground graph. So, $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ must be d-separated by $\bar{\mathbf{Z}}$ on $AGG_{\mathcal{M}B}$. ■

Theorem 4.5.4 proves that d-separation on abstract ground graphs is a sound and complete solution to identifying independence in relational models. Theorem 4.5.1 also implies that the set of conditional independence facts derived from abstract ground graphs is exactly the same as the set of conditional independencies that all distributions represented by all possible ground graphs have in common.

Corollary 4.5.1 *$\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ are d-connected given $\bar{\mathbf{Z}}$ on the abstract ground graph $AGG_{\mathcal{M}B}$ if and only if there exists a skeleton $\sigma \in \Sigma_{\mathcal{S}}$ and an item instance $b \in \sigma(B)$ such that $\mathbf{X}|_b$ and $\mathbf{Y}|_b$ are d-connected given $\mathbf{Z}|_b$ in ground graph $GG_{\mathcal{M}\sigma}$.*

Corollary 4.5.1 is logically equivalent to Theorem 4.5.4. While a simple restatement of the previous theorem, it is important to emphasize that relational d-

separation claims d-connection if and only if there exists a ground graph for which $X|_b$ and $Y|_b$ are d-connected given $\mathbf{Z}|_b$. This implies that there may be some ground graphs for which $X|_b$ and $Y|_b$ are d-separated by $\mathbf{Z}|_b$, but the abstract ground graph still claims d-connection. This may happen if the relational skeleton does not enable certain underlying relational connections. For example, if the relational skeleton in Figure 3.2 included only products that were developed by a single employee, then there would be no relationally d-connecting path in the example in Section 4.1. If this is a fundamental property of the domain (e.g., there are products developed by a single employee and products developed by multiple employees), then revising the underlying schema to include two different classes of products would yield a more accurate model, implying a larger set of conditional independencies.

Additionally, we can show that relational d-separation is equivalent to the Markov condition on relational models.

Definition 4.5.1 (Relational Markov condition) Let X be a relational variable for perspective $B \in \mathcal{E} \cup \mathcal{R}$ defined over relational schema $\mathcal{S}$. Let $nd(X)$ be the non-descendant variables of X, and let $pa(X)$ be the set of parent variables of X. Then, for relational model $\mathcal{M}_\Theta$, $P\big(X \mid nd(X), pa(X)\big) = P\big(X \mid pa(X)\big)$ if and only if $\forall x \in X|_b \; P\big(x \mid nd(x), pa(x)\big) = P\big(x \mid pa(x)\big)$ in parameterized ground graph $GG_{\mathcal{M}_\Theta \sigma}$ for all skeletons $\sigma \in \Sigma_\mathcal{S}$ and for all $b \in \sigma(B)$.

In other words, a relational variable X is independent of its non-descendants given its parents if and only if, for all possible parameterized ground graphs, the Markov condition holds for all instances of X. For Bayesian networks, the Markov condition is equivalent to d-separation [115]. Because parameterized ground graphs are Bayesian networks (implied by Lemma 4.5.1) and relational d-separation on abstract ground graphs is sound and complete (by Theorem 4.5.4), we can conclude that relational d-separation is equivalent to the relational Markov condition.

4.6 Hop Thresholds

For practical implementations, the size of the abstract ground graphs should be limited by a domain-specific threshold. In this section, we examine the effect of choosing a particular hop threshold (i.e., a singular hop threshold for all relational paths that are represented in an abstract ground graph).

First, we introduce the notion of (B, h)-reachability, which describes the conditions under which an edge in a ground graph is represented in an abstract ground graph.

Definition 4.6.1 ((B, h)-reachability) Let $GG_{\mathcal{M}\sigma}$ be the ground graph for some relational model structure $\mathcal{M}$ and skeleton $\sigma \in \Sigma_{\mathcal{S}}$. Then, $i_k.X \rightarrow i_j.Y \in GG_{\mathcal{M}\sigma}$ is (B, h)-*reachable* for perspective B and hop threshold h if there exist relational variables $P_k.X = [B, \ldots, I_k].X$ and $P_j.Y = [B, \ldots, I_j].Y$ such that $\mathit{length}(P_k) \leq h+1$, $\mathit{length}(P_j) \leq h+1$, and there exists an instance $b \in \sigma(B)$ with $i_k \in P_k|_b$ and $i_j \in P_j|_b$.

In other words, the edge $i_k.X \rightarrow i_j.Y$ in the ground graph is (B, h)-reachable if an instance of the base item $b \in \sigma(B)$ can reach i_k and i_j in at most h hops.

Example 4.6.1 Consider the ground graph shown in Figure 3.5. Let perspective B be Employee, and let the hop threshold $h = 6$. Let $i_k.X \rightarrow i_j.Y$ be the edge Laptop.*Success* $\rightarrow$ Devices.*Revenue* in the ground graph. This edge is (B, h)-reachable because of the following: Set $P_k.X$ = [Employee, Develops, Product].*Success*, $P_j.Y$ = [Employee, Develops, Product, Funds, Business-Unit].*Revenue*, and let b = Sally $\in \sigma$(Employee). We have $\mathit{length}(P_k) = 3 < 7$, $\mathit{length}(P_j) = 5 < 7$, Laptop $\in P_k|_{\text{Sally}}$, and Devices $\in P_j|_{\text{Sally}}$. $\square$

Since Definition 4.6.1 pertains to edges reachable via a particular perspective B and hop threshold h, it relates to the reachability of edges in abstract ground graphs. We denote abstract ground graphs for perspective B, limited by a hop threshold h as $AGG_{\mathcal{M}Bh}$. Definition 4.6.1 implies that (1) for every edge in ground graph $GG_{\mathcal{M}\sigma}$,

we can derive a set of abstract ground graphs for which that edge is (B, h)-reachable, and (2) for every abstract ground graph $AGG_{\mathcal{M}Bh}$, we can derive the set of (B, h)-reachable edges for a given ground graph. Given (B, h)-reachability, we can now express the soundness and completeness of abstract ground graphs.

Theorem 4.6.1 *For every acyclic relational model structure $\mathcal{M}$, perspective $B \in \mathcal{E} \cup \mathcal{R}$, and hop threshold $h_a \in \mathbb{N}^0$, the abstract ground graph $AGG_{\mathcal{M}Bh_a}$ is sound up to hop threshold h_a for all ground graphs $GG_{\mathcal{M}\sigma}$ with skeleton $\sigma \in \Sigma_{\mathcal{S}}$.*

Proof. Soundness means that for every edge $[B, \ldots, I_j].X \rightarrow [B, \ldots, I_k].Y$ in the abstract ground graph $AGG_{\mathcal{M}Bh_a}$, there exists a skeleton $\sigma \in \Sigma_{\mathcal{S}}$, a base item instance $b \in \sigma(B)$, an instance $i_j \in [B, \ldots, I_j]|b$, and an instance $i_k \in [B, \ldots, I_k]|b$ such that $i_j.X \rightarrow i_k.Y$ is a (B, h_a)-reachable edge in $GG_{\mathcal{M}\sigma}$. The proof is identical to the proof of soundness for Theorem 4.5.2. ■

Theorem 4.6.2 *For every acyclic relational model structure $\mathcal{M}$, perspective $B \in \mathcal{E} \cup \mathcal{R}$, and hop threshold $h_r \in \mathbb{N}^0$, the abstract ground graph $AGG_{\mathcal{M}Bh_a}$ is complete up to hop threshold h_r for all ground graphs $GG_{\mathcal{M}\sigma}$ with skeleton $\sigma \in \Sigma_{\mathcal{S}}$, where $h_a = \max(h_r + h_m, h_r + 2h_m - 2)$ and h_m is the maximum number of hops for a dependency in $\mathcal{M}$.*

Proof. Let $\mathcal{M} = (\mathcal{S}, \mathcal{D})$ be an arbitrary acyclic relational model structure, let $B \in \mathcal{E} \cup \mathcal{R}$ be an arbitrary perspective, and let $h_r \in \mathbb{N}^0$ be an arbitrary hop threshold.

To prove that the abstract ground graph $AGG_{\mathcal{M}Bh_a}$ is complete up to hop threshold h_r, we show that for every (B, h_r)-reachable edge $i_k.X \rightarrow i_j.Y$ in every ground graph $GG_{\mathcal{M}\sigma}$ with $\sigma \in \Sigma_{\mathcal{S}}$, there is a set of corresponding edges in $AGG_{\mathcal{M}Bh_a}$. Specifically, the (B, h_r)-reachable edge $i_k.X \rightarrow i_j.Y$ yields two sets of relational variables for some $b \in \sigma(B)$, namely $\mathbf{P_k}.\mathbf{X} = \{P_k.X \mid i_k \in P_k|_b \ \wedge \ \mathit{length}(P_k) \leq h_r + 1\}$ and $\mathbf{P_j}.\mathbf{Y} = \{P_j.Y \mid i_j \in P_j|_b \ \wedge \ \mathit{length}(P_j) \leq h_r + 1\}$, by Definition 4.6.1. Note that all

relational variables in both $\mathbf{P_k}.\mathbf{X}$ and $\mathbf{P_j}.\mathbf{Y}$ are nodes in $AGG_{\mathcal{MB}h_a}$. We show that for all $P_k.X \in \mathbf{P_k}.\mathbf{X}$ and for all $P_j.Y \in \mathbf{P_j}.\mathbf{Y}$ either (a) $P_k.X \rightarrow P_j.Y \in AGG_{\mathcal{MB}h_a}$, (b) $P_k.X \cap P'_k.X \rightarrow P_j.Y \in AGG_{\mathcal{MB}h_a}$ or $P_k.X \cap P'_k.X \rightarrow P'_j.Y \in AGG_{\mathcal{MB}h_a}$, where $i_k \in P'_k|_b$ and $i_j \in P'_j|_b$, or (c) $P_k.X \rightarrow P_j.Y \cap P'_j.Y \in AGG_{\mathcal{MB}h_a}$ or $P'_k.X \rightarrow P_j.Y \cap P'_j.Y \in AGG_{\mathcal{MB}h_a}$, where $i_k \in P'_k|_b$ and $i_j \in P'_j|_b$.

Let $\sigma \in \Sigma_{\mathcal{S}}$ be an arbitrary skeleton, let $i_k.X \rightarrow i_j.Y \in GG_{\mathcal{M}\sigma}$ be an arbitrary (B, h_r)-reachable edge drawn from $[I_j, \ldots, I_k].X \rightarrow [I_j].Y \in \mathcal{D}$ where $length([I_j, \ldots, I_k]) \leq h_m + 1$, and let $P_k.X \in \mathbf{P_k}.\mathbf{X}, P_j.Y \in \mathbf{P_j}.\mathbf{Y}$ be an arbitrary pair of relational variables. There are three cases:

(a) $P_k \in extend(P_j, [I_j, \ldots, I_k])$. Then, $length(P_k) \leq (h_r + 1) + (h_m + 1) - 1 = h_r + h_m + 1 \leq h_a + 1$. Therefore, $P_k.X$ is a node in the abstract ground graph, and $P_k.X \rightarrow P_j.Y \in AGG_{\mathcal{MB}h_a}$ by Definition 4.4.1.

(b) $P_k \notin extend(P_j, [I_j, \ldots, I_k])$, but $\exists P'_k \in extend(P_j, [I_j, \ldots, I_k])$ such that $i_k \in P'_k|_b$. Then, $length(P'_k) \leq (h_r+1)+(h_m+1)-1 = h_r+h_m+1 \leq h_a+1$. Therefore, P'_k is a node in the abstract ground graph, $P'_k.X \rightarrow P_j.Y \in AGG_{\mathcal{MB}h_a}$, and $P_k.X \cap P'_k.X \rightarrow P_j.Y \in AGG_{\mathcal{MB}h_a}$ by Definition 4.4.1.

(c) For all $P_k \in extend(P_j, [I_j, \ldots, I_k])$, it is the case that $i_k \notin P_k.X|_b$. Then by Lemma 4.4.2, there exists a P'_j such that $i_j \in P'_j|_b$ and there exists a $P''_k \in extend(P'_j, [I_j, \ldots, I_k])$. Given the way P'_j is constructed, its length is bounded by:

$$length(P'_j) \leq length(P_j) + length([I_j, \ldots, I_k]) - 3 \leq$$
$$(h_r + 1) + (h_m + 1) - 3 = h_r + h_m - 1$$

P''_k intersects with P_k since they both reach i_k, and the length of P''_k is bounded by:

$$length(P''_k) \leq length(P'_j) + length([I_j, \ldots, I_k]) - 1 \leq$$
$$(h_r + h_m - 1) + (h_m + 1) - 1 = h_r + 2h_m - 1$$

Also by Lemma 4.4.2, we know that P_j and P'_j intersect. Since $length(P''_k) \leq h_r + 2h_m - 1 \leq h_a + 1$, P''_k is a node in the abstract ground graph, $P''_k.X \rightarrow P'_j.Y \in$

$AGG_{\mathcal{M}Bh_a}$ $P''_k.X \rightarrow P'_j.Y \cap P_j.Y \in AGG_{\mathcal{M}Bh_a}$, and $P_k.X \cap P''_k.X \rightarrow P'_j.Y \in AGG_{\mathcal{M}Bh_a}$ by Definition 4.4.1.

From the above three cases, it follows that to guarantee completeness up to h_r, the abstract ground graph must contain nodes up to the hop threshold $h_a = \max(h_r + h_m, h_r + 2h_m - 2)$. ■

Theorems 4.6.1 and 4.6.2 guarantee that if an abstract ground graph is constructed with a hop threshold of h_a from perspective B, it captures all paths of dependence in all ground graphs, where (1) the variables along those paths are reachable in h_r hops from instances of B, and (2) the underlying dependencies are bounded by a threshold of h_m.

In the following, we say that d-separation holds up to a specified hop threshold h if there are no d-connecting paths involving relational variables of length greater than $h + 1$.

Theorem 4.6.3 *Relational d-separation is sound and complete for abstract ground graphs up to a specified hop threshold. Let $\mathcal{M}$ be an acyclic relational model structure, and let h_m be the maximum number of hops for a dependency in $\mathcal{M}$. Let $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ be three distinct sets of relational variables for perspective $B \in \mathcal{E} \cup \mathcal{R}$ defined over relational schema $\mathcal{S}$, and let h_r be the maximum number of hops of relational variables in $\mathbf{X}, \mathbf{Y}$, and $\mathbf{Z}$. Then, $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ are d-separated by $\bar{\mathbf{Z}}$ on the abstract ground graph $AGG_{\mathcal{M}Bh_a}$ if and only if for all skeletons $\sigma \in \Sigma_{\mathcal{S}}$ and for all $b \in \sigma(B)$, $\mathbf{X}|_b$ and $\mathbf{Y}|_b$ are d-separated by $\mathbf{Z}|_b$ up to hop threshold h_r in ground graph $GG_{\mathcal{M}\sigma}$, where $h_a = \max(h_r + h_m, h_r + 2h_m - 2)$.*

Proof. We must show that d-separation on an abstract ground graph implies d-separation on all ground graphs it represents (soundness) and that d-separation facts that hold across all ground graphs are also entailed by d-separation on the abstract ground graph (completeness).

Soundness: Assume that $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ are d-separated by $\bar{\mathbf{Z}}$ on $AGG_{\mathcal{M}Bh_a}$. Assume for contradiction that there exists a skeleton $\sigma \in \Sigma_{\mathcal{S}}$ and an item instance $b \in \sigma(B)$ such that $\mathbf{X}|_b$ and $\mathbf{Y}|_b$ are *not* d-separated by $\mathbf{Z}|_b$ in the ground graph $GG_{\mathcal{M}\sigma}$. Then, there must exist a d-connecting path p from some $x \in \mathbf{X}|_b$ to some $y \in \mathbf{Y}|_b$ given all $z \in \mathbf{Z}|_b$ such that every edge of p is (B, h_r)-reachable. By Theorem 4.6.2, $AGG_{\mathcal{M}Bh_a}$ is (B, h_r)-reachably complete, so all (B, h_r)-reachable edges in $GG_{\mathcal{M}\sigma}$ are captured by edges in $AGG_{\mathcal{M}Bh_a}$. Thus, path p must be represented from some node in $\{N_x \mid x \in N_x|_b\}$ to some node in $\{N_y \mid y \in N_y|_b\}$, where N_x, N_y are nodes in $AGG_{\mathcal{M}Bh_a}$. If p is d-connecting in $GG_{\mathcal{M}\sigma}$, then it is d-connecting in $AGG_{\mathcal{M}Bh_a}$, which implies that $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ are *not* d-separated by $\bar{\mathbf{Z}}$. Therefore, $\mathbf{X}|_b$ and $\mathbf{Y}|_b$ must be d-separated by $\mathbf{Z}|_b$.

Completeness: Assume that $\mathbf{X}|_b$ and $\mathbf{Y}|_b$ are d-separated by $\mathbf{Z}|_b$ in the ground graph $GG_{\mathcal{M}\sigma}$ for all skeletons $\sigma \in \Sigma_{\mathcal{S}}$ and for all $b \in \sigma(B)$. Assume for contradiction that $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ are *not* d-separated by $\bar{\mathbf{Z}}$ on $AGG_{\mathcal{M}Bh_a}$. Then, there must exist a d-connecting path p for some relational variable $X \in \bar{\mathbf{X}}$ to some $Y \in \bar{\mathbf{Y}}$ given all $Z \in \bar{\mathbf{Z}}$. By Theorem 4.6.1, $AGG_{\mathcal{M}Bh_a}$ is (B, h_a)-reachably sound, so every edge in $AGG_{\mathcal{M}Bh}$ must correspond to some pair of variables in some ground graph. Thus, if p is d-connecting in $AGG_{\mathcal{M}Bh_a}$, then there must exist some skeleton σ such that p is d-connecting in $GG_{\mathcal{M}\sigma}$ for some $b \in \sigma(B)$, which implies that d-separation does not hold for that ground graph. Therefore, $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ must be d-separated by $\bar{\mathbf{Z}}$ on $AGG_{\mathcal{M}Bh_a}$. ■

4.7 Naïve Relational *d*-Separation Is Frequently Incorrect

If the rules of d-separation for Bayesian networks were applied directly to the structure of relational models, how frequently would the conditional independence conclusions be correct? In this section, we evaluate the necessity of our approach—relational d-separation executed on abstract ground graphs. We empirically compare

the consistency of a naïve approach against our sound and complete solution over a large space of synthetic causal models. To promote a fair comparison, we restrict the space of relational models to those with underlying dependencies that could feasibly be represented and recovered by a naïve approach. We describe this space of models, present a reasonable approach for applying traditional d-separation to the structure of relational models, and quantify its decrease in expressive power and accuracy.

4.7.1 Defining a naïve approach

Consider the following limited definition of relational paths, which itself limits the space of models and conditional independence queries. A *simple relational path* $P = [I_j, \ldots, I_k]$ for relational schema $\mathcal{S}$ is a relational path such that $I_j \neq \cdots \neq I_k$. The sole difference between relational paths (Definition 3.3.1) and simple relational paths is that no item class may appear more than once along the latter. This yields paths drawn directly from a schema diagram. For the example in Figure 3.1, [Employee, Develops, Product] is simple whereas [Employee, Develops, Product, Develops, Employee] is not.

Additionally, we define *simple relational schemas* such that, for any two item classes $I_j, I_k \in \mathcal{E} \cup \mathcal{R}$, there exists at most one simple relational path between them (i.e., no cycles occur in the schema diagram). The example in Figure 3.1 is a simple relational schema. The restriction to simple relational paths and schemas yields similar definitions for *simple relational variables*, *simple relational dependencies*, and *simple relational models*. The relational model in Figure 3.4 is simple because it includes only simple relational dependencies.

A first approximation to relational d-separation would be to apply the rules of traditional d-separation directly to the graphical representation of relational models. This is equivalent to applying d-separation to the class dependency graph $G_{\mathcal{M}}$ (see Definition 3.6.2) of relational model $\mathcal{M}$. The class dependency graph for the model

in Figure 3.4 is shown in Figure 4.7(a). Note that the class dependency graph ignores path designators on dependencies, does not include all implications of dependencies among arbitrary relational variables, and does not represent intersection variables.

Although the class dependency graph is independent of perspectives, testing any conditional independence fact *requires* choosing a perspective. All relational variables must have a common base item class; otherwise, no method can produce a single consistent, propositional table from a relational database. For example, consider the construction of a table describing employees with columns for their salary, the success of products they develop, and the revenue of the business units they operate under. This procedure requires joining the instances of three relational variables ([EMPLOYEE].*Salary*, [EMPLOYEE, DEVELOPS, PRODUCT].*Success*, and [EMPLOYEE, DEVELOPS, PRODUCT, FUNDS, BUSINESS-UNIT].*Revenue*) for every common base item instance, from Paul to Thomas. See, for example, the resulting propositional table for these relational variables and an example query in Table 3.1 and Figure 3.7, respectively. An individual relational variable requires joining the item classes within its relational path, but combining a collection of relational variables requires joining on their common base item class. Fortunately, given a perspective and the space of simple relational schemas and models, a class dependency graph is equivalent to a *simple abstract ground graph*.

Definition 4.7.1 (Simple abstract ground graph) For simple relational model $\mathcal{M} = (\mathcal{S}, \mathcal{D})$ and perspective $B \in \mathcal{E} \cup \mathcal{R}$, the *simple abstract ground graph* $AGG^s_{\mathcal{M}B}$ is the directed acyclic graph (V, E) that abstracts the dependencies $\mathcal{D}$ among simple relational variables. The nodes consist of simple relational variables

$$\big\{[B, \dots, I_j].X \mid B \neq \cdots \neq I_j\big\},$$

and the edges connect those nodes

$$\big\{[B, \dots, I_k].Y \to [B, \dots, I_j].X \mid [I_j, \dots, I_k].Y \to [I_j].X \in \mathcal{D}$$
$$\wedge\ [B, \dots, I_k] \in extend([B, \dots, I_j], [I_j, \dots, I_k])\ \wedge\ [B, \dots, I_k].Y, [B, \dots, I_j].X \in V\big\}.$$

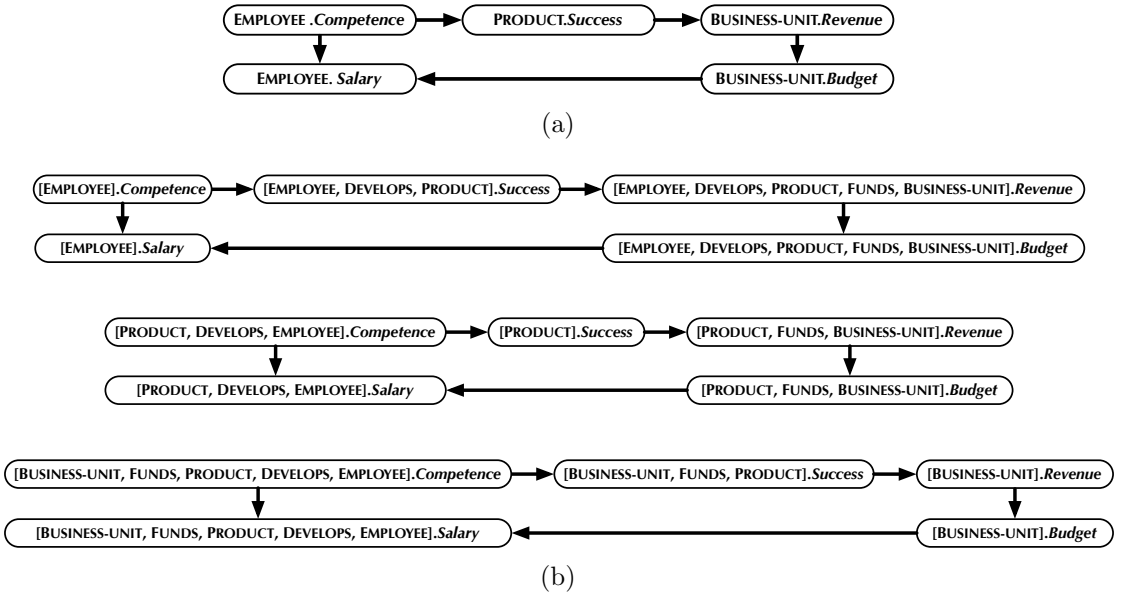


Figure 4.7: For the model in Figure 3.4, (a) the class dependency graph and (b) three simple abstract ground graphs for the EMPLOYEE, PRODUCT, and BUSINESS-UNIT perspectives.

Simple abstract ground graphs only include nodes for simple relational variables and necessarily exclude intersection variables. Lemma 3.4.1—which characterizes the intersection between a pair of relational paths—only applies to pairs of *simple* relational paths if the schema contains cycles, which is not the case for simple relational schemas by definition. As a result, the simple abstract ground graph for a given schema and model contains the same number of nodes and edges, regardless of perspective; the nodes simply have path designators redefined from the given perspective. Figure 4.7(b) shows three simple abstract ground graphs from distinct perspectives for the model in Figure 3.4. As noted above, simple abstract ground graphs are qualitatively the same as the class dependency graph, but they enable answering relational d-separation queries, which requires a common perspective in order to be semantically meaningful.

The naïve approach to relational d-separation derives conditional independence facts from simple abstract ground graphs (Definition 4.7.1). The sound and com-

plete approach described in this paper applies d-separation—with input variable sets augmented by their intersection variables—to "regular" abstract ground graphs, as described by Definition 4.4.1. Clearly, if d-separation on a simple abstract ground graph claims that $\mathbf{X}$ is d-separated from $\mathbf{Y}$ given $\mathbf{Z}$, then d-separation on the regular abstract ground graph yields the same conclusion if and only if there are no d-connecting paths in the regular abstract ground graph. Either $\mathbf{X}$ and $\mathbf{Y}$ can be d-separated by a set of simple relational variables $\mathbf{Z}$, or they require non-simple relational variables—those involving relational paths with repeated item classes.[6]

4.7.2 Evaluating the necessity of abstract ground graphs

To evaluate the necessity of regular abstract ground graphs (i.e., the additional paths involving non-simple relational variables and intersection variables), we compared the frequency of equivalence between the conclusions of d-separation on simple and regular abstract ground graphs. The two approaches are only equivalent if a minimal separating set consists entirely of simple relational variables.[7]

Thus, for an arbitrary pair of relational variables X and Y with a common perspective, we test the following on regular abstract ground graphs:

1. Is either X or Y a non-simple relational variable?
2. Are X and Y marginally independent?
3. Does a minimal separating set $\mathbf{Z}$ d-separate X and Y, where $\mathbf{Z}$ consists solely of simple relational variables?
4. Is there any separating set $\mathbf{Z}$ that d-separates X and Y?

---

[6]The theoretical conditions under which equivalence occurs are sufficiently complex that they provide little utility as they essentially require reconstructing the regular abstract ground graph and checking a potentially exponential number of dependency paths.

[7]If $\mathbf{X}$ and $\mathbf{Y}$ are d-separated given $\mathbf{Z}$, then $\mathbf{Z}$ is a *separating set* for $\mathbf{X}$ and $\mathbf{Y}$. A separating set $\mathbf{Z}$ is *minimal* if there is no proper subset of $\mathbf{Z}$ that is also a separating set.

If the answer to (1) is yes, then the naïve approach cannot apply since either X or Y is undefined for the simple abstract ground graph. If the answer to (2) is yes, then there is equivalence; this is a trivial case because there are no d-connecting paths for $\mathbf{Z} = \emptyset$. If the answer to (3) is yes, then there is a minimal separating set $\mathbf{Z}$ consisting of only simple relational variables. In this case, the simple abstract ground graph is sufficient, and we also have equivalence. If the answer to (4) is no, then no separating set $\mathbf{Z}$, simple or otherwise, renders X and Y conditionally independent.

We randomly generated simple relational schemas and models for 100 trials for each setting using the following parameters:

- Number of entity classes, ranging from 1 to 4.
- Number of relationship classes, fixed at one less than the number of entities, ensuring simple, connected relational schemas. Relationship cardinalities are chosen uniformly at random.
- Number of attributes for each entity and relationship class, randomly drawn from a shifted Poisson distribution with $\lambda = 1.0$ ($\sim Pois(1.0) + 1$).
- Number of dependencies in the model, ranging from 1 to 10.

Then, for all pairs of relational variables with a common perspective limited by a hop threshold of $h = 4$, we ran the aforementioned tests against the regular abstract ground graph, limiting its relational variables by a hop threshold of $h = 8$.

This procedure generated a total of almost 3.6 million pairs of relational variables to test. Approximately 56% included a non-simple relational variable; the naïve approach cannot be used to derive a conditional independence statement in these cases, requiring the full abstract ground graph in order to represent these variables. Of the remaining 44% (roughly 1.6 million), 82% were marginally independent, and 9% were not conditionally independent given any conditioning set $\mathbf{Z}$. Then, of the remaining

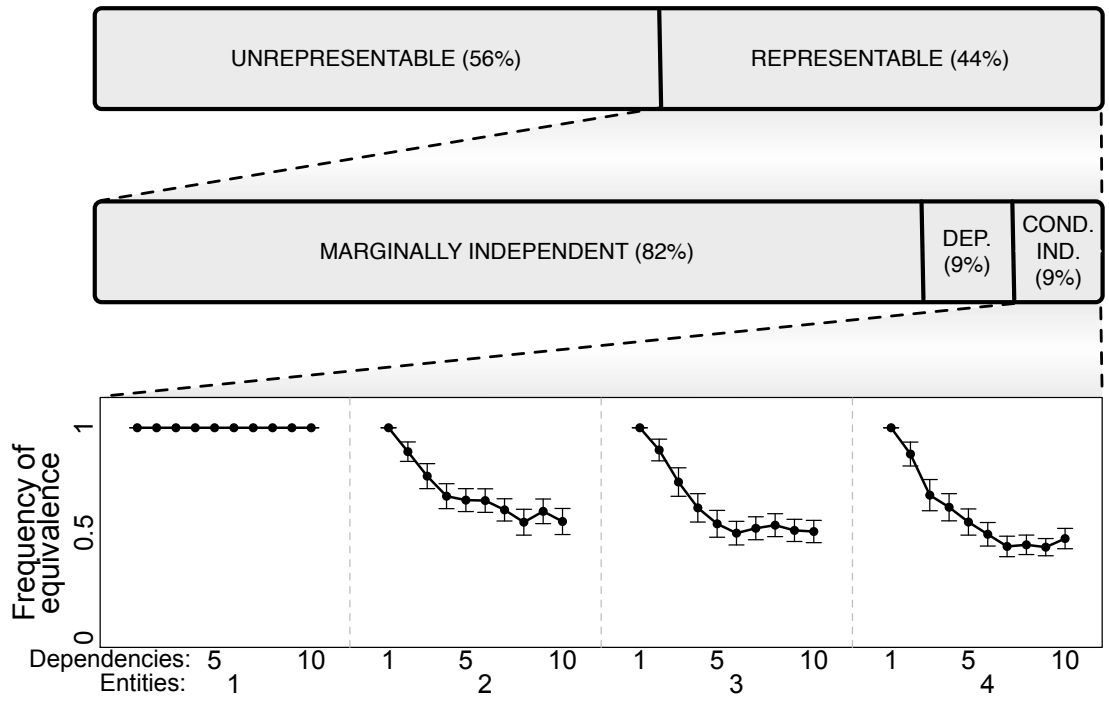


Figure 4.8: The majority (56%) of generated relational d-separation queries are not representable with the naïve approach. Of the 44% that are representable (involving only simple relational variables), 82% are marginally independent and 9% are dependent. Pairs of relational variables in the remaining 9% are conditionally independent given a non-empty separating set ($X \perp\!\!\!\perp Y \mid \mathbf{Z}$, where $\mathbf{Z} \neq \emptyset$). We test whether the *conditioning set* consists solely of simple relational variables. If so, then the naïve approach to relational d-separation is equivalent to d-separation on fully specified abstract ground graphs. This graph plots the frequency of equivalence across schemas with increasing numbers of entity classes (1–4) for varying numbers of dependencies (1–10). For schemas with more than one entity class, the frequency of equivalence decreases as the number of dependencies increases. Shown with 95% confidence intervals.

9% (roughly 145,000), we can test the frequency of equivalence for conditional independence facts with non-empty separating sets—the proportion of cases for which only simple relational variables are required in a minimal separating set $\mathbf{Z}$.

Figure 4.8 shows this frequency for schemas of increasing numbers of entity classes (1–4) for varying numbers of dependencies in the causal model (1–10). Since relational schemas with a single entity class and no relationships describe propositional data, the simple abstract ground graph is equivalent to the full abstract ground graph, which is also equivalent to the model itself. In this case, the naïve approach is always equivalent because it is exactly d-separation on Bayesian networks. For truly relational schemas (with more than one entity class and at least one relationship class), the frequency of equivalence decreases as the number of dependencies in the model increases. Additionally, the frequency of equivalence decreases more as the number of entities in the schema increases. For example, the frequency of equivalence for nine dependencies is 60.3% for two entities, 51.2% for three entities, and 43.2% for four entities.

We also learned statistical models that predict the number of equivalent and non-equivalent statements in order to identify key factors that affect the frequency of equivalence. We found that the number of dependencies and size of the relational model (regulated by the number of entities and MANY cardinalities) dictate the equivalence. As a relational model deviates from a Bayesian network, we should expect more d-connecting paths in the regular but not simple abstract ground graph. This property also depends on the specific combination of dependencies in the model. Section 4.7.3 presents details of this analysis.

This experiment suggests that applying traditional d-separation directly to a relational model structure will frequently derive incorrect conditional independence facts. Additionally, there is a large class of conditional independence queries involving non-simple variables for which such an approach is undefined. These results indicate that

fully specifying abstract ground graphs and applying d-separation augmented with intersection variables (as described in Section 4.3) is critical for accurately deriving most conditional independence facts from relational models.

4.7.3 Experimental details: Equivalence of a naïve approach

The main goal of this experiment is to quantify how often traditional d-separation applied directly to relational model structures produces incorrect conditional independence facts. This provides a rough measurement for the additional representational power of relational d-separation on abstract ground graphs. Here, we present an analysis of which factors influence the number of equivalent and non-equivalent conditional independence judgments between both approaches (naïvely applying traditional d-separation versus relational d-separation).

Specifically, we show here the results of running log-linear regression to predict the number of equivalent and non-equivalent judgments for varying schemas and models. We first applied lasso for feature selection [184] to minimize the number of predictors while maximizing model fit. We also standardized the input variables by dividing by two standard deviations, as recommended by Gelman [50]. Since the predictor for the number of dependencies is log-transformed, the standardization of that variable occurs after taking the logarithm.

In predicting the (log of the) number of equivalent conditional independencies, the following variables were significantly and substantively predictive (in order of decreasing predictive power):

- Interaction between the log of the number of dependencies and the number of entities (positive)
- Log of the number of dependencies (positive)
- Interaction between the log of the number of dependencies and the number of MANY cardinalities (negative)

Predictor	Coefficient	Partial	Semipartial
log(# dependencies) × # entities	1.38	0.232	0.085
log(# dependencies)	1.14	0.135	0.044
log(# dependencies) × # MANY cardinalities	-0.71	0.092	0.028
# entities × # relational variables	-0.32	0.044	0.013

Table 4.1: Number of equivalent conditional independence judgments: estimated standardized coefficient, squared partial correlation coefficient, and squared semipartial correlation coefficient for each predictor.

Predictor	Coefficient	Partial	Semipartial
# MANY cardinalities × # entities	-2.22	0.207	0.064
log(# dependencies) × # entities	0.90	0.165	0.048
# MANY cardinalities	3.24	0.128	0.036
log(# dependencies) × # MANY cardinalities	1.47	0.127	0.036

Table 4.2: Number of non-equivalent conditional independence judgments: estimated standardized coefficient, squared partial correlation coefficient, and squared semipartial correlation coefficient for each predictor.

- Number of entities (negative)
- Interaction between the number of entities and the number of relational variables in the AGG (negative)

The fit for the equivalent model has an $R^2 = 0.721$ for $n = 4,000$, and Table 4.1 contains the standardized coefficients as well as the squared partial and semipartial correlation coefficients for each predictor. For lasso, $\lambda = 0.0076$ offered the fewest predictors while increasing the model fit by at least 0.01.

In predicting the (log of the) number of non-equivalent conditional independencies, the following variables were significantly and substantively predictive (in order of decreasing predictive power):

- Interaction between the number of MANY cardinalities and the number of entities (negative)

- Interaction between the log of the number of dependencies and the number of entities (positive)
- Number of MANY cardinalities (positive)
- Interaction between the log of the number of dependencies and the number of MANY cardinalities (positive)

The fit for the non-equivalent model has an $R^2 = 0.755$ for $n = 4,000$, and Table 4.2 contains the standardized coefficients and the squared partial and semipartial correlation coefficients for each predictor. For lasso, $\lambda = 0.0155$ offered the fewest predictors while increasing the model fit by at least 0.01.

4.8 Experiments

To complement the theoretical results, we present three experiments on synthetic data. The primary goal of these empirical tests is to demonstrate the feasibility of applying relational d-separation in practice. The experiment in Section 4.8.1 describes the factors that influence the size of abstract ground graphs and thus the computational complexity of relational d-separation. The experiment in Section 4.8.2 evaluates the growth rate of separating sets produced by relational d-separation as abstract ground graphs become large. The results indicate that minimal separating sets grow much more slowly than abstract ground graphs. The experiment in Section 4.8.3 tests how the expectations of the relational d-separation theory match statistical conclusions on simulated data. As expected from the proofs of correctness in Section 4.5, the results indicate a close match, aside from Type I errors and certain biases of conventional statistical tests on relational data.

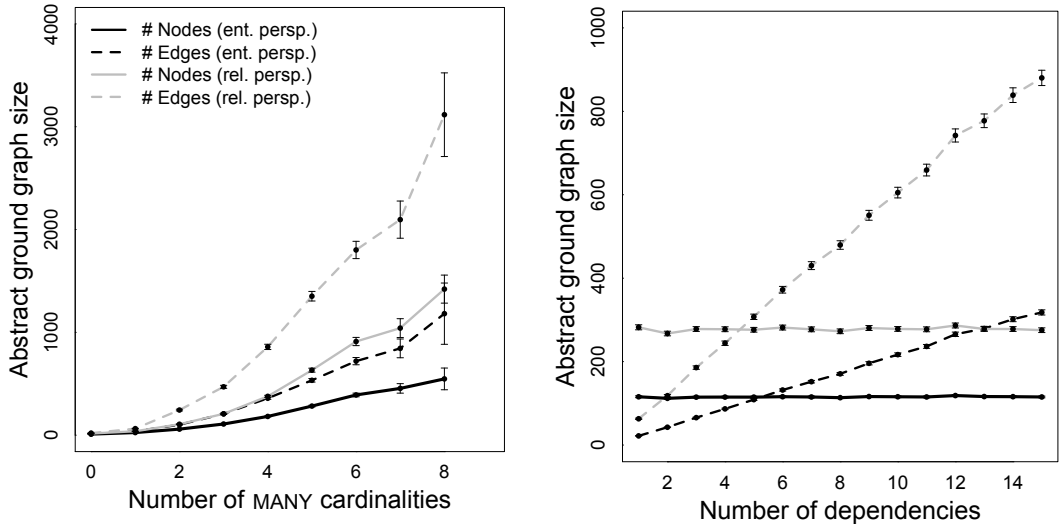
4.8.1 Abstract ground graph size

Relational d-separation is executed on abstract ground graphs. Consequently, it is important to quantify the size of abstract ground graphs and identify which factors influence their size. We randomly generated relational schemas and models for 1,000 trials of each setting using the following parameters:

- Number of entity classes, ranging from 1 to 4.
- Number of relationship classes, ranging from 0 to 4. The schema is guaranteed to be fully connected and includes at most a single relationship between a pair of entities. Relationship cardinalities are selected uniformly at random.
- Number of attributes for each entity and relationship class, randomly drawn from a shifted Poisson distribution with $\lambda = 1.0$ ($\sim Pois(1.0) + 1$).
- Number of dependencies in the model, ranging from 1 to 15.

This procedure generated a total of 450,000 abstract ground graphs, which included every perspective (all entity and relationship classes) for each experimental combination. We measure size as the number of nodes and edges in a given abstract ground graph. Figure 4.9(a) depicts how the size of abstract ground graphs varies with respect to the number of MANY cardinalities in the schema (fixed for models with 10 dependencies), and Figure 4.9(b) shows how it varies with respect to the number of dependencies in the model. Recall that for a single entity, abstract ground graphs are equivalent to Bayesian networks.

To determine the most influential factors of abstract ground graph size, we ran log-linear regression using independent variables that describe only the schema and model. Detailed results are provided in Section 4.8.4. This analysis indicates that (1) as the number of entities, relationships, attributes, and MANY cardinalities increases, the number of nodes and edges grows at an exponential rate; (2) as the number of



(a) (b)

Figure 4.9: Variation of abstract ground graph size as (a) the number of MANY cardinalities in the schema increases (dependencies fixed at 10) and (b) the number of dependencies increases. Shown with 95% confidence intervals.

dependencies in the model increases, the number of edges increases linearly, but the number of nodes remains invariant; and (3) abstract ground graphs for relationship perspectives are larger than entity perspectives because more relational variables can be defined.

4.8.2 Minimal separating set size

Because abstract ground graphs can become large, one might expect that separating sets could also grow to impractical sizes. Fortunately, relational d-separation produces minimal separating sets that are empirically observed to be small. We ran 1,000 trials of each setting using the following parameters:

- Number of entity classes, ranging from 1 to 4.
- Number of relationship classes, fixed at one less than the number of entities. Relationship cardinalities are selected uniformly at random.
- Total number of attributes across entity and relationship classes, fixed at 10.

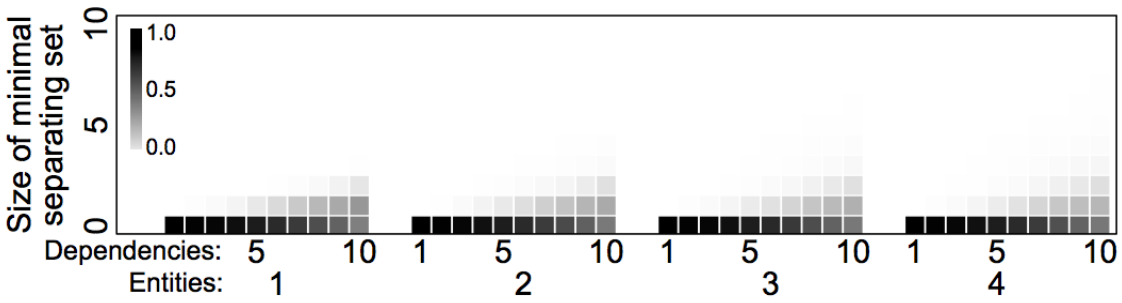


Figure 4.10: Minimal separating sets have reasonable sizes, growing only with the size of the schema and the model density. In this experiment, 99.9% of variable pairs have a minimal separating set with five or fewer variables.

- Number of dependencies in the model, ranging from 1 to 10.

For each relational model, we identified a single minimal separating set for up to 100 randomly chosen pairs of conditionally independent relational variables. This procedure generated almost 2.5 million pairs of variables.

To identify a minimal separating set between relational variables X and Y, we modified Algorithm 4 devised by Tian et al. [182] by starting with all parents of $\bar{X}$ and $\bar{Y}$, the variables augmented with the intersection variables they subsume in the abstract ground graph. While the discovered separating sets are *minimal*, they are not necessarily of *minimum* size because of the greedy process for removing conditioning variables from the separating set. Figure 4.10 shows the frequency of separating set size as both the number of entities and dependencies vary. In summation, roughly 83% of the pairs are marginally independent (having empty separating sets), 13% have separating sets of size one, and less than 0.1% have separating sets with more than five variables. The experimental results indicate that separating set size is strongly influenced by model density, primarily because the number of potential d-connecting paths increases as the number of dependencies increases.

4.8.3 Empirical validity

As a practical demonstration, we examined how the expectations of the relational d-separation theory match the results of statistical tests on actual data. We used a standard procedure for empirically measuring internal validity of algorithms. In this case, we (1) randomly generated a relational schema, (2) randomly generated a relational model structure for that schema, (3) parameterized the model structure, (4) generated synthetic data according to the model structure and parameters, (5) randomly chose relational d-separation queries according to the known ground-truth model, and (6) compared the model theory (i.e., the d-separation conclusions) against corresponding statistical tests of conditional independence.

For steps (1) and (2), we randomly generated a relational schema $\mathcal{S}$ and relational model structure $\mathcal{M}$ for $\mathcal{S}$ for 100 trials using the following settings:

- Number of entity classes, ranging from 1 to 4.
- Number of relationship classes, fixed at one less than the number of entities. Relationship cardinalities are selected uniformly at random.
- Number of attributes for each entity and relationship class, randomly drawn from a shifted Poisson distribution with $\lambda = 1.0$ ($\sim Pois(1.0) + 1$).
- Number of dependencies in the model, fixed at 10.

Dependencies were selected greedily, choosing each one uniformly at random, subject to a maximum of 3 parent relational variables for each attribute $[I_j].X$ and enforcing acyclicity of the model structure.

For step (3), we parameterized relational models using simple additive linear equations with independent, normally distributed error and the average aggregate for relational variable instances. For each attribute $[I_j].X$, we assign a conditional probability distribution

$$\sum_{[I_j,\ldots,I_k].Y \in parents([I_j].X)} \big(\beta \cdot avg([I_j,\ldots,I_k].Y)\big) + 0.1\epsilon$$

if $[I_j].X$ has parents, where

$$\beta = \frac{0.9}{|parents([I_j].X)|}$$

to provide equal contribution for each direct cause and $\epsilon \sim N(0,1)$ (error drawn from a standard normal distribution). If $[I_j].X$ has no parents, its value is just drawn from ϵ.

For step (4), we first generated a relational skeleton σ (because the current model space assumes that attributes do not cause entity or relationship existence) and then populated each attribute value by drawing from its corresponding conditional distribution. Each entity class is initialized to 1,000 instances. Relationship instances were constructed via a latent homophily process, similar to the method used by Shalizi and Thomas [161]. Each entity instance received a single latent variable, marginally independent from all other variables. The probability of any relationship instance was drawn from

$$\frac{e^{-\alpha d}}{1 + e^{-\alpha d}},$$

the inverse logistic function, where $d = |L_{E_1} - L_{E_2}|$, the difference between the latent variables on the two entities, and $\alpha = 10$, set as the decay parameter. We also scaled the probabilities in order to produce an expected degree of five for each entity instance when the cardinality of the relationship is MANY. Since the latent variables are marginally independent of all others, they are safely omitted from abstract ground graphs; their sole purpose is to generate relational skeletons that provide a greater probability of non-empty intersection variables as opposed to a random underlying link structure. We generated 100 independent relational skeletons and attribute values (i.e., 100 instantiated relational databases) for each schema and model.

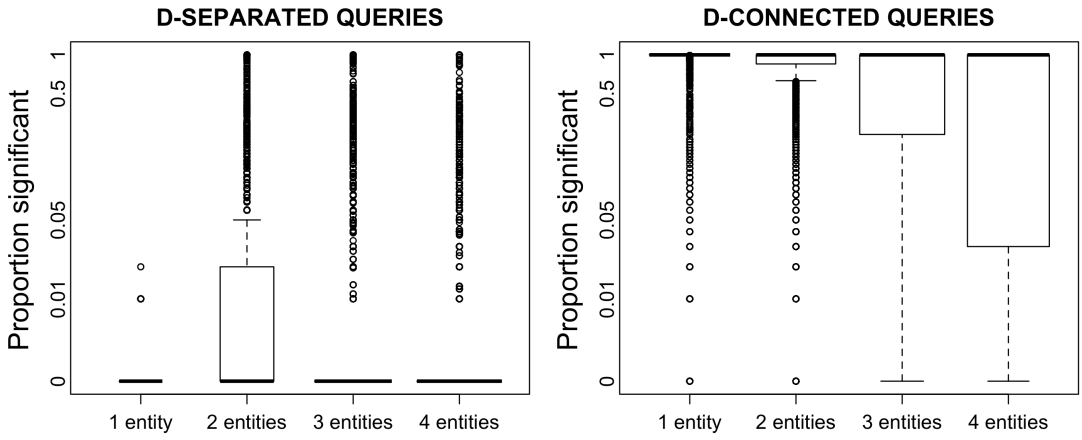


Figure 4.11: The proportion of significant trials for statistical tests of conditional independence on actual data. (Left) Evaluating queries that the model claims to be d-separated produces low rates of significant effects. (Right) Queries that the model claims are d-connected produce high rates of significant effects. Note that the generative process yields denser models for 2 entity classes since the number of dependencies is fixed at 10.

Step (5) randomly chose up to 100 true and false relational d-separation queries for a given model.[8] Since we have the ground-truth model, we can evaluate with our approach (abstract ground graphs and relational d-separation) whether these queries are true (d-separated) or false (d-connected). Each query is of the form $X \perp\!\!\!\perp Y \mid \mathbf{Z}$ such that X and Y are single relational variables, $\mathbf{Z}$ is a set of relational variables, Y has a singleton relational path (e.g., $[I_k].Y$), and all variables are from a common perspective. These queries correspond to testing potential direct causal dependencies in the relational model, similar to the tests used by constraint-based methods for learning relational models, such as RPC [102] and RCD [99].

Finally, step (6) tested conditional independence for all such $\langle X, Y, \mathbf{Z} \rangle$ d-separation queries using linear regression (because the models were parameterized linearly) for each of the 100 data instantiations. Specifically, we tested the t-statistic for the

[8]Depending on the properties of the schema and model, it may not always be feasible to identify 100 true or false d-separation statements.

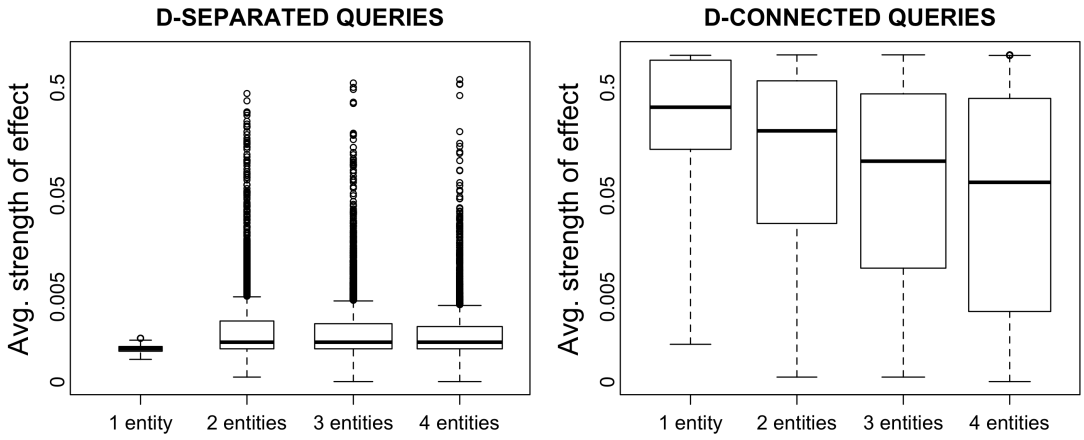


Figure 4.12: The average strength of effect of each query (measured as squared partial correlation) on actual data. (Left) Evaluating queries that the model claims to be d-separated or conditionally independent produces low average effect sizes. (Right) Queries that the model claims are d-connected or dependent produce high average effect sizes.

coefficient of $avg(X)$ in the equation $Y = \beta_0 + \beta_1 \cdot avg(X) + \sum_{Z_i \in \mathbf{Z}} \beta_i \cdot avg(Z_i)$. For each query, we recorded two measurements:

- The average strength of effect, measured as squared partial correlation—the proportion of remaining variance of Y explained by X after conditioning on $\mathbf{Z}$
- The proportion of trials for which each query was deemed significant at $\alpha = 0.01$ adjusted using Bonferroni correction with the number of queries per trial

Figure 4.11 shows the distribution of the proportion of significant trials for both true (left) and false (right) queries for varying numbers of entities. Figure 4.12 shows the corresponding average strength of effects for true (left) and false (right) queries. The graph uses a standard box-and-whisker plot with values greater or less than 1.5 times the inner quartile range—the difference between the upper and lower quartiles—marked as outliers.

In the vast majority of cases, relational d-separation is consistent with tests on actual data (i.e., most d-separated queries have low effect sizes and are rarely deemed

significant, whereas most d-connected queries have high effect sizes and are mostly deemed significant). For approximately 23,000 true queries, 14.9% are significant in more than one trial, but most are insubstantive, with only 2.2% having an average effect size greater than 0.01. There are three potential reasons why a d-separation in theory may appear to be d-connected in practice: (1) Type I error; (2) high power given a large sample size; or (3) bias. A small number of cases exhibit an interaction between aggregation and relational structure (i.e., degree or the cardinality of relational variable instances). This interaction violates the identically distributed assumption of data instances, which produces a biased estimate of effect size for simple linear regression. Linear regression does not account for these interaction effects, suggesting the need for more accurate statistical tests of conditional independence for relational data.

4.8.4 Experimental details: Abstract ground graph size

The goal of the experiment in Section 4.8.1 is to determine which factors influence the size of abstract ground graphs because the computational complexity of relational d-separation depends on their size. Specifically, we show here the results of running log-linear regression to predict the size of abstract ground graphs for varying schemas and models. We first applied lasso for feature selection [184] to minimize the number of predictors while maximizing model fit. We also standardized the input variables by dividing by two standard deviations, as recommended by Gelman [50]. Since the predictor for the number of dependencies is log-transformed, the standardization for that variable occurs after taking the logarithm.

In predicting the (log of the) number of nodes, the following variables were significantly and substantively predictive (in order of decreasing predictive power):

- Number of relationships (positive)

Predictor	Coefficient	Partial	Semipartial
# relationships	3.24	0.452	0.150
# MANY cardinalities × isEntity=F	3.09	0.349	0.109
# entities	-2.11	0.359	0.102
# MANY cardinalities × isEntity=T	2.51	0.216	0.053
# MANY cardinalities × # relationships	-0.88	0.100	0.020
# attributes	0.23	0.024	0.004

Table 4.3: Number of nodes in an abstract ground graph: estimated standardized coefficient, squared partial correlation coefficient, and squared semipartial correlation coefficient for each predictor.

- Interaction between MANY cardinalities and an indicator variable for whether the abstract ground graph is from an entity or relationship perspective (positive)
- Number of entities (negative)
- Interaction between number of MANY cardinalities and relationships (negative)
- Total number of attributes (positive)

The fit for the nodes model has an $R^2 = 0.818$ for $n = 450,000$, and Table 4.3 contains the standardized coefficients as well as the squared partial and semipartial correlation coefficients for each predictor. For lasso, $\lambda = 0.0095$ offered the fewest predictors while increasing the model fit by at least 0.01.

In predicting the (log of the) number of edges, the following variables were significantly and substantively predictive (in order of decreasing predictive power):

- Log of the number of dependencies (positive)
- Number of relationships (positive)
- Interaction between MANY cardinalities and an indicator variable for whether the abstract ground graph is from an entity or relationship perspective (positive)
- Number of entities (negative)

Predictor	Coefficient	Partial	Semipartial
log(# dependencies)	1.44	0.440	0.165
# relationships	3.86	0.395	0.138
# MANY cardinalities × isEntity=F	4.27	0.356	0.123
# entities	-2.78	0.353	0.115
# MANY cardinalities × isEntity=T	3.52	0.231	0.067
# MANY cardinalities × # relationships	-1.35	0.127	0.031

Table 4.4: Number of edges in an abstract ground graph: estimated standardized coefficient, squared partial correlation coefficient, and squared semipartial correlation coefficient for each predictor.

- Interaction between number of MANY cardinalities and relationships (negative)

The fit for the edges model has an $R^2 = 0.789$ for $n = 450,000$, and Table 4.4 contains the standardized coefficients and the squared partial and semipartial correlation coefficients for each predictor. For lasso, $\lambda = 0.0164$ offered the fewest predictors while increasing the model fit by at least 0.01.

4.9 Concluding Remarks

Chapter 3 upgraded Bayesian networks for propositional data to directed graphical models of relational data. As it turns out, even though the ground graphs of relational models have the same semantics as Bayesian networks, the conditional independencies implied by traditional d-separation on the structure of relational models does not necessarily transfer to *all* ground graphs. In fact, the comparison in Section 4.7 showed that up to 50% of naïvely derived conditional independencies are incorrect. Thus, we introduced the notion of relational d-separation that, analogous to traditional d-separation, only claims independence if it holds in all ground graphs (Section 4.3). For Bayesian networks, this requirement is trivial because of the one-to-one structural correspondence between model structure and ground graph instances. Such a correspondence fails to hold for relational models.

As a result, we introduced the abstract ground graph—a new representation that encodes the dependence implications of relational models among all relational variables for any perspective (Section 4.4). The abstract ground graph is aptly named because it *abstracts* all possible ground graphs. This enables the sound and complete derivation of conditional independence facts from relational models in the limit (Section 4.5) and for finite relational paths (Section 4.6).

In the next chapter, we present the first sound and complete algorithm for learning causal structure from relational data. This algorithm leverages the theoretical foundation developed in this chapter. Specifically, the theory of relational d-separation connects conditional independence with relational causal structure, and the abstract ground graph provides an underlying representation to reason about causal structure.

CHAPTER 5

LEARNING

Research in causal discovery has led to the identification of fundamental principles and methods for causal inference, including a *complete* algorithm—the PC algorithm—that identifies all possible orientations of causal dependencies from observed conditional independencies [122, 172, 106]. Completeness guarantees that no other method, under the same assumptions, can infer more causal dependencies from observational data. However, much of this work, including the completeness result, applies only to propositional data and Bayesian networks.

As described throughout Chapter 3, there are more expressive classes of models, including probabilistic relational models [53] and the one formalized in Section 3.5, that remove the assumption of independent and identically distributed instances required by Bayesian networks. These *relational* models represent systems involving multiple types of interacting entities with probabilistic dependencies among them. Most algorithms for learning the structure of relational models focus on statistical association. The first algorithm we developed that does address causality for relational data—the relational PC (RPC) algorithm [102]—is not complete and is susceptible to orientation errors, as we show in Section 5.5. Consequently, prior to this work, there had been no relational analog to the completeness result for Bayesian networks.

The theory of relational d-separation (see Chapter 4) connects the causal structure of a relational model and probability distributions, similar to how d-separation connects the structure of Bayesian networks and probability distributions. In this

chapter, we present the implications of abstract ground graphs and relational d-separation for learning causal models from relational data.[1]

The main contributions presented in this chapter are:

- A powerful new constraint visible in the abstract ground graph—the *relational bivariate orientation* (RBO) rule—that can identify the direction of causality for bivariate dependencies (yielding models with up to 72% additional oriented dependencies) without assumptions on the underlying distribution (Section 5.2.1)
- Relational extensions to edge orientation rules that have previously only been applied to directed acyclic graphs of propositional data (Section 5.2.2)
- A new algorithm, called *relational causal discovery* (RCD), that leverages the theory of relational d-separation and the abstract ground graph representation to learn causal models from relational data (Section 5.3)
- Proofs of soundness and completeness (enabled by the RBO rule) of the RCD algorithm under causal sufficiency (Sections 5.2.3, 5.2.4, and 5.3.2)

We also show the effectiveness of RCD with an implementation that uses a relational d-separation oracle and compare it to several alternative algorithms (Section 5.5). Finally, we demonstrate the applicability of RCD on real-world datasets drawn from the movie industry (Section 5.6.1) and scholarly citations (Section 5.6.2).

5.1 A Causal Implication of Abstract Ground Graphs

The abstract ground graph representation presents an opportunity to derive new edge orientation rules for algorithms that learn the structure of relational models. There are unique orientations of edges that are consistent with a given pattern of

[1]Portions of this chapter are drawn from Maier et al. [99] with contributions from Katerina Marazopoulou and David Arbour.

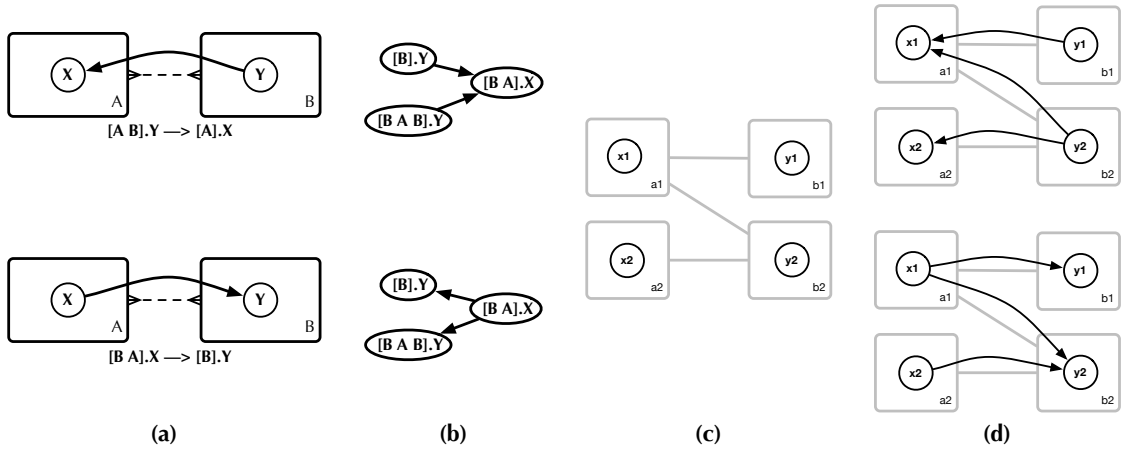


Figure 5.1: (a) Two models of a bivariate relational domain with opposite directions of causality for a single dependency (relationship class omitted for simplicity); (b) a single dependency implies additional dependencies among arbitrary relational variables, shown here in a fragment of the abstract ground graph for B's perspective; (c) an example relational skeleton; and (d) the ground graphs resulting from applying the relational model to the skeleton.

association that can only be recognized in an abstract ground graph. In contrast to bivariate IID data, it is simple to establish the direction of causality for bivariate relational data.

5.1.1 Abstract example

Consider the two bivariate, two-entity relational models depicted in Figure 5.1(a). The first model implies that values of X on A entities are caused by the values of Y on related B entities. The second model implies the opposite, that values of Y on B entities are caused by the values of X on related A entities. For simplicity, the relationship class is depicted as a dashed line between entity classes and relational paths are omitted.

Figure 5.1(b) illustrates a fragment of the abstract ground graph (for hop threshold h=4) that each of the two relational models implies. As expected, the directions of the edges in the two abstract ground graphs are counterposed. Both models produce observable statistical dependencies for relational variable pairs $\langle [B].Y, [B, A].X \rangle$ and

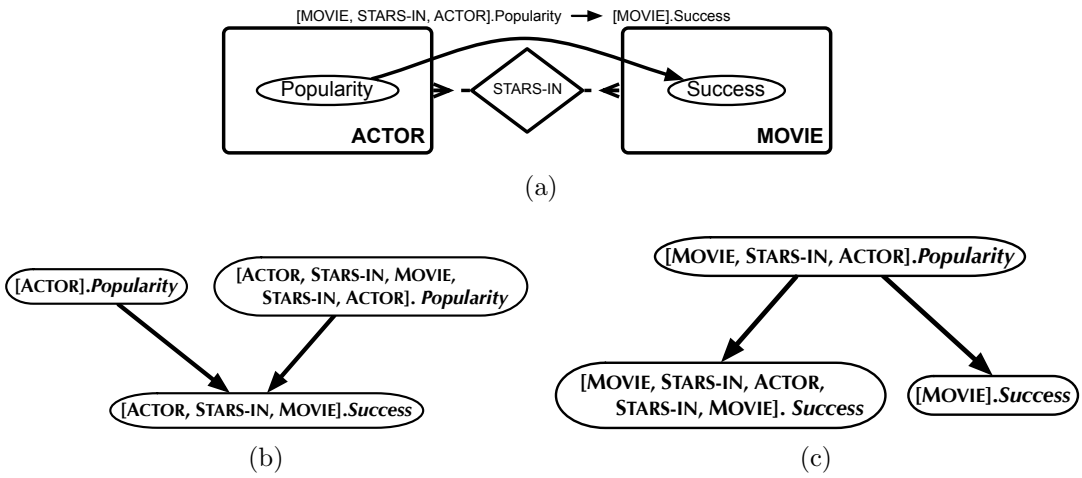


Figure 5.2: (a) An example relational model involving actors and movies with a single relational dependency stating that actor popularity causes movie success. Abstract ground graphs from (b) the ACTOR perspective and (c) the MOVIE perspective.

$\langle [B, A].X, [B, A, B].Y \rangle$. However, the relational variables $[B].Y$ and $[B, A, B].Y$ have different observable statistical dependencies: In the first model, they are marginally independent and conditionally dependent given $[B, A].X$, and in the second model, they are marginally dependent and conditionally independent given $[B, A].X$. As a result, we can uniquely determine the direction of causality of the single dependence by exploiting relational structure. (There is symmetric reasoning for relational variables from A's perspective, and this result is also applicable to ONE-to-MANY data.)

To illustrate this fact more concretely, consider the small relational skeleton shown in Figure 5.1(c) and the ground graphs applied to this skeleton in Figure 5.1(d). In the first ground graph, we have $y_1 \perp\!\!\!\perp y_2$ and $y_1 \not\perp\!\!\!\perp y_2 \,|x_1$, but in the second ground graph, we have $y_1 \not\perp\!\!\!\perp y_2$ and $y_1 \perp\!\!\!\perp y_2 \,|x_1$. These opposing conditional independence relations uniquely determine the correct causal model.

5.1.2 Real example

Consider a data set containing actors with a measurement of their popularity (e.g., price on the Hollywood Stock Exchange[2]) and the movies they star in with a measurement of success (e.g., box office revenue). A simple analysis might detect a statistical association between popularity and success, but the models in which popularity causes success and success causes popularity may be statistically indistinguishable.[3] This is a reasonable decision assuming there are no other variables to investigate, no prior knowledge of the domain, no temporal information, and no asymmetries in the conditional distributions to exploit.

Instead of being restricted to popularity-success variable pairs, a relational representation enables inspection of the varying connections among actor and movie instances. From the perspective of actors, we can ask whether one actor's popularity is conditionally independent of the popularity of other actors appearing in common movies, given the success of those movies. Similarly, from the perspective of movies, we can ask whether the success of a movie is conditionally independent of the success of other movies with common actors, given the popularity of those actors. With conditional independence, we now can determine the orientation for a single *relational* dependency.

These additional tests of conditional independence manifest when inspecting relational data with abstract ground graphs (recall Section 4.4). If actor popularity indeed causes movie success, as shown in the relational model in Figure 5.2(a), then the popularity of actors appearing in the same movie would be marginally independent. This produces a collider from the actor perspective and a common cause from the movie perspective, as shown in Figures 5.2(b) and (c), respectively. Conversely,

---

[2] `www.hsx.com`

[3] In reality, the dependence between actor popularity and movie success is likely to be complex—temporal and involving feedback. However, it is also plausible that the set of movies may consist entirely of independent films starring well-known actors or blockbusters starring newcomers.

if movie success caused actor popularity, then the success of movies would be a common cause of an individual actor's popularity as well as the popularity of other actors appearing in common movies. The abstract ground graphs would exhibit a common cause from the actor perspective and a collider from the movie perspective. With this representation, it is straightforward to identify the orientation of such a bivariate dependency.

These examples illustrate two central ideas. First, abstract ground graphs enable a new constraint on the space of causal models: relational bivariate orientation. The abstract ground graph is the underlying representation used by RCD, and the conditional independence facts derived from it form the crux of the relational bivariate orientation rule. Additionally, the rules used by the PC algorithm can also be adapted to orient the edges of abstract ground graphs (Section 5.2.2). Second, this constraint-based approach—testing for conditional independencies and reasoning about them to orient causal dependencies—is the primary strategy of the RCD algorithm (Section 5.3).

5.2 Edge Orientation

Edge orientation rules, such as those used by the PC algorithm, use patterns of dependence and conditional independence to infer the direction of causality consistent with those patterns [172]. In this section, we introduce the relational bivariate orientation rule (Section 5.2.1) and describe how the PC orientation rules can orient the edges of abstract ground graphs (Section 5.2.2). We also prove that these orientation rules are individually sound and collectively complete for causally sufficient relational data (Sections 5.2.3 and 5.2.4).

5.2.1 Bivariate edge orientation

The examples from Section 5.1 describe the application of relational bivariate orientation (RBO). The abstract ground graph representation presents an opportunity to orient dependencies that cross relationships with a MANY cardinality. RBO requires no assumptions about functional form or conditional densities, unlike the recent work by Shimizu et al. [163], Hoyer et al. [68], Peters et al. [132], and Zhang and Hyvärinen [200] to orient bivariate dependencies. See Section 5.7.2 for a summary of these model classes and their corresponding technique for detecting the direction of causality. The only required assumptions for RBO are that the underlying causal model is acyclic—which restricts the space of dependencies to those without direct or indirect feedback cycles—and causal sufficiency.

In the remainder of this section, let I_W denote the item class on which attribute W is defined, and let $X - Y$ denote an undirected edge.

Definition 5.2.1 (Relational Bivariate Orientation) Let $\mathcal{M}$ be the structure of a relational model, and let G be a partially directed abstract ground graph for $\mathcal{M}$ and perspective I_X. If $[I_X].X - [I_X ... I_Y].Y$ is in G, $card([I_Y ... I_X]) = \text{MANY}$, and $[I_X].X \perp \!\!\!\perp [I_X ... I_Y ... I_X].X \mid \mathbf{Z}$, then (1) if $[I_X ... I_Y].Y \in \mathbf{Z}$, orient as $[I_X].X \leftarrow [I_X ... I_Y].Y$; (2) if $[I_X ... I_Y].Y \notin \mathbf{Z}$, orient as $[I_X].X \rightarrow [I_X ... I_Y].Y$.

RBO is illustrated in Figure 5.3. Given Definition 5.2.1, if $[I_X ... I_Y].Y$ is a collider for perspective I_X, then $[I_Y ... I_X].X$ is a common cause for perspective I_Y, assuming $card([I_Y ... I_X]) = \text{MANY} = card([I_X ... I_Y])$. If $card([I_X ... I_Y]) = \text{ONE}$ and $card([I_Y ... I_X]) = \text{MANY}$, then RBO applies only to the abstract ground graph with perspective I_X. In the example in Figure 5.2(b), [ACTOR, STARS-IN, MOVIE].*Success* is a collider for the ACTOR perspective.

RBO is akin to detecting relational autocorrelation [76] and checking whether a distinct variable is a member of the set that eliminates the autocorrelation. It is also

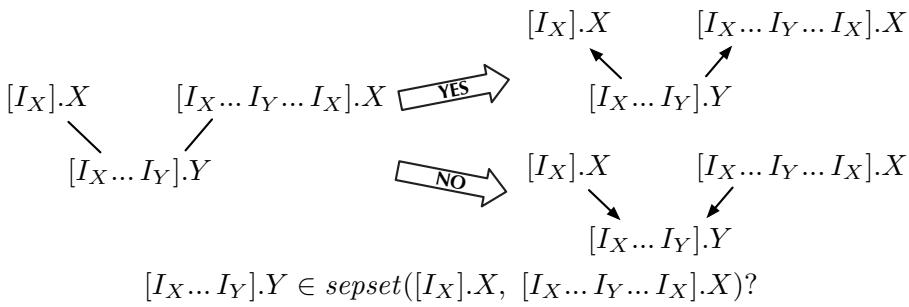


Figure 5.3: The relational bivariate orientation rule is conditional on whether $[I_X ... I_Y].Y$ is in the separating set of $[I_X].X$ and $[I_X ... I_Y ... I_X].X$.

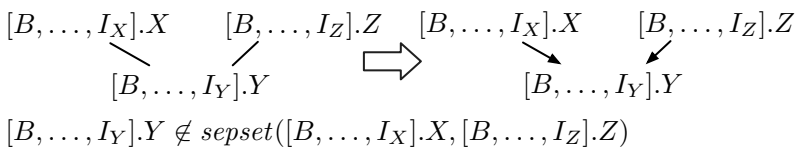
different than the collider detection rule (see Section 5.2.2) because it can explicitly orient dependencies as a common cause when the unshielded triple does not present itself as a collider. In Section 5.5, we quantify the extent to which RBO provides additional information beyond the standard PC edge orientation rules.

5.2.2 Orienting the edges of abstract ground graphs

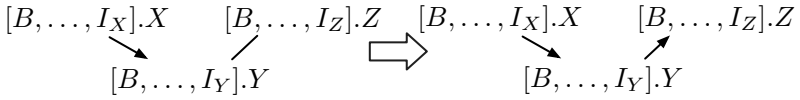
We adapt the rules for orienting edges in a Bayesian network, as used by PC [172] and characterized theoretically by Meek [106] (see Section 2.3.1), to orient relational dependencies at the level of abstract ground graphs. Figure 5.4 displays the four rules[4]—Collider Detection (CD), Known Non-Colliders (KNC), Cycle Avoidance (CA), and Meek Rule 3 (MR3)—as they would appear in an abstract ground graph.

A relational model has a corresponding set of abstract ground graphs, one for each perspective, but all are derived from the same relational dependencies. Recall from Section 4.4 that a single dependency supports many edges within and across the set of abstract ground graphs. Consequently, when a rule is activated for a *specific* abstract ground graph, the orientation of the underlying relational dependency must be propagated within and across *all* abstract ground graphs.

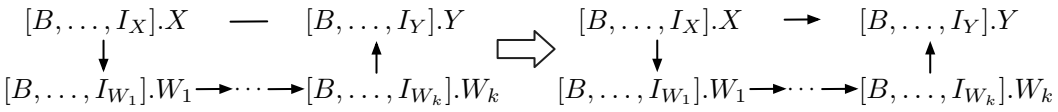
[4]An additional rule is described by Meek [106], but it only activates given prior knowledge.



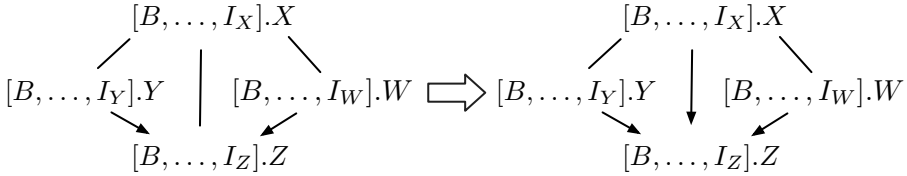
(a) Collider Detection (CD)



(b) Known Non-Colliders (KNC)



(c) Cycle Avoidance (CA)



(d) Meek Rule 3 (MR3)

Figure 5.4: Schematics of the four PC orientation rules as applied to an abstract ground graph from perspective B.

5.2.3 Soundness of orientation rules

An orientation rule is *sound* if any orientation not indicated by the rule introduces either (1) an unshielded collider in some abstract ground graph, (2) a directed cycle in some abstract ground graph, or (3) a cycle in the relational model (adapted from the definition of soundness given by Meek [106]).

Theorem 5.2.1 *Let G be a partially oriented abstract ground graph from perspective B with correct adjacencies and correctly oriented unshielded colliders by either CD or RBO. Then, KNC, CA, MR3, and the purely common cause case of RBO, as well as the embedded orientation propagation, are sound.*

Proof. The proof for KNC, CA, and MR3 is nearly identical to the proof given by Meek [106]. Where unambiguous or unnecessary, we omit specifying relational paths.

KNC: Assume for contradiction that $Y-Z$ is oriented as $Y \leftarrow Z$. Then, $X \rightarrow Y \leftarrow Z$ is an unshielded collider in G.

CA: Assume for contradiction that $X-Y$ is oriented as $X \leftarrow Y$. Then, $X \rightarrow W_1 \rightarrow \cdots \rightarrow Y \rightarrow X$ is a directed cycle in G.

MR3: Assume for contradiction that $X \leftarrow Z$. Then, CA would have oriented $Y \rightarrow X$ and $W \rightarrow X$. But this creates a new unshielded collider: $Y \rightarrow X \leftarrow W$.

Orientation propagation: Let $[B...I_X].X \rightarrow [B...I_Y].Y$ be an oriented edge in G. By the definition of abstract ground graphs, this edge stems from a relational dependency $[I_Y...I_X].X \rightarrow [I_Y].Y$. Let $[B...I_X]'.X - [B...I_Y].Y$ be an unoriented edge in G where $[B...I_X]'$ is different than $[B...I_X]$, but the edge is supported by the same underlying relational dependency. Assume for contradiction that the edge is oriented as $[B...I_X]'.X \leftarrow [B...I_Y].Y$. Then, there must exist a dependency $[I_X...I_Y].Y \rightarrow [I_X].X$ in the model, which yields a cycle. The argument is the same for abstract ground graphs from different perspectives.

RBO common cause case: Given Definition 5.2.1, no alternate perspective would have oriented the triple as a collider, and $B = I_X$. Let $[I_X].X - [I_X...I_Y].Y - [I_X...I_Y...I_X].X$ be an unoriented triple in G. Assume for contradiction that the triple is oriented as $[I_X].X \rightarrow [I_X...I_Y].Y \leftarrow [I_X...I_Y...I_X].X$. This creates a new unshielded collider. Assume for contradiction that the triple is oriented as $[I_X].X \rightarrow [I_X...I_Y].Y \rightarrow [I_X...I_Y...I_X].X$ or equivalently, the reverse direction. This implies a cycle in the model. ■

5.2.4 Completeness of orientation rules

A set of orientation rules is complete if it produces a maximally oriented graph. Any orientation of an unoriented edge must be consistent with a member of the

Markov equivalence class. Lemma 5.2.1 describes a useful property that enables the proof of completeness to reason directly about the remaining unoriented edges.

Lemma 5.2.1 *Let G be a partially oriented abstract ground graph, with correct adjacencies and oriented unshielded colliders. Let G_o be the result of exhaustively applying KNC, CA, MR3, and the purely common cause case of RBO, all with orientation propagation. In G_o, if $P.X \rightarrow P'.Y - P''.Z$, then $P.X \rightarrow P''.Z$.*

Proof. Much of this proof follows from Meek [106].

The following properties hold:

- $X \neq Z$; otherwise, RBO would have oriented $P'.Y \leftarrow P''.Z$.
- $P.X$ must be adjacent to $P''.Z$; otherwise, KNC would have oriented $P'.Y \rightarrow P''.Z$.
- $P.X \leftarrow P''.Z$ *does not* hold; otherwise, CA would have oriented $P'.Y \leftarrow P''.Z$.

Therefore, we have a structure of the form $P.X \rightarrow P'.Y - P''.Z$ and $P.X - P''.Z$.

The orientations of G_o induce a partial ordering on the vertices of G_o: $V_1 < V_2$ if V_1 is an ancestor of V_2. Let $P'.Y$ be a minimal vertex with respect to that partial ordering such that there is no ancestor V in G_o that also has an incoming edge and an adjacency.

We show that $P.X \rightarrow P''.Z$ through exhaustive enumeration of the cases under which $P.X \rightarrow P'.Y$ was oriented. The first four cases follow directly from the proof given by Meek [106].

(1) KNC oriented $P.X \rightarrow P'.Y$. Then, there exists a node $P'''.W$ such that $P'''.W \rightarrow P.X$ and $P'''.W \notin adj(P'.Y)$. $P.X$ has an incoming edge (from $P'''.W$) and an adjacency $P.X - P''.Z$ and is an ancestor of $P'.Y$. This is a contradiction since $P'.Y$ is minimal.

(2) CD or RBO (as a collider) oriented $P.X \rightarrow P'.Y$. Then, there exists a node $P'''.W$ such that $P'''.W \rightarrow P'.Y$ and $P'''.W \notin adj(P.X)$.

- If $P'''.W \notin adj(P''.Z)$, then KNC would have oriented $P'.Y \rightarrow P''.Z$. Therefore, $P'''.W$ and $P''.Z$ must be adjacent.
- If $P'''.W - P''.Z$, then MR3 would have oriented $P''.Z \rightarrow P'.Y$.
- If $P'''.W \leftarrow P''.Z$, then CA would have oriented $P'.Y \leftarrow P''.Z$.
- If $P'''.W \rightarrow P''Z$, then KNC would have oriented $P''.Z \rightarrow P.X$ and CA would have oriented $P''.Z \rightarrow P'.Y$.

(3) MR3 oriented $P.X \rightarrow P'.Y$. This case is similar to case (2).

(4) CA oriented $P.X \rightarrow P'.Y$. Then, there exists a node $P'''.W$ such that $P.X \rightarrow P'''.W \rightarrow P'.Y$. $P'''.W \in adj(P''.Z)$; otherwise, KNC would have oriented $P'.Y \rightarrow P''.Z$. The edge $P'''.W - P''.Z$ must be oriented; otherwise, $P'.Y$ would not be minimal. If $P'''.W \leftarrow P''.Z$, then CA would have oriented $P'.Y \leftarrow P''.Z$. If $P'''.W \rightarrow P''.Z$, then CA would have oriented $P.X \rightarrow P''.Z$.

(5) RBO oriented $P.X \rightarrow P'.Y$ from the I_Y perspective as a common cause. Then, $P' = [I_Y]$, $P = [I_Y ... I_X]$, and $P'' = [I_Y ... I_Z]$. Also, $[I_Y ... I_X ... I_Y].Y$ must be in G_o with $[I_Y ... I_X].X \rightarrow [I_Y ... I_X ... I_Y].Y$. By Definition 5.2.1, $card([I_Y ... I_X]) =$ ONE and $card([I_Y ... I_X]) =$ MANY.

The relational path $[I_Y ... I_Z]$ and its reverse have cardinality ONE; otherwise, RBO would have oriented $[I_Y].Y - [I_Y ... I_Z].Z$. We show that $[I_Y ... I_X].X - [I_Y ... I_Z].Z$ cannot remain unoriented.

This edge exists by the construction of abstract ground graphs: (a) $[I_Y ... I_X] \in extend([I_Y ... I_Z], [I_Z ... I_X])$ and (b) $[I_Y ... I_Z] \in extend([I_Y ... I_X], [I_X ... I_Z])$. The paths $[I_X ... I_Z]$ and $[I_Z ... I_X]$ underlie the dependency between X and Z. Facts (a) and (b) impose constraints on the relational schema and abstract ground graphs.

There are four cases for (a) depending on the relationship between $[I_Y ... I_Z]$ and $[I_Z ... I_X]$, with equivalent cases for (b).

(i) $[I_Y ... I_Z]$ and $[I_Z ... I_X]$ overlap exactly at I_Z. Then, the path from I_X to I_Z must have cardinality MANY. This implies that, from the I_Z perspective, RBO would have oriented X to Z.

(ii) $[I_Y ... I_M ... I_Z]$ and $[I_Z ... I_M ... I_X]$ overlap up to some item class I_M. This is equivalent to case (i), except I_M appears on the path from I_X to I_Z.

(iii) $[I_Z ... I_X]$ is a subpath of the reverse of $[I_Y ... I_Z]$. Then, the path from I_Z to I_Y must have cardinality MANY, which is a contradiction.

(iv) The reverse of $[I_Y ... I_Z]$ is a subpath of $[I_Z ... I_X]$. This is equivalent to case (i), except I_Y appears on the path from I_X to I_Z.

(6) Orientation propagation oriented $P.X \rightarrow P'.Y$. Then, there exists an edge for some perspective that was oriented by one of the orientation rules. From that perspective, the local structure matches the given pattern, and from cases (1)–(5), $X \rightarrow Z$ was oriented. By definition, $P.X \rightarrow P''.Z$. ■

Meek [106] provides two additional definitions and two lemmas (repeated below), which are used for proving completeness.

Definition 5.2.2 (Chordal graph) An undirected graph is *chordal* if and only if every undirected cycle of length four or more has an edge between two nonconsecutive vertices on the cycle.

Definition 5.2.3 (Consistent ordering) Let G be an undirected graph, α a total order on the vertices of G, and G_α the induced directed graph ($A \rightarrow B$ is in G_α if and only if $A < B$ with respect to α). A total order α is a *consistent* ordering with respect to G if and only if G_α has no unshielded colliders.

Lemma 5.2.2 *Only chordal graphs have consistent orderings.*

Lemma 5.2.3 *Let G be an undirected chordal graph. For all pairs of adjacent vertices A and B in G, there exist consistent total orderings α and γ such that $A \rightarrow B$ in G_α and $A \leftarrow B$ in G_γ.*

We now have sufficient background theory to prove the main completeness result for the relational edge orientation rules.

Theorem 5.2.2 *Given a partially oriented abstract ground graph, with correct adjacencies and oriented unshielded colliders, exhaustively applying KNC, CA, MR3, and RBO all with orientation propagation results in a maximally oriented graph G.*

Proof. Much of this proof follows from Meek [106]. Let E_u and E_o be the set of unoriented edges and oriented edges of G, respectively. The following two claims suffice to prove the theorem.

Claim 1 *No orientation of edges in E_u creates a cycle or unshielded collider in G that includes edges from E_o.*

Proof. Assume there exists an orientation of edges in E_u that creates a *cycle* using edges from E_o. Without loss of generality, assume that the cycle is of length three.

(1) If $A \rightarrow B \rightarrow C$ is in E_o and $A - C$ is in E_u, then CA would have oriented $A \rightarrow C$.

(2) If $A \rightarrow B \leftarrow C$ or $A \leftarrow B \rightarrow C$ is in E_o and $A - C$ is in E_u, then no orientation of $A - C$ would create a cycle.

(3) If $A \rightarrow B$ is in E_o and $B - C - A$ is in E_u, then by Lemma 5.2.1 we have $A \rightarrow C$ and no orientation of $B - C$ would create a cycle.

Assume there exists an orientation of edges in E_u that creates an *unshielded collider* using edges from E_o.

(1) Assume $A \rightarrow B$ is in E_o, $B - C$ is in E_u, and A is not adjacent to C. If $B \in sepset(A, C)$, then KNC would have oriented $B \rightarrow C$; otherwise, CD would have oriented $C \rightarrow B$.

(2) Assume $A \rightarrow B$ is in E_o and $B - C$ and $A - C$ are in E_u. By Lemma 5.2.1, $A \rightarrow C$ must be oriented. □

Claim 2 *Let G_u be the subgraph of G containing only unoriented edges. G_u is the union of disjoint chordal graphs.*

Proof. Assume that G_u is not the union of disjoint chordal graphs. Then, there exists at least one disjoint component of G_u that is not a chordal graph. From Lemma 5.2.2, every total ordering of G_u is not consistent. Let $A \rightarrow B \leftarrow C$ be an unshielded collider induced by some ordering on G_u. There are two cases:

(1) A and C are adjacent in G. The edge must be oriented; otherwise, it would appear in G_u. Both orientations of $A - C$ imply an orientation of A and B, or C and B, by Lemma 5.2.1.

(2) A and C are not adjacent in G. Then, $A - B - C$ is an unshielded triple in G. Either CD or RBO would have oriented the triple as a collider, or the triple is inconsistent with the total ordering on G_u. □

Since G is chordal, by Lemma 5.2.3, it follows that no orientation of the unoriented edges in G creates a new unshielded collider or cycle. ■

5.3 The Relational Causal Discovery Algorithm

The relational causal discovery (RCD) algorithm is a sound and complete algorithm for learning causal models from relational data.[5] RCD employs the high-level, constraint-based strategy of the PC algorithm, operating in two distinct phases [172]. RCD is similar to the relational PC (RPC) algorithm, which also learns causal relational models [102]. The differences between RPC and RCD are threefold:

(1) The underlying representation for RCD is a set of abstract ground graphs.

(2) RCD employs a new causal constraint—the relational bivariate orientation rule.

(3) RCD is sound and complete.

RPC also reasons about the uncertainty of relationship existence, but RCD assumes a prior relational skeleton (i.e., that there are no causes of the existence of entities or relationships). The remainder of this section describes the algorithmic details of RCD, and we provide an example trace of RCD's execution in Section 5.3.1 and prove its correctness in Section 5.3.2.

Algorithm 1 provides pseudocode for RCD. Initially, RCD enumerates the set of potential dependencies, in canonical form, with relational paths limited by the hop threshold (line 1). Phase I continues similarly to PC, removing potential dependencies via conditional independence tests with conditioning sets of increasing size drawn from the power set of neighbors of the effect variable (lines 4–11). Every identified separating set is recorded, and the corresponding potential dependency and its reverse are removed (lines 9–10).

---

[5] Code available at `kdl.cs.umass.edu/rcd`.

ALGORITHM 1: RCD($schema, depth, hopThreshold, P$)

```
1  PDs ← getPotentialDeps(schema, hopThreshold)
2  N ← initializeNeighbors(schema, hopThreshold)
3  S ← {}
   // Phase I
4  for d ← 0 to depth do
5      for X → Y ∈ PDs do
6          foreach condSet ∈ powerset(N[Y] \ {X}) do
7              if |condSet| = d then
8                  if X ⫫ Y | condSet in P then
9                      PDs ← PDs \ {X → Y, Y → X}
10                     S[X, Y] ← condSet
11                     break
   // Phase II
12 AGGs ← buildAbstractGroundGraph(PDs)
13 AGGs, S ← ColliderDetection(AGGs, S)
14 AGGs, S ← BivariateOrientation(AGGs, S)
15 while changed do
16     AGGs ← KnownNonColliders(AGGs)
17     AGGs ← CycleAvoidance(AGGs)
18     AGGs ← MeekRule3(AGGs)
19 return getCanonicalDependencies(AGGs)
```

The second phase of RCD determines the orientation of dependencies consistent with the conditional independencies discovered in Phase I. First, Phase II constructs a set of undirected abstract ground graphs, one for each perspective, given the remaining dependencies. RCD then iteratively checks all edge orientation rules, as described in Section 5.2. Phase II of RCD is also different from PC and RPC because it searches for additional separating sets while finding colliders and common causes with CD and RBO. Frequently, unshielded triples $X - Y - Z$ may have no separating set recorded for X and Z. For these pairs, RCD attempts to discover a new separating set, as in Phase I. These triples occur for one of three reasons:

(1) Since X and Z are relational variables, the separating set may have been discovered from an alternative perspective.

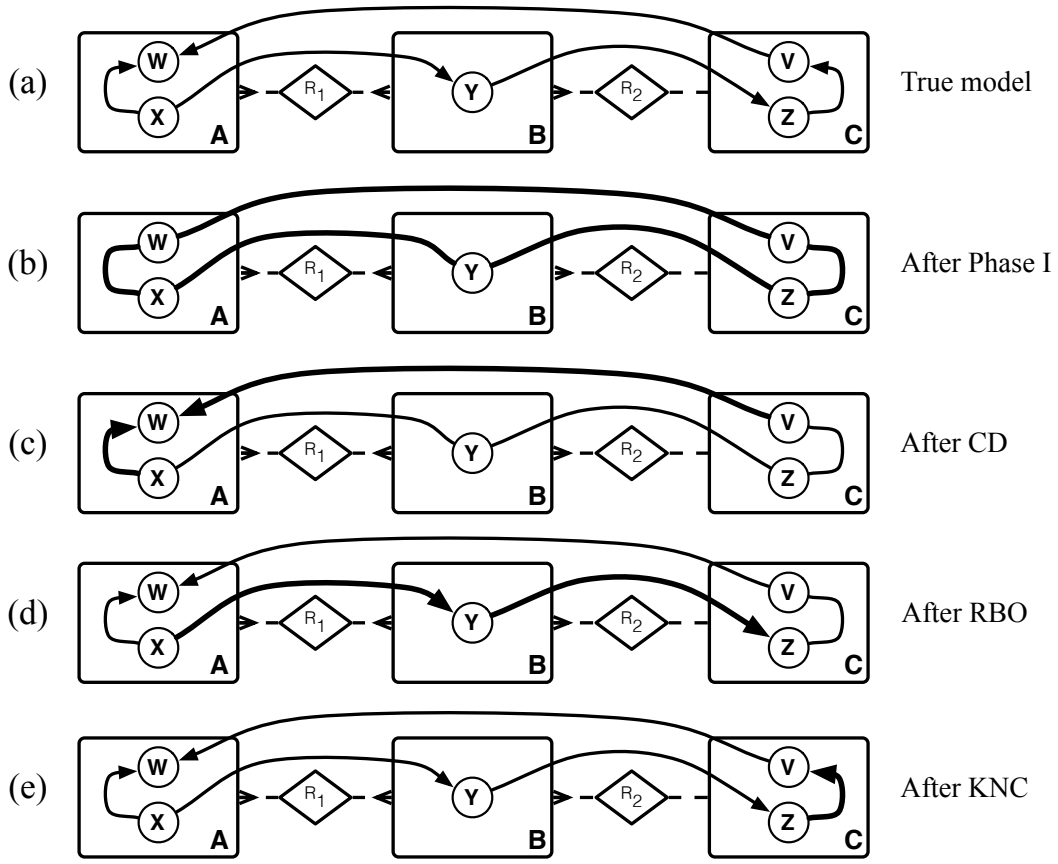


Figure 5.5: (a) Example relational causal model with five dependencies. (b) The output of RCD after Phase I recovers the correct causal skeleton. (c) After collider detection, RCD orients two dependencies. (d) After relational bivariate orientation, RCD orients two more dependencies. (e) The known non-collider rule is activated by virtue of RBO, yielding a fully oriented causal model.

(2) The total number of hops in the relational paths for X, Y, and Z may exceed the hop threshold—each dependency is subject to the hop threshold, but a pair of dependencies is limited by twice the hop threshold.

(3) The attributes of relational variables X and Z are the same, which is necessarily excluded as a potential dependency by the assumption of an acyclic model.

5.3.1 Example trace of RCD

In this section, we trace the operation of RCD on a simple relational domain (a symbolic version of the model in Figure 3.4) and describe how RBO is used to orient

edges that would otherwise remain unoriented, as well how RBO enables the activation of additional orientation rules. Consider the relational model in Figure 5.5(a). There are three entity classes (A, B, and C) connected by a MANY-to-MANY relationship class R_1 and a MANY-to-ONE relationship class R_2. The model includes five attributes for which there is a chain of four dependencies: X on A causes Y on B, which causes Z on C, which causes V also on C, which finally causes W on A. There is a fifth dependency for which X on A causes W on A. All relational paths underlying these dependencies are as expected, following the simplest path between entity classes.

After the first phase of RCD, the algorithm recovers the correct causal skeleton, as depicted in Figure 5.5(b). RCD also records that $[C, R_2, B, R_1, A].X$ is independent of $[C].V$ given $[C].Z$. Thus, RCD can immediately orient $X \rightarrow W \leftarrow Z$ as a collider, as shown in Figure 5.5(c). Without access to the RBO rule, RCD would not be able to further orient the model.

Since RCD can orient bivariate dependencies that cross MANY cardinalities, RBO correctly orients the $X \rightarrow Y$ and $Y \rightarrow Z$ dependencies, as shown in Figure 5.5(d). RCD orients the former dependency because $[B, R_1, A, R_1, B].Y$ is independent of $[B].Y$ given $[B, R_1, A].X$, which it discovers in Phase II. RCD orients the latter dependency because $[B, R_2, C].Z$ does not appear in the separating set for $[B].Y$ and $[B, R_1, A, R_1, B].Y$, as just discovered.

Finally, a new unshielded triple—$\langle Y, Z, V \rangle$—that is known to *not* be a collider (because it has not already been oriented as such) with $Y \rightarrow Z$ oriented enables the KNC rule to activate. The $Z \rightarrow V$ edge is oriented correctly, made possible by the additional orientations from RBO, shown in Figure 5.5(e). The final model learned by RCD is completely oriented.

5.3.2 Soundness and completeness of RCD

Given the algorithm description and the soundness and completeness of the edge orientation rules, we prove that RCD is sound and complete. The proof assumes causal sufficiency and a prior relational skeleton (i.e., no causes of the relational structure).

Theorem 5.3.1 *Given a schema and probability distribution P, RCD learns a correct maximally oriented model $\mathcal{M}$ assuming perfect conditional independence tests, sufficient hop threshold h, and sufficient depth.*

Proof sketch. Given sufficient h, the set of potential dependencies *PDs* includes all true dependencies in $\mathcal{M}$, and the set of neighbors N includes the true causes for every effect relational variable. Assuming perfect conditional independence tests, *PDs* includes exactly the undirected true dependencies after Phase I, and $S[X, Y]$ records a correct separating set for the relational variable pair $\langle X, Y \rangle$. However, there may exist non-adjacent pairs of variables that have no recorded separating set (for the three reasons mentioned above). Given the remaining dependencies in *PDs*, we construct the correct set of edges in *AGGs* using the methods described in Chapter 4. Next, all unshielded colliders are oriented by either CD or RBO, with correctness following from Spirtes et al. [172] and relational d-separation. Whenever a pair $\langle X, Y \rangle$ is missing a separating set in S, it is either found as in Phase I or from a different perspective. RCD then produces a maximally oriented model by the soundness (Theorem 5.2.1) and completeness (Theorem 5.2.2) results of the remaining orientation rules. ■

5.4 Evaluating the Results of Causal Discovery Algorithms

Algorithms that learn joint models of statistical associations attempt to maximize the probability that a model fits an input data set. In contrast, most causal discovery algorithms attempt to learn the causal *structure* of a set of variables. Since the goal

of causal discovery focuses on the learned structure as opposed to the model parameters, these algorithms require a fundamentally different set of evaluation techniques than have been commonly employed in statistics and machine learning (e.g., held-out likelihood, cross-validated accuracy). Evaluation is more challenging in this setting because it may require knowledge of the true generative model, and it also requires new measures of structural accuracy.

5.4.1 Evaluation approaches

Empirical evaluation begins with the choice of a data set, and for causal discovery, there are at least four general approaches. One approach is to synthetically generate causal models from some distribution over the space of model structures. For Bayesian networks, it is possible to sample uniformly at random from the space of all directed acyclic graphs [107, 73] or given structural constraints, such as induced width [74]. For relational models, no similar result has been achieved, but in Section 5.5, we employ a simple greedy approach to generate model structures. Generating synthetic causal structures is useful because it provides a controlled, systematic method to internally validate causal discovery algorithms. However, this approach assumes that the generated models are somehow representative of causal structures found "in the wild."

Another option is to generate data from models that have been encoded by domain experts and used in practice or to use real data for which the generative structure is known. Widely used models for evaluating Bayesian network learning include the ALARM network for monitoring patient vitals [9], the Asia network for describing lung disease and visits to Asia [90], and the Hailfinder weather forecast system [2]. Similarly, the growing literature on methods for bivariate causal discovery uses a variety of real-world domains with a single known causal dependency, such as the Old Faithful geyser data set [7]. Evaluating on realistic causal structures is important,

but reporting results on only several examples does not offer a rigorous investigation of an algorithm's efficacy.

In a similar vein, several researchers have published results that compare the dependencies learned from observational data with those that have been experimentally validated. This approach has been mostly applied to biological networks for which experiments are relatively cheap (e.g., yeast gene expression [71] and protein interactions [152]). This is particularly attractive because it simultaneously presents a valuable use case and convincing evaluation. Furthermore, this approach does not necessarily require knowledge of the complete ground truth model: If the domain presents the opportunity to intervene, it may be possible to compare the predictive accuracy of the learned model against the true post-interventional distribution. Unfortunately, this option is rarely used because it may require extensive collaboration with practitioners and is limited to domains that are easily manipulated. It is also not currently viable in the relational setting because developing the precise semantics of interventions remains an open research question.

A fourth alternative is to apply causal discovery algorithms to real-world data sets with no ground truth. One concern for the relational setting is that the size of data sets (i.e., the number of variables and sample size) will lead to intractable runtimes. The evaluation for this approach centers around a practical demonstration—that causal discovery algorithms can be implemented in practice and to scale. This does not provide a comprehensive investigation of causal accuracy, but it can lead to anecdotal evidence of learned dependencies. We apply this approach to two real domains in Section 5.6.

5.4.2 Evaluation measures

The output of causal discovery algorithms is typically a (partially) directed acyclic graph that represents the learned causal structure. Various measures have been de-

veloped to compare the learned and true model structures. The majority of these measures focus on characterizing structural errors or structural accuracy (see the survey conducted by de Jongh and Drużdżel [35]).

The most basic measures count the number of mistakes made by the algorithm. In their evaluation of modifications to the PC algorithm, Abellán et al. compute the number of missing links and the number of added links in the learned model as they vary sample size [1]. Colombo et al. also use these measures (among others), but present their values as the significance level threshold α is varied to analyze how sensitive the learned skeleton is to Type I errors [26]. These measures reflect the magnitude of the errors, but they do not characterize worst-case performance, and they restrict the focus to errors made during skeleton identification.

The measures originally used to evaluate the PC algorithm deconstruct errors into four classes: edge commission and omission and orientation commission and omission [172]. An edge (orientation) commission occurs when the algorithm introduces an edge (orientation) that does not exist in the true model. Conversely, an edge (orientation) omission occurs when the algorithm fails to include an edge (orientation) that does exist in the true model. These measures are also presented as relative percentages, normalized by the total number of possible errors of each kind.

In Section 5.5, we use measures similar to commission and omission, influenced by the evaluation of information retrieval systems: precision and recall. Instead of relevant and retrieved documents, we consider edges and orientations. We define skeleton precision, skeleton recall, oriented precision, and oriented recall as follows:

$$\text{skeleton precision} = \frac{|\{\text{learned edges}\} \cap \{\text{true edges}\}|}{|\{\text{learned edges}\}|}$$

$$\text{skeleton recall} = \frac{|\{\text{learned edges}\} \cap \{\text{true edges}\}|}{|\{\text{true edges}\}|}$$

$$\text{oriented precision} = \frac{|\{\text{learned orientations}\} \cap \{\text{true orientations}\}|}{|\{\text{learned orientations}\}|}$$

$$\text{oriented recall} = \frac{|\{\text{learned orientations}\} \cap \{\text{true orientations}\}|}{|\{\text{true orientations}\}|}$$

In contrast with normalized commission, the denominator for precision is the number of learned edges or orientations as opposed to the total number of edges that could have been learned. This yields a measure of accuracy that is not biased toward zero.

There are also measures that aggregate low-level errors into a summary statistic. The F_1 score is the harmonic mean of precision and recall. Another commonly used measure extends the classic Hamming distance from information theory to graph structures, and it is appropriately named the structural Hamming distance (SHD). SHD is the minimum number of edge insertions, deletions, and flips necessary to transform the learned graph into the true Markov equivalence pattern (as opposed to the true, fully directed model). SHD was defined by Tsamardinos et al. in their extensive empirical evaluation of MMHC [188], similarly defined by Acid and de Campos [4], and modified to assign smaller penalties to orientation errors by Perrier et al. [130]. Colombo et al. also present the mean and variance of SHD across various trials and as a function of α [26]. Aggregate measures are useful for summarizing performance, but they do not provide a means to examine specific errors in the way that precision and recall do.

More recently, novel approaches have been developed that depart from the traditional focus on structural accuracy to the causal implications of learned structure. If no errors were introduced by the algorithm, then the causal implications (defined by interventional distributions) of the learned model would be consistent with the true model. However, some structural errors (e.g., missing edges, incorrect orientations) may have no effect on the causal implications, and some errors are more deleterious

than others. To that end, Peters and Bühlmann devised an analog to structural Hamming distance called structural intervention distance (SID) [131]. SID is equal to the number of interventional distributions across all pairs of variables that differ between the learned and true model. While this begins to evaluate the actual underlying goal of causal discovery, it requires identifiability results of Pearl's *do*-calculus and interventional distributions, specifically the characterization of adjustment sets shown by Shpitser et al. [165]. There is currently no relational analog to these concepts or SID.

Another recent measure compares the estimated causal effects of the learned model against the results of actual experiments. This approach, called intervention-calculus when the DAG is absent (IDA), computes a lower bound on each causal effect [96]. Given these bounds, all pairs of variables can be ranked by the size of their causal effect and compared to the true ranking. Currently, there has been just a single domain—single interventions on yeast gene expression data [71]—for which this method has been applied [95, 26]. While their applicability is limited to domains that can be manipulated, both the IDA and SID evaluation approaches are on the general path toward providing convincing evidence to the larger community that causal discovery algorithms are practical and broadly applicable.

5.5 Synthetic Experiments

The proofs of soundness and completeness offer a qualitative measure of the effectiveness of RCD—no other method can learn a more accurate causal model from observational data. To complement the theoretical results, we provide a *quantitative* measure of RCD's performance and compare that against the performance of alternative constraint-based algorithms.

We evaluate RCD against two alternative algorithms. The first algorithm is RPC [102]. This provides a comparison against state-of-the-art causal structure learning for relational data. The second algorithm is the PC algorithm executed on relational data

that has been propositionalized from a specific perspective—termed Propositionalized PC (PPC). Propositionalization reduces relational data to a single, propositional table [87]. (See Section 3.8.4 for more details.) For each perspective, we include all relational dependencies that contain no repeated item classes in the relational paths. We then take the best and worst perspectives for each trial by computing the average F_1 score of its skeleton and oriented models.

We generated 1,000 random causal models over randomly generated schemas for each of the following combinations: entities (1–4); relationships (one less than the number of entities) with cardinalities selected uniformly at random; attributes per item drawn from $Pois(\lambda=1)+1$; and relational dependencies (1–15) limited by a hop threshold of 4 and at most 3 parents per variable. This procedure yielded a total of 60,000 synthetic models. Note that this generates simple Bayesian networks when there is a single entity class. We ran RCD, RPC, and PPC for each perspective, using a relational d-separation oracle with hop threshold 8 for the abstract ground graphs.

We compared the learned causal models with the true causal model. For each trial, we recorded the precision (the proportion of learned edges in the true model) and recall (the proportion of true edges in the learned model) for both the undirected skeleton after Phase I and the partially oriented model after Phase II. (See Section 5.4 for a discussion on evaluating causal discovery algorithms.) Figure 5.6 displays the average across 1,000 trials for each algorithm and measure. We omit error bars as the maximum standard error was less than 0.015.

All algorithms learn identical models for the single-entity case because they reduce to PC when analyzing propositional data. For truly relational data, algorithms that reason over relational representations are necessary for accurate learning. RCD and RPC recover the exact skeleton, whereas the best and worst PPC cases learn flawed skeletons (and also flawed oriented models), with high false positive and high false negative rates. This is evidence that propositionalizing relational data may lead to

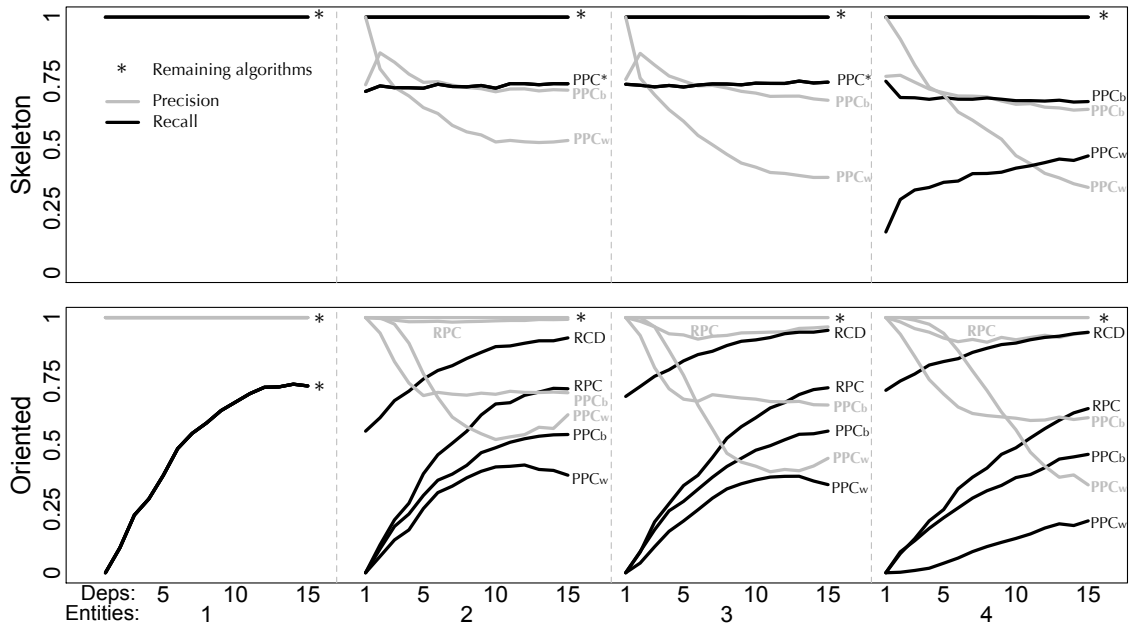


Figure 5.6: Skeleton and oriented precision and recall for the RCD and RPC algorithms, as well as the best and worst perspective for PPC as a baseline. Results are averaged over 1,000 models for each setting.

inaccurately learned causal models. (See Section 3.8.4 for an extended discussion about propositionalization.)

For oriented models, the RCD algorithm vastly exceeds the performance of all other algorithms. As the soundness result suggests, RCD achieves an oriented precision of 1.0, whereas RPC introduces orientation errors due to reasoning over the class dependency graph and missing additional separating sets. For recall, which is closely tied to the completeness result, RCD ranges from roughly 0.56 (for 1 dependency and 2 entities) to 0.94 (for 15 dependencies and 4 entities). While RPC and PPC cannot orient models with a single dependency, the relational bivariate orientation rule allows RCD to orient models using little information. RCD also discovers more of the underlying causal structure as the complexity of the domain increases with respect to both relational structure (more entity and relationship classes) and model density.

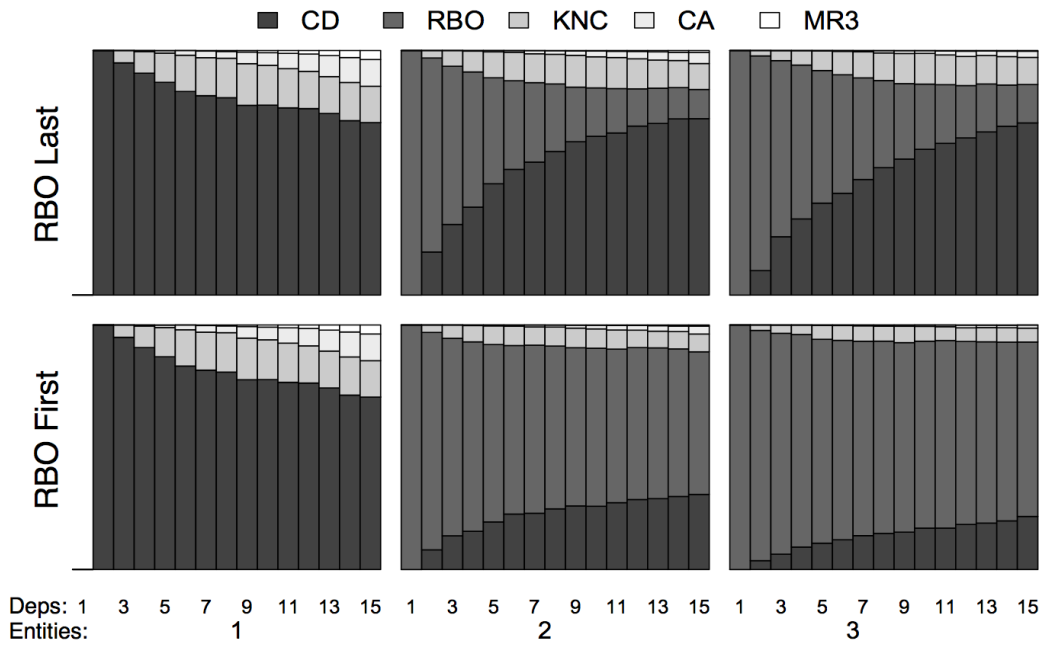


Figure 5.7: Frequency of edge orientation rules in RCD, with RBO last (above) and first (below).

To quantify the unique contribution that RBO provides, we applied RBO as the *final* orientation rule in Phase II and recorded the frequency with which each edge orientation rule is activated (see Figure 5.7). As expected, RBO never activates for the single-entity case because all paths have cardinality ONE. For truly relational domains, RBO orients between 11% and 100% of the oriented edges. However, this does not fully capture the broad applicability of RBO. Therefore, we also recorded the frequency of each edge orientation rule when RBO is applied *first* in Phase II of RCD. In this case, for at least two entity classes, RBO orients between 58% and 100% of the oriented edges.

Finally, we recorded the number of conditional independence tests used by the RCD and RPC algorithms. RCD learns a more accurate model than RPC, but at the cost of running additional tests of independence during Phase II. Fortunately, these extra tests do not alter the asymptotic complexity of the algorithm, requiring on average 31% more tests.

5.6 Practical Demonstration

In Sections 5.6.1 and 5.6.2 below, we present models that RCD learned from relational data drawn from the movie industry and scholarly publishing. To do so, we implemented a practical version of RCD that replaces the relational d-separation oracle with conditional independence tests from a finite sample.

Accurate tests of conditional independence for relational data remains an open research area. In Section 4.8.3, the experiment validating relational d-separation conveyed just one type of relational bias present in standard statistical tests. The work by Rattigan outlines several known statistical biases of relational data and describes potential hypothesis tests that address those problems [139]. As an initial demonstration, we impose a restrictive assumption—that all variables are drawn from normal distributions and linear models—and use linear regression and t-tests of coefficients to determine conditional independence. We also default to the average aggregation function for relational variables. More realistic implementations of RCD can replace linear regression with conditional independence tests that rely on fewer assumptions and search across different aggregators.

We also introduce two thresholds for judging conditional independence. First, RCD uses the standard α threshold for determining whether a p-value is significant. This controls for Type I errors, with a default setting of $\alpha = 0.01$. Unlike other implementations of constraint-based methods, we also set an effect size threshold that governs whether a significant dependence is also *substantive*. Because sample sizes are typically large enough to attain high statistical power and produce statistically significant test values, it can be important to filter out dependencies with weak effects. This is similar to the Bayesian approach used in the BCCD algorithm that ranks independence statements by their likelihood [25], or more generally, score-based approaches that penalize additional structure. Varying the effect size threshold (and

re-running RCD) can facilitate comparison of learned models with different structural densities.

5.6.1 Movie industry

We applied RCD to the MovieLens+ database, a combination of the UMN MovieLens database (`www.grouplens.org`); box office, director, and actor information collected from IMDb (`www.imdb.com`); and average critic ratings from Rotten Tomatoes (`www.rottentomatoes.com`). Of the 1,733 movies with this additional information, we sampled 10% of the user ratings yielding roughly 75,000 ratings. For testing conditional independence, RCD checks the significance of coefficients in linear regression and uses the average aggregation function for relational variables. The RCD-generated model is displayed in Figure 5.8.

We ran RCD with a hop threshold of 4, maximum depth of 3, and an effect size threshold of 0.01. Because constraint-based methods are known to be order-dependent [26], we ran RCD 100 times and used a two-thirds majority vote to determine edge presence and orientation. Out of approximately 690 potential dependencies, RCD discovered 27 having no separating set. One interesting dependency is that the average number of films that actors have starred in affects the number of films the director has directed—perhaps well-established actors tend to work with experienced directors. Also, note that genre is a composition of binary genre attributes.

5.6.2 Scholarly citations

We applied RCD to PubMed (`http://www.ncbi.nlm.nih.gov/pubmed`), a data set consisting of biomedical articles, their authors, and their citations to and from other articles in PubMed. The data also contain the publishing venue and author institution of each paper. We computed additional relational features concerning author impact, as measured by h-index [64], venue impact [46], and paper topics (e.g., similarity to other papers in the same venue, entropy of the topic distribution).

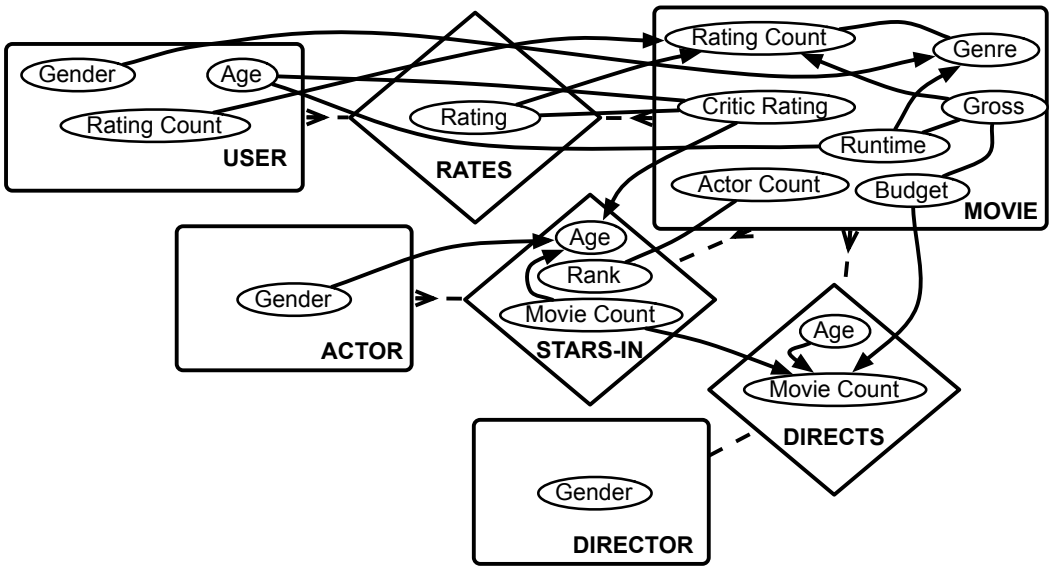


Figure 5.8: RCD-learned model of MovieLens+.

We randomly sampled 10,000 papers, retaining their 53,000 authors and 333 distinct venues. These authors are associated with roughly 2,300 institutions. Because citations are only within PubMed and the sample of papers, we only retained 530 citations. Similar to the analysis of MovieLens+, RCD used linear regression, the average aggregation function, a hop threshold of 4, maximum depth of 3, and effect size threshold of 0.01. The RCD-generated model is displayed in Figure 5.9.

Out of approximately 1,200 potential dependencies, RCD discovered 14 dependencies for which no separating set could be found. One interesting dependency is that the impact factor of a venue influences the maximum h-index of the authors of a paper. This could be due to prominent journals publishing papers by prolific authors. Several dependencies with high effect sizes involve the temporal sequence of citations (i.e., the number of citations after year one affects a paper's second year citations, etc.).

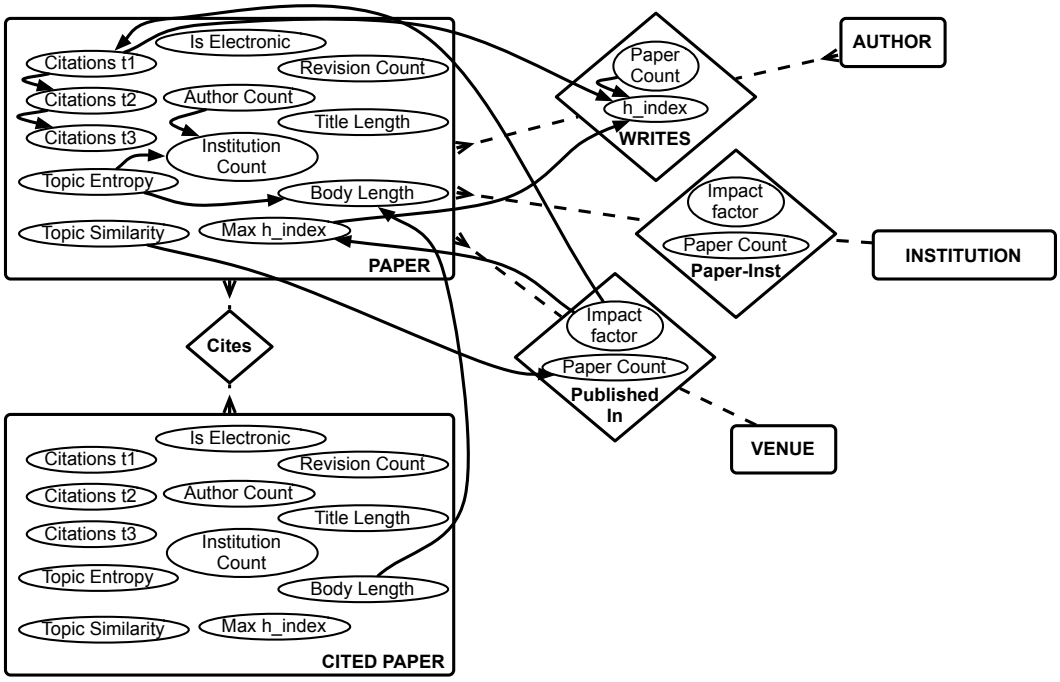


Figure 5.9: RCD-learned model of PubMed.

5.7 Related Work

Aside from propositionalizing relational data and using existing techniques as described in Section 3.8.4 (which may succumb to fundamental problems for causal discovery), there are at least two distinct related research areas with similar aspects to RCD. First, many relational representations and corresponding structure learning algorithms have been proposed in the statistical relational learning literature. Second, a class of methods for orienting bivariate dependencies from propositional data has been gaining attention over the past eight years.

5.7.1 Statistical relational learning

To the best of our knowledge, the only structure learning algorithms for relational data that explicitly reason about causality are RCD and its predecessor, relational PC (RPC). RPC was the first to attempt to learn causal models [102], but we have shown that it is not complete, mainly due to the absence of the RBO rule. We have

also shown that RPC is not sound in its second phase, as it may introduce orientation errors, even in the large sample limit, due to its reliance on class dependency graphs as opposed to abstract ground graphs. Although RPC was developed to additionally reason about relationship existence uncertainty—a capability beyond RCD—it was not grounded theoretically.

All other relational structure learning algorithms focus on modeling statistical associations and optimizing the fit of the joint distribution. The class of relational models over which RCD learns is similar to probabilistic relational models (PRMs). All PRM learning algorithms follow the search-and-score paradigm, with unknown causal implications [43]. For learning Bayesian networks, in the sample limit with a known node ordering, it can be shown that constraint-based methods, using independence tests of conditional cross entropy, learn identical models to search-and-score methods, using Kullback-Leibler divergence [29]. However, even relaxing the node ordering assumption produces results for which "...conditional independence searching *can* be more refined than using scoring metrics" (quotation from Cowell's conclusions [29]). Evaluating alternative paradigms for causal relational learning is open for future research.

There are also various algorithms for learning the structure of undirected models of relational data that avoid the acyclicity requirement of directed acyclic models but cannot represent causality. The relational Markov network was the first such model with an accompanying structure learning algorithm [177], but it was soon outclassed by Markov logic networks (MLNs). MLNs were introduced to express more complex distributions, and many researchers have focused on developing methods for learning and inference in these models. Of particular note are the first dedicated structure learning algorithm (aside from the simplistic inductive logic programming methods used orignally) developed by Kok and Domingos [82], the bottom-up approach taken

by Mihalkova and Mooney [108], and the improved approach of Kok and Domingos that finds structural motifs [83].

5.7.2 Methods for orienting bivariate dependencies

The relational bivariate orientation rule can detect the direction of causality between a pair of relational variables with *no* assumptions on the underlying functional form. For propositional data, there is no analogous result if we do not impose additional distributional assumptions. However, a growing set of results on *additive noise models* enables the identification of bivariate orientations. For the pairs of variables that cannot be oriented with RBO, these approaches could conceivably be integrated with the RCD algorithm.

The additive noise model is a large subset of all functional parameterizations of Bayesian networks. Under these models, it can be shown that the direction of causality for a statistical dependence between two random variables X and Y can be uniquely identified if the error term of one variable, say η_X, is independent of the other variable, say Y, but the reverse does not hold. Specific forms of additive noise models were introduced by Shimizu et al. for linear models with non-Gaussian noise [163], Hoyer et al. for nonlinear additive noise models [68], Tillman et al. for weakly additive noise models [185], Zhang and Hyvärinen for post-nonlinear models [200], and Peters et al. for discrete additive noise models [132]. Recently, the first consistency results were proven under the additive noise model, but there are no known finite sample convergence rates [86].

To learn models with more than two variables under this setting, Peters et al. extend the notion of additive noise models to *identifiable functional model classes* (IFMOCs) [133]. This work extends all the bivariate additive noise models mentioned above to the multivariate, conditional setting, and they develop a causal discovery algorithm that can completely identify causal structure if the true generating model is

an IFMOC. Although the consistency of this approach remains unproven, this general research direction could present an opportunity for learning causal relational models in combination with RCD.

5.8 Concluding Remarks

Relational d-separation and the abstract ground graph representation from Chapter 4 provide a new opportunity to develop theoretically correct algorithms for learning causal structure from relational data. In this chapter, we presented the relational causal discovery (RCD) algorithm and proved it sound and complete for discovering causal models from causally sufficient relational data (Section 5.3). The completeness result is enabled by the causal implications of abstract ground graphs. Specifically, relational bivariate orientation (RBO), which can detect the orientation of bivariate dependencies, only manifests when examining the implications of relational models with abstract ground graphs (Section 5.2.1). Powered by RBO, the RCD algorithm achieves recall of oriented relational models over a previous state-of-the-art algorithm that is 18% to 72% greater on average.

The primary objective of this thesis is to extend causal discovery to expressive representations that more closely model real-world domains. While this goal has certain theoretical and practical challenges, it is necessary for effective causal discovery. The experiment in Section 5.5 provides evidence that native representations and learning algorithms for relational data retain valuable information for causal discovery. Furthermore, rich representations lead to new approaches for discovering additional causal constraints—the RBO rule is just a single example. In prior work, we have shown that relational blocking is another useful method for causal discovery [141], and it is likely that representations that are more expressive than relational will present more such opportunities.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

The increasing connectedness across different levels and aspects of society and the ability for computer systems to record individual interactions has led to an availability of massive, complex data sets for seemingly any domain. The knowledge and insights buried within these data sets can transform public interests and commercial prospects. Such is the allure and promise of the so-called "Big Data" movement: The technology to compile, process, and analyze massive, complex data sets will revolutionize industry and society by yielding more informed decision-making.

In many respects, these goals can be achieved by detecting patterns of association. Applications that rely on improved sensing, such as computer vision, voice-recognition, or certain aspects of robotics, are driven by large sets of observational data and complex models of association. However, to produce substantial change in real domains that are studied by researchers in many disciplines (e.g., social science, medicine, defense science, behavioral science, economics), merely identifying association cannot explain observations or produce meaningful, actionable knowledge to support decision-making. This requires *causal* models, and the tools and algorithms to learn and reason about these models need to be developed and improved. In the following sections, I review the contributions made in this thesis toward that goal and offer several directions for future research that continue this progression.

6.1 Summary of Contributions

Traditional causal discovery in propositional representations has provided a solid theoretical and, to some extent, practical foundation. However, propositional representations can only model the attributes of a single type of entity, which is too restrictive for many relational, real-world systems that involve the interactions among multiple types of entities. The work presented in this thesis extends the representation, reasoning, and learning for causal discovery in domains that are inherently relational.

Chapter 3 described a formal representation for relational models, focusing on the precise semantics for instantiating relational paths and variables with a specific relational skeleton. These concepts form the basis for representing causal dependencies, and they enable the theory for deriving conditional independence facts.

Chapter 4 presented the theory of relational d-separation upon showing that traditional d-separation for Bayesian networks fails to correctly derive all conditional independencies for a relational model. I introduced the abstract ground graph representation, which lies between the model and ground levels, in order to capture all paths of dependence among relational variables. I showed that abstract ground graphs enable the sound and complete derivation of the conditional independencies encoded in a relational model.

Chapter 5 developed a sound and complete constraint-based structure learning algorithm, called relational causal discovery (RCD). This algorithm leverages the connection between conditional independence and relational causal structure provided by relational d-separation, and takes advantage of the implications of the abstract ground graph representation. RCD employs a new orientation rule—relational bivariate orientation (RBO)—that can identify the causal direction of dependencies between pairs of relational variables, with no parametric assumptions, and enables the completeness proofs of RCD.

6.2 Future Work

While this thesis deals primarily with the theoretical foundation for relational causal discovery, it provides capabilities that should be further developed for real-world applications. I describe several possible directions that can extend this work to more effective and practical causal discovery.

- **Devise accurate methods for testing conditional independence.** The effectiveness of constraint-based algorithms, such as RCD, hinge on the accuracy of its tests of conditional independence. For example, Tsamardinos and Borboudakis show that permutation tests can improve the accuracy of learned Bayesian networks because sample-specific reference distributions are more accurate than the asymptotic limiting distributions of conventional tests [187]. If the statistical tests require parametric assumptions that are not met, then the recovered structure will likely include false positive and negative edges. For demonstrative purposes, we implemented a simple test of independence (using linear regression) in RCD, but more robust tests are necessary for real-world applications.

  There is a large body of research on developing hypothesis tests, but most approaches focus on two-sample, bivariate tests of dependence, and there is almost no research on tests for relational data. A few recent approaches for testing propositional data, such as the Bayesian multiresolution test for continuous variables [105], kernel tests [59, 201], and conditional correlation independence [138], appear promising for relational extensions. The structured kernel test for non-IID data [202] and the tests devised by Rattigan [139] should also be investigated as potential candidates.

- **Develop alternative constraint-based strategies and causal discovery paradigms.** Many different constraint-based strategies have been developed to

fix certain classes of errors that can be introduced by PC under finite samples (see Section 2.3.2). These strategies could conceivably be extended to relational structure learning. Additionally, the search-and-score and hybrid paradigms have not been explored for learning causal structure of relational models. While learning PRMs has, thus far, been limited to search-and-score, those algorithms currently choose a single model. In contrast, a relational extension to greedy equivalence search (GES) [23] would learn a Markov equivalence class as RCD does. A broad set of algorithms that learn relational causal structure would enable a large-scale empirical analysis that could be conducted to compare performance under various finite-sample scenarios.

- **Incorporate blocking to relax causal sufficiency.** One assumption made by RCD to provide soundness and completeness is that the data are causally sufficient, involving no latent common causes. In recent work, we showed that *relational blocking*—a generalization of blocking designs, twin studies, and multilevel models—is easily expressed within a relational representation [141, 139]. Blocking is a fundamentally new operator that exploits relational structure to provide constraints on the space of causal relational models in the presence of latent confounders. Blocking is different from tests of conditional independence because it (1) reduces variability and increases statistical power, (2) relaxes causal sufficiency by controlling for observed and latent variables, and (3) does not induce dependence when conditioning on common effects. In work that is currently under submission, we formalize blocking using the relational d-separation theory, integrate it with conditional independence tests in RCD, and show that it can relax causal sufficiency while increasing efficiency (i.e., reducing the number of tests necessary to recover causal structure).

- **Incorporate Bayesian priors and evidence.** The output of RCD and other constraint-based algorithms is a single causal structure, and its execution is entirely data-driven. Recent work by Borboudakis and Tsamardinos has shown that expert prior knowledge of causal dependencies and paths can be incorporated into structure learning algorithms [12, 13]. Additionally, Claassen and Heskes developed a constraint-based algorithm that uses a posterior score to rank dependencies by some Bayesian confidence [25]. These approaches can guide structure learning with expert judgments and control the output by its sensitivity to effect sizes, which should be beneficial in practical settings.

- **Increase the expressiveness of the underlying representation.** The overarching theme of this thesis is to extend traditional causal discovery to the more expressive relational representation. However, there are additional data complexities beyond relational structure that would be useful to model for causal discovery. The representational limitations described in Section 3.9 all present useful research directions for extending representation, reasoning, and learning. Specifically, extending our model class to explicitly represent latent variables, existence of entity and relationship instances, ontologies over entity and relationship classes, temporal dynamics, and feedback is important future work, especially in the context of learning causal models of realistic domains.

6.3 Broader Impact

Beyond future technical directions, the contributions of this thesis provide a foundation that has the potential to produce innovative theory, systems, and applications that could have an impact on a wide range of important domains. Below, I present several implications of this research that have a potential broad impact in other fields and real domains.

- **Correct analyses of previous real-world studies.** The results in Chapter 4 point to potential flaws in the design and analysis of some real-world studies. If researchers of social or economic systems choose inappropriate data and model representations, then their analyses may omit important classes of dependencies. Specifically, the relational d-separation theory implies that choosing a propositional representation from an inherently relational domain may lead to serious errors. An abstract ground graph from a given perspective defines the exact set of variables that must be included in any propositionalization. The absence of any relational variable (including intersection variables) may unnecessarily violate causal sufficiency, which could result in the inference of a causal dependency where conditional independence was not detected. This thesis indicates that researchers should carefully consider how to represent their domains in order to accurately reason about conditional independence, and it may be possible to correct previous studies that claimed to uncover significant effects where independence should have been observed.

- **Generalize interventions under interference.** The relational d-separation theory connects relational causal structure with conditional independence. As with traditional d-separation, this could be developed further into a theory for identifying the causal effects of relational interventions. Specifically, Pearl's *do*-calculus—a series of three rules for manipulating probability distributions based on simple interventions (setting a variable's value) [122]—could be transferred to relational or network interventions [189]. This has the potential to revolutionize and generalize how interventions are performed and measured in epidemiological studies, network marketing, and other domains. Current approaches have focused on models of interference using the potential-outcome framework with minor relaxations of the stable-unit treatment value assump-

tion for simple social networks [179]. With a relational representation, we could generalize interventions under interference to arbitrary relational effects.

- **Characterize preexisting quasi-experimental designs and identify new designs by using expressive representations.** Quasi-experimental designs (QEDs) are a suite of techniques that emulate control and randomization in order to support causal conclusions [160]. Although routinely used by social scientists, identifying applicable QEDs is a painstaking manual process, and the literature includes dozens of types of designs with many variations (e.g., instrumental variable designs, regression discontinuity designs, interrupted time-series designs). Moreover, QEDs can only be identified and applied to data with a rich underlying representation, but researchers who employ them do not rely on explicit data or model representations. This presents an opportunity to characterize and leverage new methods for causal discovery that are only applicable in relational, or more expressive, representations. This thesis provides evidence that *new* designs for learning causal constraints, such as relational bivariate orientation (RBO), can be discovered in expressive representations.

  Our experience manually applying different QEDs to real-world complex systems, including Stack Overflow [119, 141], Wikipedia [101], and IMDb [75], indicates that many QEDs utilize the same underlying basic operations. QEDs attempt to eliminate alternative causal models by controlling for various effects, but the methods in which they do so appear to be limited to a small set of operators. The design space can mostly be characterized by instances of blocking, conditioning, and sampling [101]. Relational blocking (described in Section 6.2) is one example of a QED that can be graphically characterized with a relational representation and can be integrated with a causal structure learning algorithm. Another example is the instrumental variable design generalized by Brito and Pearl that uses graphical conditions for Bayesian networks

[16]. The constraint-based approach used in this thesis for learning joint causal models is particularly extensible and conducive to incorporating new, modular operators based on these designs.

- **Build mixed-initiative systems for computer-aided causal discovery.** The focus of this thesis is on a purely automated methodology, but a long-term goal is to build interactive, mixed-initiative systems that enable an expert to be actively involved in the learning process. This extends well beyond incorporating prior knowledge, allowing a user to influence the execution of the system based on intermediate discoveries. Computer-aided causal discovery for social scientists and other researchers could dramatically change the way scientific research is conducted and causal knowledge is shared.

- **Learn causal models of high-impact real-world domains.** With the growing ability to record complex data in many domains, there is a great opportunity to apply causal discovery with expressive underlying representations. The research presented in this thesis and its near-term extensions have the potential to inform our understanding and produce positive change in systems involving social networks, health care and medicine, economic and commercial enterprises, energy and transportation networks, and defense interests. Similar to how Bayesian networks have improved the probabilistic reasoning capabilities of diagnostic systems, learning and deploying expressive causal models in real domains could advise and inform new policies as society increasingly relies on advanced technology.

BIBLIOGRAPHY

[1] Abellán, Joaquín, Gómez-Olmedo, Manuel, and Moral, Serafín. Some variations on the PC algorithm. In *Proceedings of the Third European Workshop on Probabilistic Graphical Models* (2006), pp. 1–8.

[2] Abramson, Bruce, Brown, John, Edwards, Ward, Murphy, Allan, and Winkler, Robert L. Hailfinder: A Bayesian system for forecasting severe weather. *International Journal of Forecasting 12*, 1 (1996), 57–71.

[3] Acid, Silvia, and de Campos, Luis M. A hybrid methodology for learning belief networks: BENEDICT. *International Journal of Approximate Reasoning 27*, 3 (2001), 235–262.

[4] Acid, Silvia, and de Campos, Luis M. Searching for Bayesian network structures in the space of restricted acyclic partially directed graphs. *Journal of Artificial Intelligence Research 18* (2003), 445–490.

[5] Angrist, Joshua D., and Krueger, Alan B. Instrumental variables and the search for identification: From supply and demand to natural experiments. *Journal of Economic Perspectives 15*, 4 (Fall 2001), 69–85.

[6] Angrist, Joshua D., and Pischke, Jörn-Steffen. *Mostly Harmless Econometrics: An Empiricist's Companion*. Princeton University Press, Princeton, NJ, 2009.

[7] Azzalini, Adelchi, and Bowman, Adrian W. A look at some data on the Old Faithful geyser. *Journal of the Royal Statistical Society. Series C (Applied Statistics) 39*, 3 (1990), 357–365.

[8] Barker, Richard. *CASE Method: Entity Relationship Modeling*. Addison-Wesley, Boston, MA, 1990.

[9] Beinlich, Ingo A., Suermondt, Henri J., Chavez, R. Martin, and Cooper, Gregory F. The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. In *Proceedings of the Second European Conference on Artificial Intelligence in Medicine* (1989), pp. 247–256.

[10] Berkson, Joseph. Limitations of the application of fourfold table analysis to hospital data. *Biometrics Bulletin 2*, 3 (June 1946), 47–53.

[11] Bishop, Christopher M. Latent variable models. In *Learning in Graphical Models*, Michael I. Jordan, Ed. MIT Press, Cambridge, MA, 1999, pp. 371–403.

[12] Borboudakis, Giorgos, and Tsamardinos, Ioannis. Incorporating causal prior knowledge as path-constraints in Bayesian networks and maximal ancestral graphs. In *Proceedings of the Twenty-Ninth International Conference on Machine Learning* (2012), pp. 1799–1806.

[13] Borboudakis, Giorgos, and Tsamardinos, Ioannis. Scoring and searching over Bayesian networks with causal and associative priors. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence* (2013), pp. 102–111.

[14] Boutilier, Craig, Friedman, Nir, Goldszmidt, Moises, and Koller, Daphne. Context-specific independence in Bayesian networks. In *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence* (1996), pp. 115–123.

[15] Bowers, Jake, Fredrickson, Mark M., and Panagopoulos, Costas. Reasoning about interference between units: A general framework. *Political Analysis 21*, 1 (2013), 97–124.

[16] Brito, Carlos, and Pearl, Judea. Generalized instrumental variables. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence* (2002), pp. 85–93.

[17] Buntine, Wray L. Operations for learning with graphical models. *Journal of Artificial Intelligence Research 2* (1994), 159–225.

[18] Campbell, Donald, and Stanley, Julian. *Experimental and Quasi-Experimental Designs for Research*. Rand McNally, Chicago, IL, 1966.

[19] Charniak, Eugene. Bayesian networks without tears. *AI Magazine 12*, 4 (1991), 50–63.

[20] Chen, Xue-wen, Anantha, Gopalakrishna, and Wang, Xinkun. An effective structure learning method for constructing gene networks. *Bioinformatics 22*, 11 (2006), 1367–1374.

[21] Cheng, Jie, Bell, David A., and Liu, Weiru. Learning belief networks from data: An information theory based approach. In *Proceedings of the Sixth International Conference on Information and Knowledge Management* (1997), pp. 325–331.

[22] Chickering, David Maxwell. Learning Bayesian networks is NP-complete. In *Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics* (1996), pp. 121–130.

[23] Chickering, David Maxwell. Optimal structure identification with greedy search. *Journal of Machine Learning Research 3*, 3 (2002), 507–554.

[24] Claassen, Tom, and Heskes, Tom. A logical characterization of constraint-based causal discovery. In *Proceedings of Twenty-Seventh Conference on Uncertainty in Artificial Intelligence* (2011), pp. 135–144.

[25] Claassen, Tom, and Heskes, Tom. A Bayesian approach to constraint based causal inference. In *Proceedings of Twenty-Eighth Conference on Uncertainty in Artificial Intelligence* (2012), pp. 207–216.

[26] Colombo, Diego, and Maathuis, Marloes H. Order-Independent Constraint-Based Causal Structure Learning. *arXiv preprint arXiv:1211.3295* (2013).

[27] Colombo, Diego, Maathuis, Marloes H., Kalisch, Markus, and Richardson, Thomas S. Learning high-dimensional directed acyclic graphs with latent and selection variables. *The Annals of Statistics 40*, 1 (2012), 294–321.

[28] Cooper, Gregory F., and Herskovits, Edward. A Bayesian method for the induction of probabilistic networks from data. *Machine learning 9*, 4 (1992), 309–347.

[29] Cowell, Robert G. Conditions under which conditional independence and scoring methods lead to identical selection of Bayesian network models. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence* (2001), pp. 91–97.

[30] Daly, Rónán, and Shen, Qiang. Learning Bayesian network equivalence classes with ant colony optimization. *Journal of Artificial Intelligence Research 35* (2009), 391–447.

[31] D'Ambrosio, Bruce, Altendorf, Eric, and Jorgensen, Jane. Ecosystem analysis using probabilistic relational modeling. In *Proceedings of the International Joint Conference on Artificial Intelligence Workshop on Learning Statistical Models from Relational Data* (2003).

[32] Darwiche, Adnan. Bayesian networks. *Communications of the ACM 53*, 12 (2010), 80–90.

[33] Dash, Denver. Restructuring dynamic causal systems in equilibrium. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics* (2005), pp. 81–88.

[34] Dawid, A. Philip. Conditional independence in statistical theory. *Journal of the Royal Statistical Society. Series B (Methodological) 41*, 1 (1979), 1–31.

[35] de Jongh, Martijn, and Drużdżel, Marek J. A comparison of structural distance measures for causal Bayesian network models. In *Recent Advances in Intelligent Information Systems, Challenging Problems of Science, Computer Science series*, Mieczyslaw Klopotek, Adam Przepiorkowski, Slawomir T. Wierzchon, and Krzysztof Trojanowski, Eds. Academic Publishing House EXIT, 2009, pp. 443–456.

[36] Dean, Thomas, and Kanazawa, Keiji. A model for reasoning about persistence and causation. *Computational Intelligence 5*, 3 (1989), 142–150.

[37] Eberhardt, Frederick. Experimental indistinguishability of causal structures. *Philosophy of Science 80*, 5 (2013), 684–696.

[38] Fast, Andrew S. *Learning the Structure of Bayesian Networks with Constraint Satisfaction*. Ph.D. thesis, University of Massachusetts Amherst, 2010.

[39] Fisher, Ronald A. *The Design of Experiments*. Oliver and Boyd, Edinburgh, 1935.

[40] Flach, Peter A. Knowledge representation for inductive learning. In *Proceedings of the Fifth European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty* (1999), pp. 160–167.

[41] Frank, Ove, and Strauss, David. Markov graphs. *Journal of the American Statistical Association 81*, 395 (1986), 832–842.

[42] Friedman, Nir. Inferring cellular networks using probabilistic graphical models. *Science 303*, 5659 (2004), 799–805.

[43] Friedman, Nir, Getoor, Lise, Koller, Daphne, and Pfeffer, Avi. Learning probabilistic relational models. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence* (1999), vol. 16, pp. 1300–1309.

[44] Friedman, Nir, Nachman, Iftach, and Pe'er, Dana. Learning Bayesian network structure from massive datasets: The "sparse candidate" algorithm. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence* (1999), pp. 206–215.

[45] Fung, Robert, and Del Favero, Brendan. Applying Bayesian networks to information retrieval. *Communications of the ACM 38*, 3 (1995), 42–48.

[46] Garfield, Eugene. Citation analysis as a tool in journal evaluation. *Science 178* (1972), 471–479.

[47] Geiger, Dan, and Pearl, Judea. On the logic of causal models. In *Proceedings of the Fourth Annual Conference on Uncertainty in Artificial Intelligence* (1988), pp. 136–147.

[48] Geiger, Dan, Verma, Thomas, and Pearl, Judea. d-separation: From theorems to algorithms. In *Proceedings of the Fifth Conference on Uncertainty in Artificial Intelligence* (1989), pp. 118–125.

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

[50] Gelman, Andrew. Scaling regression inputs by dividing by two standard deviations. *Statistics in Medicine 27*, 15 (2008), 2865–2873.

[51] Gelman, Andrew, and Hill, Jennifer. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press, New York, 2007.

[52] Getoor, Lise. *Learning Statistical Models from Relational Data*. Ph.D. thesis, Stanford University, 2001.

[53] Getoor, Lise, Friedman, Nir, Koller, Daphne, Pfeffer, Avi, and Taskar, Ben. Probabilistic relational models. In *Introduction to Statistical Relational Learning*, Lise Getoor and Ben Taskar, Eds. MIT Press, Cambridge, MA, 2007, ch. 5, pp. 129–174.

[54] Getoor, Lise, Friedman, Nir, Koller, Daphne, and Taskar, Ben. Learning probabilistic models of link structure. *Journal of Machine Learning Research 3* (2002), 679–707.

[55] Getoor, Lise, Koller, Daphne, and Friedman, Nir. From instances to classes in probabilistic relational models. In *Proceedings of the International Conference on Machine Learning Workshop on Attribute-Value and Relational Learning* (2000).

[56] Getoor, Lise, Rhee, Jeanne T., Koller, Daphne, and Small, Peter. Understanding tuberculosis epidemiology using structured statistical models. *Artificial Intelligence in Medicine 30*, 3 (2004), 233–256.

[57] Getoor, Lise, and Taskar, Ben, Eds. *Introduction to Statistical Relational Learning*. MIT Press, Cambridge, MA, 2007.

[58] Gilks, Walter R., Thomas, Andrew, and Spiegelhalter, David J. A language and program for complex Bayesian modeling. *The Statistician 43* (1994), 169–177.

[59] Gretton, Arthur, Fukumizu, Kenji, Teo, Choon Hui, Song, Le, Schölkopf, Bernhard, and Smola, Alex J. A kernel statistical test of independence. In *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems*. 2007, pp. 585–592.

[60] Harris, Naftali, and Drton, Mathias. PC algorithm for nonparanormal graphical models. *Journal of Machine Learning Research 14*, 1 (2013), 3365–3383.

[61] Heckerman, David. Probabilistic similarity networks. *Networks 20*, 5 (1990), 607–636.

[62] Heckerman, David, Meek, Chris, and Koller, Daphne. Probabilistic entity-relationship models, PRMs, and plate models. In *Introduction to Statistical Relational Learning*, Lise Getoor and Ben Taskar, Eds. MIT Press, Cambridge, MA, 2007, ch. 7, pp. 201–238.

[63] Heckerman, David, Meek, Christopher, and Koller, Daphne. *Probabilistic Models for Relational Data*. Tech. Rep. MSR-TR-2004-30, Microsoft Research, Redmond, WA, March 2004.

[64] Hirsch, Jorge E. An index to quantify an individual's scientific research output. *Proceedings of the National Academy of Sciences of the United States of America 102*, 46 (2005), 16569–16572.

[65] Holland, Paul W. Statistics and causal inference. *Journal of the American Statistical Association 81*, 396 (December 1986), 945–960.

[66] Holland, Paul W., and Leinhardt, Samuel. An exponential family of probability distributions for directed graphs. *Journal of the American Statistical Association 76*, 373 (1981), 33–50.

[67] Hox, Joop J. *Multilevel Analysis: Techniques and Applications*, 2nd ed. Taylor & Francis, New York, 2010.

[68] Hoyer, Patrik O., Janzing, Dominik, Mooij, Joris M., Peters, Jonas, and Schölkopf, Bernhard. Nonlinear causal discovery with additive noise models. In *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems*. 2009, pp. 689–696.

[69] Huang, Yimin, and Valtorta, Marco. Identifiability in causal Bayesian networks: A sound and complete algorithm. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence* (2006), pp. 1149–1154.

[70] Hudgens, Michael G., and Halloran, M. Elizabeth. Toward causal inference with interference. *Journal of the American Statistical Association 103*, 482 (2008), 832–842.

[71] Hughes, Timothy R., Marton, Matthew J., Jones, Allan R., Roberts, Christopher J., Stoughton, Roland, Armour, Christopher D., Bennett, Holly A., Coffey, Ernest, Dai, Hongyue, He, Yudong D., Kidd, Matthew J., King, Amy M., Meyer, Michael R., Slade, David, Lum, Pek Y., Stepaniants, Sergey B., Shoemaker, Daniel D., Gachotte, Daniel, Chakraburtty, Kalpana, Simon, Julian, Bard, Martin, and Friend, Stephen H. Functional discovery via a compendium of expression profiles. *Cell 102*, 1 (2000), 109–126.

[72] Hyttinen, Antti, Eberhardt, Frederick, and Hoyer, Patrik O. Learning linear cyclic causal models with latent variables. *Journal of Machine Learning Research 13* (2012), 3387–3439.

[73] Ide, Jaime S., and Cozman, Fabio G. Random generation of Bayesian networks. In *Advances in Artificial Intelligence, Proceedings of the Sixteenth Brazilian Symposium on Artificial Intelligence*. 2002, pp. 366–376.

[74] Ide, Jaime S., Cozman, Fabio G., and Ramos, Fabio T. Generating random Bayesian networks with constraints on induced width. In *Proceedings of the Sixteenth European Conference on Artificial Intelligence* (2004), pp. 323–327.

[75] Jensen, David, Fast, Andrew, Taylor, Brian, and Maier, Marc. Automatic identification of quasi-experimental designs for discovering causal knowledge. In *Proceedings of the Fourteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2008), pp. 372–380.

[76] Jensen, David, and Neville, Jennifer. Linkage and autocorrelation cause feature selection bias in relational learning. In *Proceedings of the Nineteenth International Conference on Machine Learning* (2002), pp. 259–266.

[77] Jensen, David, Neville, Jennifer, and Hay, Michael. Avoiding bias when aggregating relational data with degree disparity. In *Proceedings of the Twentieth International Conference on Machine Learning* (2003), pp. 274–281.

[78] Jensen, Finn V. *Introduction to Bayesian Networks*. Springer-Verlag, New York, 1996.

[79] Kalisch, Markus, Mächler, Martin, Colombo, Diego, Maathuis, Marloes H., and Bühlmann, Peter. Causal inference using graphical models with the R package pcalg. *Journal of Statistical Software 47*, 11 (2012), 1–26.

[80] Kenny, David A., and Judd, Charles M. Consequences of violating the independence assumption in analysis of variance. *Psychological Bulletin 99*, 3 (1986), 422–431.

[81] Kersting, Kristian, and De Raedt, Luc. *Basic Principles of Learning Bayesian Logic Programs*. Tech. Rep. 174, Institute for Computer Science, University of Freiberg, 2002.

[82] Kok, Stanley, and Domingos, Pedro. Learning the structure of Markov logic networks. In *Proceedings of the Twenty-Second International Conference on Machine Learning* (2005), pp. 441–448.

[83] Kok, Stanley, and Domingos, Pedro. Learning Markov logic networks using structural motifs. In *Proceedings of the Twenty-Seventh International Conference on Machine Learning* (2010), pp. 551–558.

[84] Koller, Daphne, and Friedman, Nir. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, Cambridge, MA, 2009.

[85] Koller, Daphne, and Pfeffer, Avi. Probabilistic frame-based systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence* (1998), pp. 580–587.

[86] Kpotufe, Samory, Sgouritsa, Eleni, Janzig, Dominik, and Schölkopf, Bernhard. Consistency of Causal Inference under the Additive Noise Model. *arXiv preprint arXiv:1312.5770* (2013).

[87] Kramer, Stefan, Lavrač, Nada, and Flach, Peter. Propositionalization approaches to relational data mining. In *Relational Data Mining*, Sašo Džeroski and Nada Lavrač, Eds. Springer-Verlag, New York, NY, 2001, pp. 262–286.

[88] Krogel, Mark-André. *On Propositionalization for Knowledge Discovery in Relational Databases*. Ph.D. thesis, Otto von Guericke University Magdeburg, 2005.

[89] Laskey, Kathryn B. MEBN: A language for first-order Bayesian knowledge bases. *Artificial Intelligence 172*, 2 (2008), 140–178.

[90] Lauritzen, Steffen L., and Spiegelhalter, David J. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society. Series B (Methodological) 50*, 2 (1988), 157–224.

[91] Lavrač, Nada. *Principles of Knowledge Acquisition in Expert Systems*. Ph.D. thesis, Faculty of Technical Sciences, University of Maribor, 1990.

[92] Lavrač, Nada, Železný, Filip, and Flach, Peter. RSD: Relational subgroup discovery through first-order feature construction. In *Proceedings of the Twelfth International Conference on Inductive Logic Programming* (2002), pp. 149–165.

[93] Lee, Valerie E., and Bryk, Anthony S. A multilevel model of the social distribution of high school achievement. *Sociology of Education 62*, 3 (July 1989), 172–192.

[94] Lemeire, Jan, Meganck, Stijn, and Cartella, Francesco. Robust independence-based causal structure learning in absence of adjacency faithfulness. In *Proceedings of the Fifth European Workshop on Probabilistic Graphical Models* (2010), pp. 169–177.

[95] Maathuis, Marloes H., Colombo, Diego, Kalisch, Markus, and Bühlmann, Peter. Predicting causal effects in large-scale systems from observational data. *Nature Methods 7*, 4 (2010), 247–248.

[96] Maathuis, Marloes H., Kalisch, Markus, and Bühlmann, Peter. Estimating high-dimensional intervention effects from observational data. *Annals of Statistics 37*, 6A (2009), 3133–3164.

[97] Mahdi, Rami, and Mezey, Jason. Sub-local constraint-based learning of Bayesian networks using a joint dependence criterion. *Journal of Machine Learning Research 14* (2013), 1563–1603.

[98] Maier, Marc, Marazopoulou, Katerina, Arbour, David, and Jensen, David. Flattening network data for causal discovery: What could go wrong? *Workshop on Information in Networks* (2013).

[99] Maier, Marc, Marazopoulou, Katerina, Arbour, David, and Jensen, David. A sound and complete algorithm for learning causal models from relational data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence* (2013), pp. 371–380.

[100] Maier, Marc, Marazopoulou, Katerina, and Jensen, David. Reasoning about Independence in Probabilistic Models of Relational Data. *arXiv preprint arXiv:1302.4381* (2013).

[101] Maier, Marc, Rattigan, Matthew, and Jensen, David. *Discovering Causal Knowledge by Design*. Tech Report 09-47, University of Massachusetts Amherst, Computer Science Department, 2009.

[102] Maier, Marc, Taylor, Brian, Oktay, Hüseyin, and Jensen, David. Learning causal models of relational domains. In *Proceedings of the Twenty-Fourth National Conference on Artificial Intelligence* (2010), pp. 531–538.

[103] Manski, Charles F. Identification of treatment response with social interactions. *The Econometrics Journal 16*, 1 (2013), S1–S23.

[104] Margaritis, Dimitris, and Thrun, Sebastian. Bayesian network induction via local neighborhoods. In *Advances in Neural Information Processing Systems 12, Proceedings of the Thirteenth Annual Conference on Neural Information Processing Systems*. 1999, pp. 505–511.

[105] Margaritis, Dimitris, and Thrun, Sebastian. A Bayesian multiresolution independence test for continuous variables. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence* (2001), pp. 346–353.

[106] Meek, Christopher. Causal inference and causal explanation with background knowledge. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence* (1995), pp. 403–410.

[107] Melançon, Guy, Dutour, Isabelle, and Bousquet-Mélou, Mirelle. *Random Generation of DAGs for Graph Drawing*. Tech. Rep. INS-R0005, Dutch Research Center for Mathematical and Computer Science (CWI), 2000.

[108] Mihalkova, Lilyana, and Mooney, Raymond J. Bottom-up learning of Markov logic network structure. In *Proceedings of the Twenty-Fourth International Conference on Machine Learning* (2007), pp. 625–632.

[109] Milch, Brian, Marthi, Bhaskara, Russell, Stuart J., Sontag, David, Ong, Daniel L., and Kolobov, Andrey. BLOG: Probabilistic models with unknown objects. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence* (2005), pp. 1352–1359.

[110] Milch, Brian, and Russell, Stuart. First-order probabilistic languages: Into the unknown. In *Inductive Logic Programming: Proceedings of the Sixteenth International Conference on Inductive Logic Programming* (2007), pp. 10–24.

[111] Minka, Tom, and Winn, John. Gates. In *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems*. 2009, pp. 1073–1080.

[112] Murphy, Kevin P. *Dynamic Bayesian Networks: Representation, Inference and Learning*. Ph.D. thesis, University of California, Berkeley, 2002.

[113] Natarajan, Sriraam, Tadepalli, Prasad, Altendorf, Eric, Dietterich, Thomas G., Fern, Alan, and Restificar, Angelo C. Learning first-order probabilistic models with combining rules. In *Proceedings of the Twenty-Second International Conference on Machine Learning* (2005), pp. 609–616.

[114] Neal, Radford M. On deducing conditional independence from d-separation in causal graphs with feedback. *Journal of Artificial Intelligence Research 12*, 1 (February 2000), 87–91.

[115] Neapolitan, Richard E. *Learning Bayesian Networks*. Pearson Prentice Hall, Upper Saddle River, NJ, 2004.

[116] Neville, Jennifer, and Jensen, David. Leveraging relational autocorrelation with latent group models. In *Proceedings of the Fifth IEEE International Conference on Data Mining* (2005), pp. 322–329.

[117] Neville, Jennifer, and Jensen, David. Relational dependency networks. *Journal of Machine Learning Research 8* (May 2007), 653–692.

[118] Neyman, Jerzy. On the application of probability theory to agricultural experiments. Essay on Principles. Section 9. *Statistical Science 5*, 4 (1923), 465–472.

[119] Oktay, Hüseyin, Taylor, Brian J., and Jensen, David. Causal discovery in social media using quasi-experimental designs. In *Proceedings of the SIGKDD/ACM Workshop on Social Media Analytics* (2010).

[120] Pearl, Judea. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Fransico, CA, 1988.

[121] Pearl, Judea. Causal diagrams for empirical research. *Biometrika 82*, 4 (1995), 669–688.

[122] Pearl, Judea. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, New York, NY, 2000.

[123] Pearl, Judea. Causal inference in statistics: An overview. *Statistics Surveys 3* (2009), 96–146.

[124] Pearl, Judea. Letter to the editor: Remarks on the method of propensity score. *Statistics in Medicine 28* (2009), 1415–1416.

[125] Pearl, Judea. On a class of bias-amplifying variables that endanger effect estimates. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence* (2010), pp. 417–424.

[126] Pearl, Judea, and Dechter, Rina. Identifying independencies in causal graphs with feedback. In *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence* (1996), pp. 420–426.

[127] Pellet, Jean-Philippe, and Elisseeff, André. Using Markov blankets for causal structure learning. *Journal of Machine Learning Research 9* (2008), 1295–1342.

[128] Pellet, Jean-Philippe, and Elisseeff, André. Finding latent causes in causal networks: An efficient approach based on Markov blankets. In *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems*. 2009, pp. 1249–1256.

[129] Perlich, Claudia, and Provost, Foster. Distribution-based aggregation for relational learning with identifier attributes. *Machine Learning 62*, 1-2 (2006), 65–105.

[130] Perrier, Eric, Imoto, Seiya, and Miyano, Satoru. Finding optimal Bayesian network given a super-structure. *Journal of Machine Learning Research 9* (2008), 2251–2286.

[131] Peters, Jonas, and Bühlmann, Peter. Structural Intervention Distance (SID) for Evaluating Causal Graphs. *arXiv preprint arXiv:1306.1043* (2013).

[132] Peters, Jonas, Janzing, Dominik, and Schölkopf, Bernhard. Causal inference on discrete data using additive noise models. *IEEE Transactions on Pattern Analysis and Machine Intelligence 33*, 12 (2011), 2436–2450.

[133] Peters, Jonas, Mooij, Joris M., Janzing, Dominik, and Schölkopf, Bernhard. Identifiability of causal graphs using functional models. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence* (2011), pp. 589–598.

[134] Poole, David. First-order probabilistic inference. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence* (2003), vol. 3, pp. 985–991.

[135] Poole, David. Logical generative models for probabilistic reasoning about existence, roles and identity. In *Proceedings of the Twenty-Second National Conference on Artificial Intelligence* (2007), pp. 1271–1277.

[136] Ramakrishnan, Raghu, and Gehrke, Johannes. *Database Management Systems*, 2nd ed. McGraw-Hill, Inc., New York, NY, 2002.

[137] Ramsey, Joseph, Spirtes, Peter, and Zhang, Jiji. Adjacency-faithfulness and conservative causal inference. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence* (2006), pp. 401–408.

[138] Ramsey, Joseph D. A Scalable Conditional Independence Test for Nonlinear, Non-Gaussian Data. *arXiv preprint arXiv:1401.5031* (2014).

[139] Rattigan, Matthew J. *Leveraging Relational Representations for Causal Discovery*. Ph.D. thesis, University of Massachusetts Amherst, 2012.

[140] Rattigan, Matthew J., and Jensen, David. Leveraging d-separation for relational data sets. In *Proceedings of the Tenth IEEE International Conference on Data Mining* (2010), pp. 989–994.

[141] Rattigan, Matthew J., Maier, Marc, and Jensen, David. Relational blocking for causal discovery. In *Proceedings of the Twenty-Fifth National Conference on Artificial Intelligence* (2011), pp. 145–151.

[142] Rehg, James M., Murphy, Kevin P., and Fieguth, Paul W. Vision-based speaker detection using Bayesian networks. In *Proceedings of the Thirteenth IEEE Conference on Computer Vision and Pattern Recognition* (1999), pp. 110–116.

[143] Richardson, Matthew, and Domingos, Pedro. Markov logic networks. *Machine Learning 62*, 1–2 (2006), 107–136.

[144] Richardson, Thomas, and Spirtes, Peter. Ancestral graph Markov models. *The Annals of Statistics 30*, 4 (2002), 962–1030.

[145] Richardson, Thomas S. *Feedback Models: Interpretation and Discovery*. Ph.D. thesis, Carnegie Mellon University, 1996.

[146] Richardson, Thomas S. A factorization criterion for acyclic directed mixed graphs. *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence* (2009), 462–470.

[147] Robins, Garry, Pattison, Pip, Kalish, Yuval, and Lusher, Dean. An introduction to exponential random graph (p*) models for social networks. *Social Networks 29* (2007), 173–191.

[148] Rosenbaum, Paul R. Interference between units in randomized experiments. *Journal of the American Statistical Association 102*, 477 (2007), 191–200.

[149] Rosenbaum, Paul R., and Rubin, Donald B. The central role of the propensity score in observational studies for causal effects. *Biometrika 70*, 1 (1983), 41–55.

[150] Rubin, Donald B. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology 66*, 5 (October 1974), 688–701.

[151] Russell, Stuart, and Norvig, Peter. *Artificial Intelligence: A Modern Approach*, 3rd ed. Prentice Hall, 2010.

[152] Sachs, Karen, Perez, Omar, Pe'er, Dana, Lauffenburger, Douglas A., and Nolan, Garry P. Causal protein-signaling networks derived from multiparameter single-cell data. *Science 308*, 5721 (2005), 523–529.

[153] Sadeghi, Kayvan, and Lauritzen, Steffen. Markov properties for mixed graphs. *Bernoulli 20*, 2 (2014), 676–696.

[154] Scheines, Richard. An introduction to causal inference. In *Causality in Crisis? Statistical Methods and the Search for Causal Knowledge in the Social Sciences*, Vaughan R. McKim and Steven P. Turner, Eds. University of Notre Dame Press, 1997, pp. 185–199.

[155] Schmidt, Mark, and Murphy, Kevin. Modeling discrete interventional data using directed cyclic graphical models. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence* (2009), pp. 487–495.

[156] Schulte, Oliver, Khosravi, Hassan, and Man, Tong. Learning directed relational models with recursive dependencies. *Machine Learning 89*, 3 (2012), 299–316.

[157] Scott, John, and Carrington, Peter J., Eds. *The SAGE Handbook of Social Network Analysis*. SAGE Publications, London, 2011.

[158] Segal, Eran, Taskar, Ben, Gasch, Audrey, Friedman, Nir, and Koller, Daphne. Rich probabilistic models for gene expression. *Bioinformatics 17*, suppl 1 (2001), S243–S252.

[159] Shachter, Ross D. Bayes-Ball: The rational pastime (for determining irrelevance and requisite information in belief networks and influence diagrams). In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence* (1998), pp. 480–487.

[160] Shadish, William R., Cook, Thomas D., and Campbell, Donald T. *Experimental and Quasi-Experimental Designs for Generalized Causal Inference*. Houghton Mifflin, Boston, MA, 2002.

[161] Shalizi, Cosma R., and Thomas, Andrew C. Homophily and contagion are generically confounded in observational social network studies. *Sociological Methods & Research 40*, 2 (2011), 211–239.

[162] Shi, Chuan, Kong, Xiangnan, Yu, Philip S., Xie, Sihong, and Wu, Bin. Relevance search in heterogeneous networks. In *Proceedings of the Fifteenth International Conference on Extending Database Technology* (2012), pp. 180–191.

[163] Shimizu, Shohei, Hoyer, Patrik O., Hyvärinen, Aapo, and Kerminen, Antti. A linear non-Gaussian acyclic model for causal discovery. *Journal of Machine Learning Research 7* (2006), 2003–2030.

[164] Shpitser, Ilya, and Pearl, Judea. Complete identification methods for the causal hierarchy. *Journal of Machine Learning Research 9* (2008), 1941–1979.

[165] Shpitser, Ilya, VanderWeele, Tyler J., and Robins, James M. On the validity of covariate adjustment for estimating causal effects. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence* (2010), pp. 527–536.

[166] Shrier, Ian. Letter to the editor. *Statistics in Medicine 27* (2008), 2740–2741.

[167] Singh, Moninder, and Valtorta, Marco. An algorithm for the construction of Bayesian network structures from data. In *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence* (1993), pp. 259–265.

[168] Sjölander, Arvid. Letter to the editor: Propensity scores and M-structures. *Statistics in Medicine 28* (2009), 1416–1420.

[169] Snijders, Tom A. B., and Bosker, Roel J. *Multilevel Analysis: An Introduction to Basic and Advanced Multilevel Modeling*, 2nd ed. Sage, London, 2011.

[170] Sommestad, Teodor, Ekstedt, Mathias, and Johnson, Pontus. A probabilistic relational model for security risk analysis. *Computers & Security 29*, 6 (2010), 659–679.

[171] Spirtes, Peter. Directed cyclic graphical representations of feedback models. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence* (1995), pp. 491–498.

[172] Spirtes, Peter, Glymour, Clark, and Scheines, Richard. *Causation, Prediction and Search*, 2nd ed. MIT Press, Cambridge, MA, 2000.

[173] Spirtes, Peter, Meek, Christopher, and Richardson, Thomas. Causal inference in the presence of latent variables and selection bias. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence* (1995), pp. 499–506.

[174] Steck, Harald. On the use of skeletons when learning in Bayesian networks. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence* (2000), pp. 558–565.

[175] Steck, Harald, and Tresp, Volker. Bayesian belief networks for data mining. In *Proceedings of the Second Workshop on Data Mining and Data Warehousing* (1999), pp. 145–154.

[176] Sun, Yizhou, Han, Jiawei, Yan, Xifeng, Yu, Philip S., and Wu, Tianyi. PathSim: Meta path-based top-k similarity search in heterogeneous information networks. In *Proceedings of the VLDB Endowment* (2011), pp. 992–1003.

[177] Taskar, Ben, Abbeel, Pieter, and Koller, Daphne. Discriminative probabilistic models for relational data. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence* (2002), pp. 485–492.

[178] Taskar, Ben, Segal, Eran, and Koller, Daphne. Probabilistic classification and clustering in relational data. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence* (2001), pp. 870–878.

[179] Tchetgen Tchetgen, Eric J., and VanderWeele, Tyler J. On causal inference in the presence of interference. *Statistical Methods in Medical Research 21*, 1 (2012), 55–75.

[180] ten Cate, Balder. On the logic of d-separation. In *Proceedings of the Eighth International Conference on Principles of Knowledge Representation and Reasoning* (2002), pp. 568–577.

[181] Teyssier, Marc, and Koller, Daphne. Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence* (2005), pp. 584–590.

[182] Tian, Jin, Paz, Azaria, and Pearl, Judea. *Finding Minimal D-separators*. Tech. Rep. R-254, UCLA, Computer Science Department, February 1998.

[183] Tian, Jin, and Pearl, Judea. A general identification condition for causal effects. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence* (2002), pp. 567–573.

[184] Tibshirani, Robert. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), 267–288.

[185] Tillman, Robert E., Gretton, Arthur, and Spirtes, Peter. Nonlinear directed acyclic structure learning with weakly additive noise models. In *Advances in Neural Information Processing Systems 22, Proceedings of the Twenty-Third Annual Conference on Neural Information Processing Systems*. 2009, pp. 1847–1855.

[186] Toulis, Panos, and Kao, Edward. Estimation of causal peer influence effects. In *Proceedings of the Thirtieth International Conference on Machine Learning* (2013), pp. 1489–1497.

[187] Tsamardinos, Ioannis, and Borboudakis, Giorgos. Permutation testing improves Bayesian network learning. In *Machine Learning and Knowledge Discovery in Databases Part III, Proceedings of the Twenty-First European Conference on Machine Learning*. 2010, pp. 322–337.

[188] Tsamardinos, Ioannis, Brown, Laura E., and Aliferis, Constantin F. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning 65*, 1 (October 2006), 31–78.

[189] Valente, Thomas W. Network interventions. *Science 337*, 6090 (2012), 49–53.

[190] VanderWeele, Tyler J., Vandenbroucke, Jan P., Tchetgen Tchetgen, Eric J., and Robins, James M. A mapping between interactions and interference: Implications for vaccine trials. *Epidemiology 23*, 2 (March 2012), 285–292.

[191] Verma, Thomas, and Pearl, Judea. Causal networks: Semantics and expressiveness. In *Proceedings of the Fourth Annual Conference on Uncertainty in Artificial Intelligence* (1988), pp. 352–359.

[192] Wermuth, Nanny. Probability distributions with summary graph structure. *Bernoulli 17*, 3 (2011), 845–879.

[193] Winn, John. Causality with gates. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics* (2012), pp. 1314–1322.

[194] Woodward, James. *Making Things Happen: A Theory of Causal Explanation*. Oxford University Press, USA, 2003.

[195] Xiang, Rongjing, and Neville, Jennifer. Relational learning with one network: An asymptotic analysis. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics* (2011), pp. 779–788.

[196] Xie, Xianchao, and Geng, Zhi. A recursive method for structural learning of directed acyclic graphs. *Journal of Machine Learning Research 9* (March 2008), 459–483.

[197] Yehezkel, Raanan, and Lerner, Boaz. Bayesian network structure learning by recursive autonomy identification. *Journal of Machine Learning Research 10* (2009), 1527–1570.

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

[199] Zhang, Jiji, and Spirtes, Peter. Intervention, determinism, and the causal minimality condition. *Synthese 182*, 3 (2011), 335–347.

[200] Zhang, Kun, and Hyvärinen, Aapo. On the identifiability of the post-nonlinear causal model. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence* (2009), pp. 647–655.

[201] Zhang, Kun, Peters, Jonas, Janzing, Dominik, and Schölkopf, Bernhard. Kernel-based conditional independence test and application in causal discovery. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence* (2011), pp. 804–813.

[202] Zhang, Xinhua, Song, Le, Gretton, Arthur, and Smola, Alex J. Kernel measures of independence for non-iid data. In *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems*. 2009, pp. 1937–1944.