TEMPORAL AND RELATIONAL GRAPHICAL MODELS FOR CAUSALITY: REPRESENTATION AND LEARNING

A Dissertation Presented

by

KATERINA MARAZOPOULOU

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 2017

College of Information and Computer Sciences

TEMPORAL AND RELATIONAL GRAPHICAL MODELS FOR CAUSALITY: REPRESENTATION AND LEARNING

A Dissertation Presented

by

KATERINA MARAZOPOULOU

Approved as to style and content by:

DEDICATION

To my grandparents.

ACKNOWLEDGMENTS

This thesis would not have being possible without my advisor, David Jensen. David has been a key factor to starting, surviving, and completing this adventure. I am sincerely grateful to him for his guidance throughout the years, his constant support, his contagious enthusiasm, for always keeping an open mind to new ideas, for always being understanding, for his wise advice on so many areas.

I would also like to thank my committee members—Justin Gross, Ben Marlin, and Dan Sheldon—for their feedback. Moreover, I am grateful to Dan Corkill for his insightful comments, to Cindy Loiselle for proof-reading so many of my drafts, and to Deb Bergeron for taking care of all administrative tasks so seemingly effortlessly. Finally, Leeanne, thank you for helping me navigate the bureaucratic aspects of grad school and making sure that everything goes (almost) according to plan.

I would like to thank my labmates throughout the years, for all the inspiring conversations we had, the late-night paper submissions, and—plain simple—for their company: David Arbour, James Atwood, Elisabeth Baseman, Javier Burroni, Kaleigh Clary, Lisa Friedland, Dan Garant, Amanda Gentzel, Phil Kirlin, Marc Maier, Huseyin Oktay, Matt Rattigan, Brian Taylor, Sam Witty. Among them, I would especially like to thank my co-authors. First of all, Marc Maier, who took me under his wing when I first joined the lab. His patience and attention to detail made him one of the best examples to follow, although his legendary time-management skills will be hard to match for generations of KDL students to come. I need to separately mention and thank David Arbour, with whom our graduate studies overlapped the most. His never-ending production of ideas, his deep knowledge of the literature, his valuable input on my work, and—let's face it—his cynicism have been an irreplaceable part of my graduate studies. Finally, I would like to thank Dan Garant, for counterbalancing David's cynicism, for making stack overflow unnecessary when he was around, for always being positive, creative, and helpful.

There have been many people in Amherst, friends and housemates, that made my life here so enjoyable, each in their own unique way. Laura, Gerard R.C., John, Steven, Cindy, Gerard M., Britta, Marie, Francesca, Sam P., Marc, Ilene, Sam H., Phil, Giorgo, Nikola, Stefan, Shiraj, Dirk, Jae, Zeki, Sophal, Pano, Rodrigo, Kevin: life in western Mass would not have been the same without you. I am also grateful to my friends back in Greece and to my extended family. Despite the thousands of miles (and the seven time zones) between us, you have remained a big part of my life and you make home still feel like home. Last, but not least, I would like to thank Konstantinos, my "fellow traveller" in studies and, more importantly, in life. Thank you for the countless hours of conversations, your emotional and academic support, for always being there for me, for always believing in me.

Finally, words are not enough to express how grateful I am to my family—my amazing parents and my incredible sister—for their unlimited support, endless understanding, and unconditional love. Thank you for giving me the foundations to pursue my dreams and for encouraging me every step of the way. 101055 I cannot believe how lucky I have been to have you as my family.

ABSTRACT

TEMPORAL AND RELATIONAL GRAPHICAL MODELS FOR CAUSALITY: REPRESENTATION AND LEARNING

SEPTEMBER 2017

KATERINA MARAZOPOULOU

Diploma, NATIONAL TECHNICAL UNIVERSITY OF ATHENS

M.Sc., IMPERIAL COLLEGE LONDON

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor David Jensen

Discovering causal dependence is central to understanding the behavior of complex systems and to selecting actions that will achieve particular outcomes. The majority of work in this area has focused on propositional domains, where data instances are assumed to be independent and identically distributed (i.i.d.). However, many real-world domains are inherently *relational*, i.e., they consist of multiple types of entities that interact with each other, and *temporal*, i.e., they change over time. This thesis focuses on causal modeling for these more complex relational and temporal domains. This thesis provides an in-depth investigation of the properties of relational models and is extending their expressivity to include a temporal dimension. Specifically, we first investigate alternative ways to ground relational models, and we provide an in-depth analysis of the impact of alternative grounding semantics for feature construction, causal effect estimation, and model selection. Then, we extend relational models to represent discrete time. We generalize the theory of d-separation for this class of temporal and relational models. Finally, we provide a constraint-based algorithm, TRCD, to learn the structure of temporal relational models from data.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

We are surrounded by complex systems, such as social networks, trading systems, manufacturing plants, health monitoring systems, and online stores. The complexity of these systems arises from their vast size, their dynamic nature, and their structure—they comprise multiple types of entities, interacting in various ways. One of the challenges we face when using and/or building such systems is that we do not necessarily understand the underlying mechanisms that govern their behavior and, consequently, how changing one part of the system will affect other aspects of the system. This is why *causality* has become a critical area of study.

Causal models provide a powerful mechanism for reasoning over the effects of interventions. Broadly speaking, causal inference answers the question "How will changing X affect Y?" Thus, causal modeling provides actionable knowledge that can be used for decision making, policy evaluation, etc. Inferring causal dependence is strictly more difficult than inferring statistical association. To add to that, there are certain factors that make causal inference even more challenging, such as the complexity of the systems under study. In this thesis, we focus on complex systems that comprise multiple types of interacting entities (*relational* systems) that evolve over time (*temporal* systems). From a statistical point of view, the addition of temporal and relational components translates to data instances that are not independent and identically distributed (i.i.d.). For non-i.i.d. data instances, conventional statistical approaches will infer invalid and misleading models.

Another factor to consider in causal analysis is the type of available data or, more generally, the type of analysis that researchers can perform. Randomized experiments have long been the leading method for inferring causal dependence. However, in many cases, experiments are too expensive, time-consuming, or even unethical to perform and researchers only have access to observational data. Discovering causal relationships and estimating causal effects in this observational setting requires a different set of tools. For example, the framework of *do*-calculus [61] allows inferences about when a causal effect is identifiable from observational data. However, the details of *do*-calculus have only been formalized in the context of propositional models.

In this thesis, we address some of these challenges for causal modeling through the use of graphical models for causal inference in complex dynamic systems. Specifically, we focus on relational models, which model complex interactions between multiple types of entities. For the first part of this thesis, we investigate the intricacies of the semantics of relational models and the impact that different semantics have for various types of data analysis, focusing on estimation of average treatment effects. In the second part of this thesis, we introduce a graphical model representation that extends relational models to include a temporal component. We provide a characterization of d-separation properties for this class of models. Finally, we present a constraint-based algorithm to learn the structure of temporal relational graphical models from data.

These expressive temporal and relational graphical models allow us to represent more accurately, and arguably in a more intuitive way, a plethora of complex phenomena. For example, the combination of relational and temporal components allows us to represent and reason about diffusion processes on networks with multiple types of nodes and multiple types of edges. The d-separation properties of these models lay the groundwork for reasoning about the effects of interventions, for identifying causal effects, and for designing experiments in these complex types of systems. Finally, most methods for estimation of causal effects, either experimental or observational, assume that the underlying causal model of the domain is known in advance. However, in many cases, this is not true. The model is either completely unknown, or only some parts of it can be successfully extracted from domain experts. Structure learning algorithms provide an algorithmic way to learn such models from data, at the very least as a starting point for further refinement.

1.1 Causality

Causality is central to understanding the behavior of complex systems and to selecting actions that will achieve particular outcomes. Thus, causality is implicitly or explicitly central to mainstream AI areas such as planning, cognitive modeling, computational sustainability, game playing, multiagent systems, and robotics. Causal inference is also central to many areas beyond AI, including medicine, public policy, and nearly all areas of science.

Causal modeling is often divided into two tasks [77]. The first task is that of learning a causal model given a data set and, optionally, given some additional information, such as background knowledge. This is usually referred to as *causal structure learning* or *causal discovery.* The goal is to find a set of causal models that contains the true causal model that produced the given data. The second task is estimating the effects of interventions given a causal model. This task is known as *causal inference.* Notice that for the task of causal inference, the causal model of the domain is assumed to be known.

1.2 Relational and Temporal

The vast majority of work in the area of causality has focused on *propositional* domains, where data instances are considered to be independent and identically distributed (i.i.d.). However, many real-world domains are more complex, they consist of multiple types of entities that interact with each other. Such domains are referred to as *relational domains* or *networks*. Examples of relational domains include social networks (people are friends with other people), academic publishing (authors write papers, papers cite other papers), epidemiology (patients visit hospitals, patients are related to other patients), education (students go to schools, teachers teach in schools, students interact with teachers).

Relational models, contrary to propositional models, can represent dependencies between individuals. For example, on a social network, relational models can express dependencies of the form "the behavior of my friends affects my behavior" (i.e., interference between units). Moreover, relational models can capture dependencies between entities of different types. Consider for example an academic institution with students and courses. A relational model can express dependencies of the form "the GPA of the students depends on the difficulty of the courses they are taking".

Many real-world systems of interest are dynamic, i.e., they change over time. For example, a social network can change over time in multiple ways. The structure of the networks might change: new users are added, users are removed, new friendships are formed, and so on. Apart from the network structure, the attributes of the users are not static. For example, a user's behavior (such as eating habits) might change over time, and this might influence the behavior of their peers.

To accurately represent dynamic relational domains and reason over them, we need representations and models that can handle that level of complexity. Formalizing temporal relational models opens the door to approaching a variety of causal and non-causal tasks for temporal relational domains, such as inferring dependencies in temporal relational domains from data, estimating causal effects from observational and/or experimental relational time-series, designing experiments in dynamic relational domains.

It is worth noting that this additional expressivity is particularly important for causal discovery. Ultimately, the goal of causal discovery is to uncover the causal structure that underlies a system's behavior. If the true causal structure is not within the expressive power of the modeling formalism we use, then there is no way to represent and learn that structure from data. This is not necessarily true for the task of prediction, since in that case it is sufficient to find features that are highly correlated with the variable of interest.

1.3 Contributions

This thesis makes the following contributions:

- We investigate the impact of alternative grounding semantics for relational models. Specifically, we focus on relational models with a single type of entity, a single type of relationship, and with long-range relational dependencies (i.e., dependencies that are induced by the extended neighborhood of a node as opposed to its immediate neighbors). We show the impact that alternative grounding semantics have for feature construction and model fit [52].

- We investigate the impact of multiple types of relationships for causal analysis. We focus on relational models with a single type of entity and multiple types of relationships (heterogeneous networks). We show how the presence of multiple types of relationships impacts estimation of average treatment effect and model selection [53].

- We present a formalization of (discrete time) temporal relational models, an expressive class of models that can capture probabilistic dependencies between variables on different types of entities within and across time points [54].

- We extend the notion of abstract ground graphs [50]—a lifted representation that allows reasoning about the conditional independencies implied by a relational model—to the case of temporal relational models, and we show that temporal abstract ground graphs are a sound and complete abstraction for ground graphs of temporal relational models. Temporal abstract ground graphs can be used to answer d-separation queries for temporal relational models [54].

- We extend an existing constraint-based algorithm for inferring causal dependence in relational data—the relational causal discovery (RCD) algorithm—to incorporate time, thus providing a constraint-based method to learn causal models from temporal relational data. We show that the temporal relational causal discovery (TRCD) algorithm is sound and complete under certain standard assumptions [54].

1.4 Dissertation Outline

In this thesis, we first provide the necessary background on graphical models and causal discovery for propositional domains (chapter 2). In chapter 3, we describe in detail the existing representation for relational models following the approach of Maier et al. [50] and the semantics of relational models.

Next, we investigate alternative grounding semantics for relational models, i.e., alternative ways to "roll-out" dependencies from the relational to the propositional level. We show the impact that alternative grounding semantics have for various tasks of interest (feature construction, model fit, estimation of causal effects) under two different scenarios. Chapter 4 focuses on the case of networks with one type of relationship with probabilistic dependencies between a node and nodes in its extended neighborhood [52]. Chapter 5, focuses on the case of heterogeneous networks with two types of relationships and probabilistic dependencies from a node's immediate peers [53].

Chapter 6 introduces the representation for temporal relational models, temporal abstract ground graphs, and temporal relational d-separation. Finally, chapter 7 describes TRCD, a constraint-based algorithm to learn the structure of temporal relational models from data [54].

CHAPTER 2

BACKGROUND

This thesis is centered around expressive families of graphical models (i.e., temporal and relational graphical models) and their use in causal modeling. In this section, we describe how graphs and graphical models are tied to probability distributions and how they can be interpreted causally. We build on work in directed acyclic graphs and Bayesian networks, a simple and widely used framework for propositional domains. This is fundamental work that serves as the foundation for the representations and methods we develop for the more expressive relational and temporal domains.

Specifically, in this chapter, we introduce the basic notions for graphs, we describe how graphs represent probability distributions, and provide a brief overview of Markov equivalence classes. We then move on to how graphs can be interpreted causally. Finally, we describe existing methods for causal discovery from propositional data, or in other words, how to learn the structure of propositional graphical models when given a propositional data set.

2.1 Graphs and Probability Distributions

A graph G is a tuple $G = \langle V, E \rangle$, where V is the set of vertices and $E \subseteq V \times V$ is the set of edges. A graph is a *directed acyclic graph* (DAG) if it does not contain directed cycles. A graph is *partially directed* if it has both directed and undirected edges. The *skeleton* of a graph G is the undirected graph that can be obtained by substituting every edge of G with an undirected edge. A *path* between two nodes $X, Y \in V$ is a sequence of nodes $X, V_1, \ldots, V_n, Y$ such that there exists an edge between any two consecutive nodes of the path. A *directed path* between two nodes X and Y is a sequence of nodes $X \rightarrow V_1 \rightarrow \ldots \rightarrow V_n \rightarrow Y$. A *v-structure* or *collider* on a graph G is an ordered triple of nodes $\langle X, Y, Z\rangle$ such that $X \rightarrow Y \leftarrow Z$ and there is no edge between X and Z. The *parents* of a node $X \in V$ are the nodes with incoming edges to X: $Pa(X) = \{Y \in V | (Y, X) \in E\}$. A node Y is a *descendant* of X if there exists a directed path from X to Y.

Directed acyclic graphs can be used to compactly represent joint probability distributions over sets of random variables. To be more specific, the structure of the graph encodes information about the set of (marginal or conditional) independencies that hold in that joint distribution. A directed acyclic graph encodes a set of independencies according to the *local Markov condition*.

Definition 2.1 (Local Markov condition)**.** Let $G = \langle V, E\rangle$ be a directed acyclic graph. For every node in $V_i \in V$, the local Markov condition dictates that: V_i is independent of its non-descendants given its parents in the graph G.

The local Markov condition allows us to infer independencies locally, for every node conditioned on its immediate ancestors. However, the set of independencies inferred directly from the local Markov condition implies a larger set of conditional independencies. The latter can be derived through the *global Markov condition* or *d-separation* [84, 27]. The independencies implied by the local Markov condition are exactly those that can be derived using d-separation.

Intuitively, two nodes are d-separated if there are no paths of dependence (d-connecting paths) between these two nodes in the graph.

Definition 2.2 (d-connecting path)**.** Let $G = \langle V, E\rangle$ be a directed acyclic graph, $V_1, V_2 \in V$ two nodes in G, and $\mathbf{Z} \subset V$ a set of variables such that $V_1, V_2 \notin \mathbf{Z}$. A path between V_1 and V_2 is *d-connecting* given a set $\mathbf{Z}$, if every collider along the path is in $\mathbf{Z}$ or has a descendant in $\mathbf{Z}$ and no other nodes are in $\mathbf{Z}$.

Definition 2.3 (d-separation)**.** Let $G = \langle V, E \rangle$ be a directed acyclic graph, $V_1, V_2 \in V$ two nodes in G, and $\mathbf{Z} \subset V$ a set of variables such that $V_1, V_2 \notin \mathbf{Z}$. V_1 and V_2 are *d-separated* given $\mathbf{Z}$ if there are no d-connecting paths between V_1 and V_2 given $\mathbf{Z}$.

A consequence of the way graphs encode independencies is that the joint distribution of the variables factorizes according to the structure of the graph. Specifically, the joint distribution is the product of "local" distributions of a node given its parents in the graph:

$$P(V_1, \ldots, V_n) = \prod_{V_i \in V} P(V_i \mid Pa(V_i))$$

Directed acyclic graphs serve as a building component of Bayesian networks. To be more specific, Bayesian networks are directed graphical models that represent sets of probability distributions.

Definition 2.4 (Bayesian network)**.** A Bayesian network over a set of random variables V consists of

(i) a directed acyclic graph $G = \langle V, E \rangle$ (known as the structure of the model), and

(ii) a set of parameters $\mathbf{\Theta}$, where every parameter $\theta \in \mathbf{\Theta}$ is a conditional distribution of a node given its parents.

For the purposes of this thesis, we focus on the structure of graphical models and parameters will be mostly ignored.

2.2 Markov Equivalence Classes

Bayesian networks with different structures can encode the same set of independencies. Consider, for example, the case of two random variables, X and Y, that are marginally dependent, i.e., $X \not\perp\!\!\!\perp Y$. Both $X \rightarrow Y$ and $X \leftarrow Y$ are structures that are consistent with the above set of independencies (or lack thereof, to be more precise). In general, the set of graph structures that encode the same of independencies form an equivalence class, specifically, a *Markov equivalence class.*

Definition 2.5 (Markov Equivalence)**.** Two graphs are *Markov or independence equivalent* if and only if they represent the same assertions of conditional independence.

The above definition provides a conceptual description of Markov equivalence. For the case of directed acyclic graphs, there exists a precise characterization of Markov equivalence classes with respect to the structure of the graphs.

Proposition 2.6 (From [85])**.** Two DAGs are Markov equivalent if and only if they have the same skeleton and the same v-structures.

Since models that belong to the same Markov equivalence class share the same set of conditional independencies, the notion of conditional independence alone is not enough to help us distinguish among models in the same Markov equivalence class. However, there are other characteristics of probability distributions that might differ, even for two models that encode the same set of independencies. For example, Verma constraints [85, 79] and dormant independence (a subset of Verma constraints) [75]. Sapecifically for the case of two variables, there exist methods that identify the direction of dependence based on asymmetries arising from the functional form of the dependence. These include the linear non-Gaussian acyclic model (LiNGAM) [74], the nonlinear additive noise model [36], and the post nonlinear causal model [92].

2.3 Causal Semantics for Graphs

Up to this point, graphs have not been assigned a causal interpretation, they just encode probabilistic independencies. In order to interpret graphs causally, we need to make an additional set of assumptions. On a high level, these assumptions entail that the Markov condition (and equivalently d-separation) provide the correct correspondence between causal structure and probabilistic independence. A definition of causal DAGs, taken from Hernan et al. [34], is the following:

Definition 2.7 (Causal Directed Acyclic Graph)**.** A causal DAG is a DAG in which the lack on an arrow $X \rightarrow Y$ can be interpreted as the absence of a direct causal effect of X on Y, relative to the other variables of the graph. Moreover, all common causes, even if unmeasured, of any pair of variables are themselves on the graph.

The semantics of causality are tied to the notion of manipulations. If X is a cause of Y, then if an external mechanism was able to set the value of X, then that should result in changes in the distribution of Y. The notion of manipulations can be formalized through Pearl's framework of *do*-calculus [61]. Intervening on a node X and setting its value to x is defined through a mathematical operator $do(x)$. The effect of $do(x)$ on a graph G is a modified graph G_x where the parents of X have been removed and the value of X is set to x. The intuitive interpretation of that operation is that by intervening on the value of X, the probabilistic relationship between X and its parents is removed. This new graph describes the post-interventional distribution P_x. Pearl's *do*-calculus consists of three rules that allow to infer interventional quantities from observational ones. A probabilistic query is said to be *identifiable* if by successive applications of the rules of *do*-calculus all interventional quantities can be replaced by their observational counterparts.

2.4 Learning the Structure of Bayesian Networks from Data

Let $V = \{V_1, \ldots, V_d\}$ be a set of d random variables and $P(V_1, \ldots, V_d\}$ the joint distribution of these variables. Let $\mathcal{D} = \{\mathbf{x_1}, \ldots, \mathbf{x_n}\}$ be a data set consisting of n independent and identically distributed (i.i.d.) samples from the joint distribution. The task of structure learning can be formalized as learning the structure of G when given data set $\mathcal{D}$. There are three main families of algorithms that learn the structure of graphical models from data.

(i) *Constraint-based* algorithms eliminate models whose structure is not consistent with the observed conditional independencies. These algorithms operate in two phases. First, they learn an undirected graph. This is achieved through systematic application of hypothesis tests to infer whether two nodes can be rendered independent conditioned on a set of nodes. Independence of two nodes is translated to absence of an edge between these two nodes in the graph. Then, they apply a series of deterministic orientation rules to retrieve a partially directed graph that corresponds to the Markov equivalence class of the true model. Algorithms in this category include PC [76], FCI [76], IC [62], and Grow Shrink [55].

(ii) *Score-based* methods search heuristically through the space of possible directed acyclic graphs and pick the one that maximizes a scoring function. Such algorithms include greedy hill climbing search, and search using tabu lists. The output of score-based methods is usually a fully directed model. It is worth mentioning Greedy Equivalence Search [9], a score-based approach that searches over the space of equivalence classes, as opposed to the space of DAGs.

(iii) *Hybrid* algorithms combine elements from both constraint-based and score-based approaches. They first follow the constraint-based approach: through hypothesis tests they constrain the space of available models by eliminating models whose structure violates the conditional independencies inferred from data. Then, for the orientation, instead of employing the deterministic orientation rules of constraint-based algorithms, they switch to a score-based approach and search over the constrained space to find the best oriented model. Hybrid methods learn a fully oriented model. An example of a hybrid algorithm is MMHC [81].

Many of the existing algorithms for learning the structure of causal models make the following three assumptions.

1. *Causal Markov condition:* A variable is independent of its non-effects conditioned on its direct causes. This implies that the independencies that can be

read off the structure of the graph (I_G) are a subset of the independencies that hold in the underlying distribution (I_D): $I_G \subseteq I_D$.

2. *Faithfulness:* The set of independencies that hold in the distribution, can be read from the structure of the graph: $I_D \subseteq I_D$. Intuitively, the faithfulness assumption ensures that all independencies that hold in the distribution can be attributed to the structure of the causal graph and they are not coincidental (for example, because of a particular combination of parameters that neutralize each other).

3. *Causal Sufficiency:* There are no unmeasured common causes of the measured variables.

In this thesis, we focus on constraint-based algorithms. Specifically, we focus on PC [76] and its relational counterpart, RCD [49]. Under the assumptions of causal Markov condition, faithfulness, and causal sufficiency, PC and RCD have been shown to be sound and complete in the large sample limit (i.e., with perfect tests of conditional independence).

CHAPTER 3
RELATIONAL MODEL

In this chapter we describe the notions necessary to define a graphical model for relational domains. We follow the approach and notation of Maier, Marazopoulou, and Jensen [50]. In section 3.1 we introduce the basic notions of our relational representation: relational schemas, relational paths, relational variables, relational dependencies, and relational models. These are concepts defined on the relational level and, as such, they constitute templates that can be instantiated. In section 3.2 we describe in detail the instantiation and the semantics of each of these concepts. Finally, we describe the probabilistic semantics associated with relational models and their various interpretations.

3.1 Relational Concepts

The basic building block of a relational model is the *relational schema*, which specifies what types of entities and relationships exist in the domain of interest.

Definition 3.1 (Relational schema)**.** A *relational schema* is a tuple

$$\mathcal{S} = \langle \mathcal{E}, \mathcal{R}, \mathcal{A}, \mathit{card} \rangle$$

where

- $\mathcal{E}$ is a set of entity classes.
- $\mathcal{R}$ a set of relationship classes. Each $R \in \mathcal{R}$ is an abbreviation for a tuple of entities $R = \langle E_1, \ldots, E_a \rangle$, where a is the arity of relationship R.

- $\mathcal{A}$ is a set of attribute classes. Each $A \in \mathcal{A}$ is an abbreviation for a function $A : \mathcal{E} \cup \mathcal{R} \mapsto D_A$ that maps an entity class or relationship class to some appropriate domain D_A.
- The cardinality function $\mathit{card} : \mathcal{E} \times \mathcal{R} \mapsto \{\texttt{one}, \texttt{many}\}$ is a partial function that maps a pair consisting of a relationship class and an entity class that participates in that relationship to the set $\{\texttt{one}, \texttt{many}\}$.

Intuitively, the cardinality function constrains the number of times an entity instance can participate in a relationship. We use the notation $E \in R$ to denote that entity E participates in relationship R, i.e., E appears in the tuple for R. When the distinction between entity and relationship classes is not relevant, we refer to them jointly as *item* classes. We will use the notation $I \in \mathcal{E} \cup \mathcal{R}$ to refer to an item class of a given schema $\mathcal{S}$. Moreover, we use the notation $\mathcal{A}(I)$ to refer to the set of attributes associated with item class I.

A relational schema can be graphically depicted with an Entity-Relationship (ER) diagram. Entities classes are depicted as rectangles, relationship classes are represented as diamonds, and attribute classes are drawn as ovals inside the entity or relationship class they refer to. The cardinality constraints are shown with crow's feet notation. Specifically, the fact that an entity participates in a relationship as `many`, is depicted using crow's foot notation. Crow's feet stands for `many` while a line with no endpoint stands for `one`. A detailed description of ER diagrams can be found in Ramakrishnan and Gehrke [65].

As an example, consider a (simplified) university domain consisting of students that take courses. Figure 3.1 shows the ER diagram for this academic domain. The domain consists of two entity classes (*Student* and *Course*), and one relationship class (*Takes*). The entity classes *Student* and *Course* participate in the relationship *Takes*. The entity class *Student* has one attribute, *gpa*, and the entity class *Course* has one attribute, *difficulty*. In this example, students can enroll in multiple courses and a

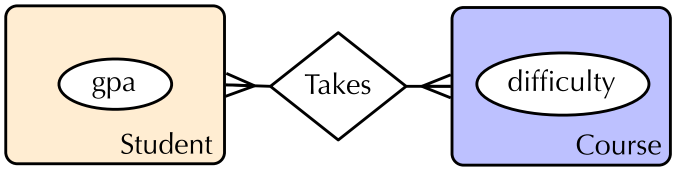


Figure 3.1: Relational schema for the example academic domain. The domain consists of two entity classes (*Student* and *Course*) and one relationship class (*Takes*). The cardinality with which *Student* and *Course* participate in the relationship *Takes* is `many`.

course can be taken by multiple students. To match our formal definitions, for this example the relational schema is $S = \langle \mathcal{E}, \mathcal{R}, \mathcal{A} \rangle$ where:

$$\begin{aligned}
\mathcal{E} &= \{\mathit{Student}, \mathit{Course}\} \\
\mathcal{R} &= \{\mathit{Takes}\} \\
\mathcal{A} &= \{\mathit{gpa}, \mathit{difficulty}\} \\
\mathit{gpa} : \mathit{Student} &\mapsto [0, 4] \\
\mathit{difficulty} : \mathit{Course} &\mapsto \{\texttt{easy}, \texttt{difficult}\} \\
\mathit{card}(\mathit{Student}, \mathit{Takes}) &= \texttt{many} \\
\mathit{card}(\mathit{Course}, \mathit{Takes}) &= \texttt{many} \\
\mathit{Takes} &= \langle \mathit{Student}, \mathit{Course} \rangle \\
\mathcal{A}(\mathit{Student}) &= \{\mathit{gpa}\} \\
\mathcal{A}(\mathit{Course}) &= \{\mathit{difficulty}\}
\end{aligned}$$

Given a relational schema, we can specify *relational paths*, which intuitively correspond to ways of traversing the schema.

Definition 3.2 (Relational path). Let $\mathcal{S} = \langle \mathcal{E}, \mathcal{R}, \mathcal{A}, \mathit{card} \rangle$ be a relational schema. A relational path $P = [I_1, \ldots, I_n]$ for $\mathcal{S}$ is an alternating sequence of entity and relationship classes such that:

- $I_1, \ldots, I_n \in \mathcal{E} \cup \mathcal{R}$.
- For every pair of consecutive item classes E, R or R, E, it holds that $E \in R$.
- For every triple of consecutive item classes of the form R, E, R, it holds that $card(E, R) = \texttt{many}$.

The first item that appears on the path is called the *base item.* Relational paths are templates and their instantiation results in a *set of items* reachable from *a given starting item* along that path (see Definition 3.7 in the next section). Maier et al. [50] have shown that the above definition of relational paths syntactically characterizes valid paths (i.e., paths that have a non-empty instantiation, as described below in Definition 3.7).

For the schema shown in Figure 3.1, example paths are the following:

$$[\mathit{Student}, \mathit{Takes}, \mathit{Course}] : \text{the set of courses a student takes}$$

$$[\mathit{Student}, \mathit{Takes}, \mathit{Course}, \mathit{Takes}, \mathit{Student}] : \text{the set of students that take the same courses with a given student}$$

Relational variables consist of a relational path and an attribute that can be reached through that path. More formally:

Definition 3.3 (Relational variable)**.** A *relational variable* is a relational path and an attribute class, $[I_1, \ldots, I_n].A$ such that $A \in \mathcal{A}(I_n)$.

For example, the relational variable $[\mathit{Student}, \mathit{Takes}, \mathit{Course}].\mathit{difficulty}$ corresponds to the set of *difficulty* attributes of the courses that a student takes. The base item for this relational variable is *Student*. Relational variables are effectively templates. The instantiation of a relational variable results in a *set of random variables* (see Definition 3.8 in the next section). Instantiating this relational variable for a given student results in a set of random variables of type *difficulty*, one for every course that the given student takes.

Probabilistic dependencies can be defined between relational variables.

Definition 3.4 (Relational dependency)**.** A *relational dependency* consists of two relational variables with a common base item, $[I_1, \ldots, I_k].A_k \to [I_1].A_1$.

The left-hand-side of a dependency is often called the *cause* or the *treatment*, and the right-hand-side is known as the *effect* or *outcome*. Dependencies are said to be in *canonical form* when the path of the effect relational variable is a single item. For canonical dependencies, the path of the cause relational variable describes how dependence is induced.

Similarly to relational variables and relational paths, relational dependencies are also templates. A single relational dependency, when instantiated, corresponds to *a set of dependencies* in the ground graph (see Definition 3.9 in the next section).

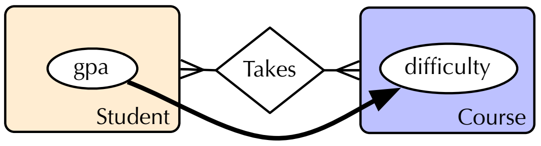
As an example, consider the following relational dependency in canonical form:

$$[\mathit{Course}, \mathit{Takes}, \mathit{Student}].\mathit{gpa} \to [\mathit{Course}].\mathit{difficulty}$$

which states that the difficulty of a course is affected by the set of GPAs of all students taking that course. Presumably, instructors adjust the difficulty of the course based on the grade-point average of enrolled students.

A *relational model* is a collection of relational dependencies defined over a single relational schema along with their parameterizations (a conditional probability distribution for each attribute given its parents).

Definition 3.5 (Relational model)**.** A *relational model* is a tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{D}, \Theta \rangle$, where $\mathcal{S}$ is a relational schema, $\mathcal{D}$ is a set of probabilistic dependencies defined over $\mathcal{S}$,



$$[Course, Takes, Student].gpa \rightarrow [Course].difficulty$$

Figure 3.2: Relational model for the example academic domain. The relational model consists of the relational schema (shown in Figure 3.1) and a set of probabilistic dependencies, depicted as directed arrows together with a label annotation that specifies how the dependence is induced. The relational dependence shown here translates to "the difficulty of a course depends on the gpa of the students that take the course".

Table 3.1: Summary of relational concepts and their corresponding instantiations.

Relational concept	Instantiation
relational schema	relational skeleton
relational path	terminal set
relational variable	relational variable instance
relational model	ground graph

and $\mathbf{\Theta}$ is a set of parameters for the probabilistic dependencies, where every parameter $\theta \in \mathbf{\Theta}$ is a conditional distribution of an attribute class given its parents in $\mathcal{D}$:

$$P\big([I].A|Pa([I].A)\big)$$

where $I \in \mathcal{E} \cup \mathcal{R}$, $A \in \mathcal{A}(I)$, and $Pa([I].A) = \{P.A'|P.A' \rightarrow [I].A \in \mathcal{D}\}$.

The relational schema together with the set of dependencies is referred to as the *structure of the relational model.* The structure of a relational model can be depicted by superimposing the dependencies on the ER diagram of the relational schema, as shown in Figure 3.2, and labeling each arrow with the corresponding relational dependency.

3.2 Instantiation of Relational Concepts

The relational concepts presented so far are defined on a first-order level, or in other words, they are templates. The next step is to specify how these templates can be instantiated. This instantiation defines the semantics assigned to the relational concepts defined earlier. Table 3.1 summarizes the relational concepts and their corresponding instantiations.

A *relational skeleton* is a partial instantiation of a relational schema. It specifies the entity and relationship instances that exist in the domain. It does not specify values for the attributes.

Definition 3.6 (Relational skeleton)**.** Let $\mathcal{S} = \langle \mathcal{E}, \mathcal{R}, \mathcal{A}, \mathit{card} \rangle$ be a relational schema. A *relational skeleton* σ for $\mathcal{S}$ consists of:

- A set of entity instances $\{e_i^1, \ldots, e_i^n\}$ for every entity class $E_i \in \mathcal{E}$.
- A set of relationship instances $\{r_i^1, \ldots, r_i^m\}$ for every relationships $R_i \in \mathcal{R}$. The set of relationships instances must be consistent with the cardinality constraints card.

Figure 3.3 shows an example relational skeleton for the relational schema of Figure 3.1. The skeleton consists of three *Student* instances—Alice, Bob, and Charlie—and two *Course* instances—CS101 and CS201. Alice and Bob are taking both classes, and Charlie is taking CS201.

Given a relational skeleton, relational paths can be instantiated, starting from a specific item and reaching a set of items along the path.

Definition 3.7 (Terminal set)**.** Let σ be a relational skeleton for relational schema $\mathcal{S}$. Let $P = [I_1 \ldots I_n]$ be a relational path for $\mathcal{S}$ and i_1 an instance of item I_1 in the skeleton σ. The *terminal set* $P|_{i_1}$ is defined inductively as:

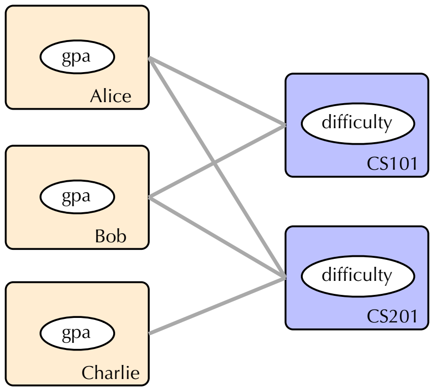


Figure 3.3: Example relational skeleton for the schema shown in Figure 3.1. This partial instantiation includes three students and two courses. The skeleton does not specify values for the attributes.

$$\begin{aligned} P^1|_{i_1} &= [I_i]|_{i_1} = \{i_1\} \\ &\vdots \\ P^n|_{i_1} &= [I_i, \ldots, I_n]|_{i_1} = \bigcup_{i_m \in P^{n-1}|_{i_1}} \Big\{ i_k | \quad \Big((i_m \in i_k \text{ if } I_k \in \mathcal{R}) \text{ or } (i_k \in i_m \text{ if } I_k \in \mathcal{E}) \Big) \\ &\qquad\qquad \text{and } i_n \notin \bigcup_{j=1}^{n-1} P^j|_{i_1} \Big\} \end{aligned}$$

The terminal set for a relational path $P = [I_1 \ldots I_n]$ is a set of instances of the item class I_n. It is important to note that this definition of terminal sets makes use of what Maier et al. [50] call *bridge burning semantics*. Intuitively speaking, terminal sets are constructed by traversing the relational skeleton beginning at a single instance of the first item in the path and reaching a set of instances of the last item in the path. However, when items are repeated along the path, ambiguities arise. Under the bridge burning semantics, no instances are revisited. In other words, as the skeleton is traversed following the relational path specification, if at any point we visit a node for the second time, then the path is considered a dead-end and not expanded any more. The term $i_n \notin \bigcup_{j=1}^{n-1} P^j|_{i_1}$ in the definition of terminal sets enforces that constraint.

The construction of terminal sets has a key role in instantiating relational models, as we explain below, as well as in the definition of relational d-separation. In the literature, relational models often gloss over the choice of relational semantics. For example, in the framework of PRMs [28], it is not specified what semantics to use when there are repeated items in a path (slot chain). On the other hand, DAPER models [33] use first-order formulas to fully specify the grounding of a model. Finally, Lee et al. [45] make use of what they call path semantics, as an alternative to bridge burning semantics.

As an example, consider the schema shown in Figure 3.1 and the skeleton shown in Figure 3.3. Below are example relational paths and their corresponding terminal sets.

$$\begin{aligned}
[\mathit{Student}]|_{\text{Bob}} &= \{\text{Bob}\} \\
[\mathit{Student}, \mathit{Takes}, \mathit{Course}]|_{\text{Alice}} &= \{\text{CS101, CS201}\} \\
[\mathit{Student}, \mathit{Takes}, \mathit{Course}]|_{\text{Charlie}} &= \{\text{CS201}\} \\
[\mathit{Student}, \mathit{Takes}, \mathit{Course}, \mathit{Takes}, \mathit{Student}]|_{\text{Charlie}} &= \{\text{Bob}\}
\end{aligned}$$

A relational variable is defined as a relational path and an attribute class of the last item that appears on the path. Instantiating the relational path results in a set of item instances in the skeleton. Instantiating the corresponding relational variable results in a set of attributes, one for every item instance in the terminal set of the relational path. In other words, instantiating a relational variable results in *a set of random variables*, the attributes corresponding to the terminal set of the path. More formally:

Definition 3.8 (Relational variable instance)**.** Let σ be a relational skeleton, i_1 an instance of item class I_1 in the skeleton σ, and $[I_1, \dots, I_k].A$ a relational variable. An instance of the relational variable, denoted as $[I_1 \ldots, I_k].A|_{i_1}$, is the set of attribute instances $\{i_k.A \mid A \in \mathcal{A}(i_k) \text{ and } i_k \in [I_1, \ldots, I_k]|_{i_1}\}$.

Given a relational model $\mathcal{M}$ and a relational skeleton σ, we can construct a *ground graph* $GG_{M\sigma}$ by applying the relational dependencies as specified in the model to the specific instances of the relational skeleton. This process is known as *grounding* or *rolling-out* the relational model.

Definition 3.9 (Ground graph)**.** Let $\mathcal{M} = \langle \mathcal{S}, \mathcal{D}, \Theta \rangle$ be a relational model and σ a relational skeleton for $\mathcal{S}$. The ground graph for model $\mathcal{M}$ and skeleton σ is a directed acyclic graph $GG_{M\sigma} = \langle V, E \rangle$ such that:

- $V = \{i.A | I \in \mathcal{E} \cup \mathcal{R} \text{ and } A \in \mathcal{A}(I)\}$
- $E = \{i_t.A_t \rightarrow i_o.A_o | [I_o, \ldots, I_t].A_t \rightarrow [I_o].A_o \in \mathcal{D} \text{ and } i_t \in [I_o, \ldots, I_t]|_{i_o}\}$

Figure 3.4 shows the ground graph for the model of Figure 3.2 applied on the relational skeleton shown in Figure 3.3. Note that every node in the ground graph corresponds to a random variable. Every $i.A$ is assigned the parameter specified for $[I].A$. This is common convention in relational learning and is known as *templating* or *parameter-tying*. The ground graph is effectively a Bayesian network. However, the size of the ground graph varies with the size of the relational skeleton (i.e., the size of the data set).

The ground graph effectively defines the semantics of a relational model, or in other words, the probability distribution it represents. If we assume a fixed relational skeleton, then a relational model defines a probability distribution over the attributes of the entities and relationships of the model.

Definition 3.10 (Semantics of relational model)**.** Let $\mathcal{S} = \langle \mathcal{E}, \mathcal{R}, \mathcal{A}, \mathit{card} \rangle$ be a relational schema, $\mathcal{M} = \langle \mathcal{S}, \mathcal{D}, \Theta \rangle$ a model defined for that schema, σ a relational skeleton for $\mathcal{S}$, and $GG_{M\sigma} = \langle V, E \rangle$ the corresponding ground graph. The relational model

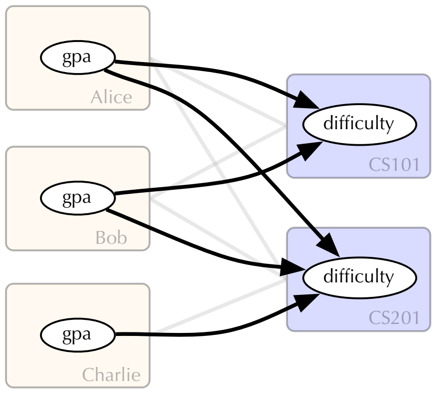


Figure 3.4: Ground graph for the model of Figure 3.1 applied on the relational skeleton shown in Figure 3.3. To aid visualization, the relational skeleton is shown in the background.

defines a probability distribution over the attributes of the entity and relationship instances in the model:

$$P(GG_{M\sigma}) = \prod_{i.A \in V} P\big(i.A | Pa_{GG_{M\sigma}}(i.A)\big)$$

where every $i.A$ is assigned the parameter specified for $[I].A$.

This is consistent with the semantics defined for probabilistic relational models (PRMs) [28].

Since the semantics of relational models are effectively reduced to the probabilistic semantics of their ground graphs, the resulting ground graphs need to define coherent probability distributions. This is guaranteed if the ground graphs are acyclic. It can be shown that if the relational model structure does not contain cycles,[1] then the resulting ground graphs will be acyclic. In this work, we restrict our attention to relational models that do not contain the kind of relational autocorrelation that gives rise to cycles in the ground graph. Moreover, we do not consider relational schemas that contain cycles through `one-one` relationships. This is not a limitation on the expressive power of our formalism, since such cycles can be represented in a single entity type.

3.3 Learning Relational Models

We now turn our attention on how to learn relational models. We focus on structure learning, i.e., how to learn the structure of relational models from data. As pointed out by Xiang et al. [90], learning in the relational setting has been treated by researchers in two distinct ways:

1. Learning from a single relational skeleton. In this case, datapoints are the nodes of the relational skeleton. Increasing the sample size corresponds to increasing the size of the relational skeleton. This paradigm is used, for example, in social network analysis where there exists a single social network (e.g., the Facebook social graph). Increasing the sample size translates to considering a larger portion of the single social graph. The work of Xiang et al. [90] provides an asymptotic analysis of consistency for relational learning methods in this case.

2. Learning from multiple relational skeletons. In this case, a datapoint corresponds to an entire relational skeleton. The population consists of independent samples of relational skeletons (the relational skeletons might be different but they need to follow the same relational schema). Increasing the sample size corresponds to increasing the number of relational skeletons. Many bio-related tasks fall into this category, such as learning protein structure and chemical compounds.

We restrict our attention to the first case (learning from a single relational skeleton). In this case, the structure learning task can be formalized as follows.

[1]The relational model structure ignoring the label specification of the dependencies is known as the *class-dependency graph*. If the class dependency graph is acyclic, then all ground graphs of that model will be acyclic [28].

Definition 3.11 (Structure learning task)**.** Assume we are given a relational schema $\mathcal{S}$, a relational skeleton σ consistent with schema $\mathcal{S}$, and a data set $\mathbb{D}$ consistent with σ. Structure learning is the task of learning the set of relational dependencies $\mathcal{D}$ for model $\mathcal{M} = \langle \mathcal{S}, \mathcal{D} \rangle$.

The data set $\mathbb{D}$ is a fully populated relational database with schema $\mathcal{S}$, for the skeleton σ. To be more precise, $\mathbb{D}$ contains a table for ever item class $I \in \mathcal{E} \cup \mathcal{R}$. A table for item I has a column for every attribute of item class I and a row for every item instance $i \in \sigma(I)$. The rows contain the values of every attribute (in the appropriate domain) for every item instance $i \in \sigma(I)$.

Most methods for learning the structure of relational models follow the score-based paradigm. For example, Bayesian model selection has been adapted for probabilistic relational models (PRMs) [28]. Formally, the task is to compute the posterior probability of a structure $\mathcal{M}$ (specifically, of the set of dependencies $\mathcal{D}$) given a data set $\mathbb{D}$. By applying Bayes rule, this can be computed as:

$$P(\mathcal{D}|\mathbb{D}, \sigma) \propto P(\mathbb{D}|\mathcal{D}, \sigma) P(\mathcal{D}|\sigma)$$

The choice of model structure (i.e., of the set of dependencies) is assumed to be independent of the choice of the skeleton: $P(\mathcal{D}|\sigma) = P(\mathcal{D})$. For Bayesian networks, it is common to impose a uniform prior over structures. However, in the relational case this is not straightforward, since there might be an infinite number of possible structures. This happens, for example, when the relational dependencies can have infinite length (friends, friends of friends, friends of friends of friends, and so on). A possible solution is to penalize long dependencies, by requiring the prior over structures $P(\mathcal{D})$ to be inversely proportional to the total length of dependencies in the model.

The term $P(\mathbb{D}|\mathcal{D}, \sigma)$ of the above equation can be further decomposed into:

$$P(\mathbb{D}|\mathcal{D}, \sigma) = \int P(\mathbb{D}|\mathcal{D}, \sigma, \theta_{\mathcal{D}}) P(\theta_{\mathcal{D}}|\mathcal{D}) d\theta_{\mathcal{D}}$$

where $P(\theta_{\mathcal{D}}|\mathcal{D})$ is the prior of parameter values for each possible structure. Again, contrary to the case of Bayesian networks, this is not straightforward to define for relational models since there are infinitely many alternative structures. Getoor et al. [28] defer this issue to future work.

Regarding constraint-based algorithms for relational models, the first such algorithm was introduced by Maier, Marazopoulou, and Jensen [49]. The RCD algorithm operates in the same way that PC does, but leverages the theory of relational d-separation introduced by Maier, Marazopoulou, and Jensen [50] to translate independencies inferred from the data to independencies that are implied by the model structure.[2] Building on the theory of relational d-separation, Lee et al. [46] provide RCD-light, a constraint-based algorithm similar to RCD that performs fewer test of conditional independence. Lee et al. [45] introduce a RpCD, an algorithm similar to RCD that learns the structure of relational causal models under path semantics, as opposed to bridge burning semantics used by RCD. Finally, Ishak et al. [38] introduce a hybrid algorithm that uses the first phase of RCD to learn unoriented dependencies and a score-based method to learn orientations.

[2]A detailed description of relational d-separation, abstract ground graphs, and RCD is given in Chapters 6 and 7, in the interest of placing this material closer to where it is used.

CHAPTER 4

ALTERNATIVE GROUNDING SEMANTICS FOR RELATIONAL MODELS

In the relational model presented so far, we made certain simplifying assumptions regarding the form of the relational dependencies. To be more specific, the relational paths were constrained to traverse the schema and bridge-burning semantics were enforced when instantiating relational paths, i.e., an individual seen along a path will not be revisited. In the general case, we can imagine alternative ways to translate a relational dependence to propositional ones, for example by not enforcing bridge-burning semantics. The goal of this chapter is to investigate alternative grounding semantics and the impact they have on feature construction and model fit.[1]

We focus on social networks, a subcase of relational models that contain a single entity-type and a single-relationship type. Social networks—for example Facebook, Twitter, and LinkedIn—have become ubiquitous. Such systems produce large amounts of data that record the attributes of individuals, but also information about the connections and interactions among these individuals. These rich data sets allow researchers to investigate various aspects of social behavior on a scale that previously was not possible. Many of these phenomena involve *social* or *peer-effects* which measure the level of influence exerted on individuals by their *neighborhood*—other individuals that are near to them in the network. Surprisingly, no clear semantics exist for defining the set of nodes that belong to that neighborhood. As we show,

[1]Marazopoulou, Arbour, Jensen. Refining the Semantics of Social Influence. NIPS Workshop (2014) [52]

ambiguity in how neighborhoods are defined can have significant impact on analyses examining these effects. Specifically, values for the aggregates of these sets can differ substantially depending on the specific semantics used, which may lead to different conclusions.

4.1 An Example

As a motivating example, consider the following question: *Does the happiness of an individual depend on the happiness of her extended social circle (friends of friends)?* On the face of it, this problem is relatively straightforward to formulate. First, we would gather data about an individual's extended social circle. We can then aggregate those values and estimate if they have an effect on her happiness[2]. However, we have failed to be explicit about a critical component: *what exactly constitutes membership in someone's extended social circle?* There are (at least) two equally reasonable options:

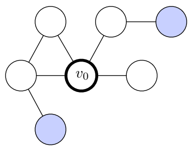
1. Only include individuals who are connected to one of her friends but not to her (friends of her friends who are not her friends).

2. Include all individuals connected to one of her friends, including those that are her friends (friends of her friends no matter whether they are her friends or not).

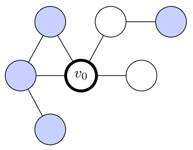
Figure 4.1 provides a visualization of these different semantics for a given node of a small network. While both options are reasonable, they lead to significantly different estimates of the values of the set, as we show in this chapter.

The effects of using different semantics are amplified when considering longer paths of dependence (e.g., friends of my friends of my friends, etc.), since longer

[2]Note that we are intentionally ambiguous regarding the exact method of estimation. The implications of this work apply to a large number of techniques both predictive and causal.



(a) Friends of my friends who are not my friends.



(b) Friends of my friends who could be my friends.

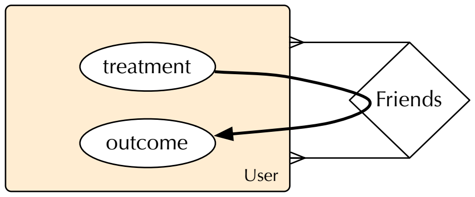
Figure 4.1: Example of alternative semantics on a small network. The node with the highlighted outline is the central node of the analysis. Shaded nodes are the ones that belong to the set of "friends of friends" for the central node under different types of semantics.

paths admit more semantics. Such longer-range dependencies are not unrealistic. In fact, according to Christakis and Fowler [12], social networks obey the "three-degrees of influence" rule. In other words, an individual can affect others that are up to three connections away (friends, friends of friends, friends of friends of friends).

In what follows, we provide a formal description of the problem and we define two different semantics. We then provide experimental evidence using the Erdös-Rényi model, Barabási model, and Watts-Strogatz model that quantifies the effect of using the different semantics on both set construction and subsequent model-selection. Finally, we examine the impact of these semantics on sixteen real-world networks from the Stanford network analysis project (SNAP) repository.

4.2 Related Work

A plethora of work exists that analyzes the properties of social networks. However, the actual construction for sets of peers is relatively understudied. A closely related work is that of Ugander et al. [83], which focuses on analyzing the structure of the Facebook graph. The authors explicitly make a distinction between the number of non-unique friends of friends (the number of paths of length two from the initial node that do not return to the initial node) and unique friends of friends (the set of nodes that can be reached from the initial node with paths of length two). That distinction



$[User, Friends, User, Friends, User].treatment \rightarrow [User].outcome$

Figure 4.2: Relational model for a social network (single entity/single relationship). The relational dependence is induced by a user's second level peers (friends of friends).

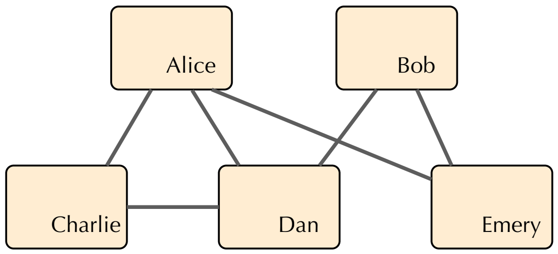


Figure 4.3: Example relational skeleton for a social network domain, i.e., for the relational model shown in Figure 4.2. The skeleton consists of a set of users and their friendships. It is effectively an undirected graph.

could be thought of as an additional type of semantics. It is worth pointing out that our work has connections to feature construction for relational learning, as well as to probabilistic relational models, such as PRMs [28] and Markov logic networks [68]. More specifically, relational models are lifted representations that can be grounded to propositional models, based on the specified grounding semantics. Those semantics specify how an edge on the relational level "translates" to an edge on the propositional level.

4.3 Problem Setup

In this chapter, we focus on relational model with one type of entity and one type of relationship and long-range relational dependencies, such as the model shown in Figure 4.2. For a schema with one entity and one relationship class, the relational skeleton is simply an undirected graph $G = \langle V, E \rangle$, as shown in Figure 4.3.

Our focus is on peer or social effects. Specifically, we focus on the effect that a node's friends of friends have on that specific node. Assume that we are given an initial vertex, $v_0 \in V$. We investigate the effect of two different semantics for constructing the set of friends of friends of v_0. Let $\mathbf{V_k}$ be the set of vertices that can be reached from v_0 with a path length of k (without repetition of edges along that path). One construction of our set is to simply consider all vertices that are found at path length 2, i.e., $\mathbf{V_2}$. Another approach is to include only the vertices whose shortest path is of length two, i.e. $\mathbf{V_2} \setminus \mathbf{V_1}$. The question that remains is what is the effective difference between employing these two strategies in terms of feature construction and model selection.

4.4 Synthetic Experiments

In this section, we use synthetically generated networks to showcase the difference between the two alternative grounding semantics. To be more specific, the first set of experiment uses synthetically generated networks to quantify the impact of the different semantics on terminal sets (i.e., feature construction). The second set of experiments uses synthetic data generated on top of the synthetic networks to measure the impact of the different semantics on model selection.

4.4.1 Effect on Terminal Sets

The simplest measure to capture the difference between these semantics is to quantify the disparity between the sets constructed using the two strategies described above. Note that we are ignoring attribute values and focusing on the simple presence/absence of a vertex in a set. In this experiment, we first generated random networks with varying size (10, 50, 100, 200, 300, 400, 500 nodes) using the following models:

1. Erdös-Rényi model [20]. Specifically, we used the $G(n, p)$ model, which generates a random graph with n nodes and any two nodes are connected with probability p. For this experiment we used $p =$0.1-0.9 incremented by intervals of 0.1.

2. Barabási-Albert model [7]. This model creates networks starting with one vertex and adding new vertices at each time step that are connected to existing vertices with probability proportional to the number of links that a vertex has (preferential attachment). For this experiment, we varied the power of the preferential attachment (0-3 by increments of 0.5).

3. Watts-Strogatz model [89]. This model starts with a regular ring lattice with n nodes, each connected to k neighbors on both side. Then every edge is "rewired" with probability p to connect to a different node (avoiding self-loops). For this experiment, we varied the size of the neighborhood (1, 5, 10) and the rewiring probability (0.1-0.9 by increments of 0.2).

We then measured the Jaccard distance between the sets constructed under the different semantics, averaged across all nodes of a network and averaged over 500 trials. The Jaccard distance between two sets A and B is defined as $J(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|}$. Intuitively, this quantifies the overlap of two sets, while accounting for the size of both.

The results for networks generated by Erdös-Rényi models are shown in Figure 4.4. As the density of the network increases (higher p), the Jaccard distance increases as well. For networks generated using the Watts-Strogatz model, the results are shown in Figure 4.5. For values of the neighborhood parameter larger than 1, the two different semantics result in sets that have a non-zero Jaccard distance. Finally, for networks generated using the Barabási model, the two semantics lead invariably to the same sets (Jaccard distance is 0). This can be explained by the lack of many triangles in

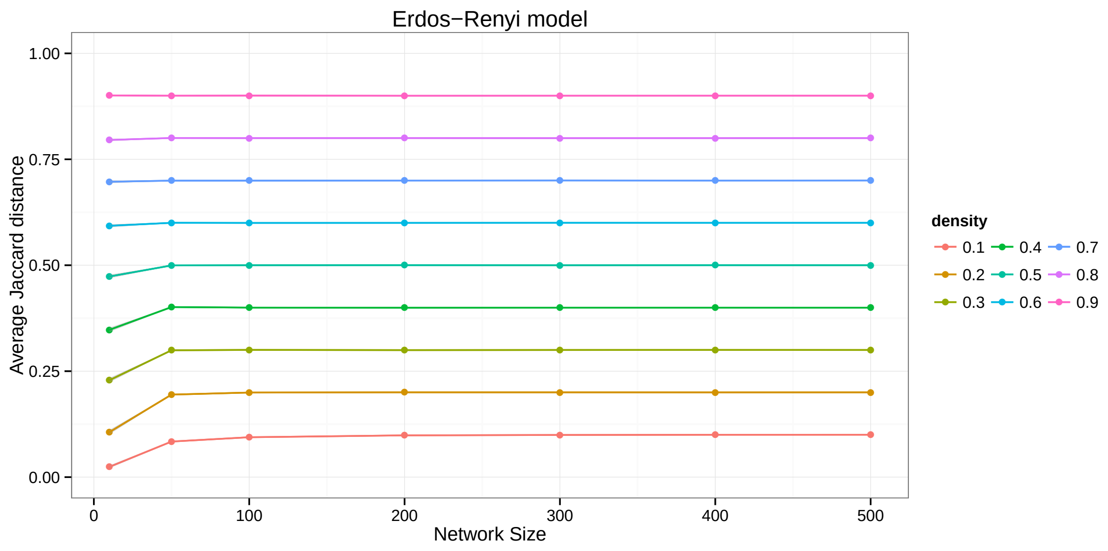


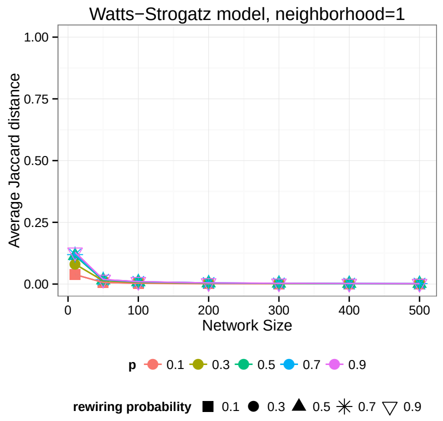
Figure 4.4: Average Jaccard distance for the two different semantics on random networks generated using the Erdös-Rényi model with varying p and network size.

graphs generated by the Barabási model. For example, consider the graph shown in Figure 4.6. For the ego node shown in yellow, vertices with shortest path of length two are shown in blue, and vertices with a path of length two are shown with highlighted border. These two sets of vertices overlap completely, therefore, their Jaccard distance is 0.

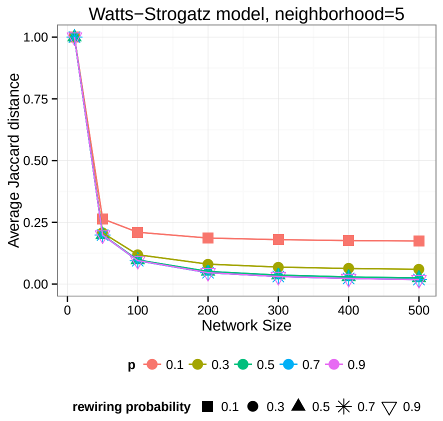
4.4.2 Effect on Model Fit

This experiment goes one step beyond measuring the distance between the sets created under different semantics. We seek to assess how the different semantics affect model selection. We consider attributed graphs, where each node has two attributes, treatment T and outcome O. We assume that the true generating model has a dependency of the form: "The treatment T of the friends of my friends affects my outcome O." We generated synthetic data sets using the following procedure.

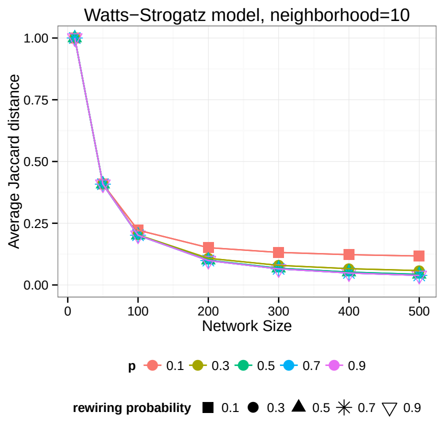
- Generate network structures using the Erdös-Rényi model and the Watts-Strogatz model. For this part, we either varied the size of the network, or the param-



(a) neighborhood=1



(b) neighborhood=5



(c) neighborhood=10

Figure 4.5: Average Jaccard distance for the two different semantics on random networks generated using the Watts-Strogatz model with varying neighborhood size, rewiring probability, and network size.

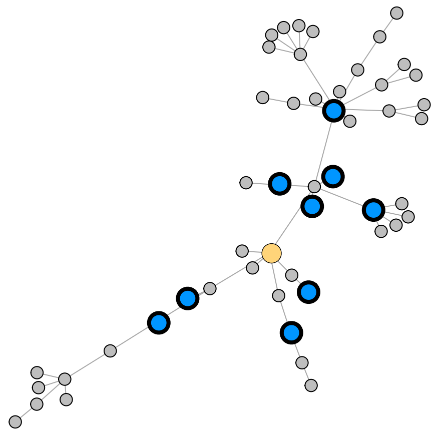


Figure 4.6: Graph with 50 nodes generated by the Barabási-Albert model with power of preferential attachment set to one. For the ego node shown in yellow, vertices with shortest path of length two are shown in blue, and vertices with a path of length two are shown with highlighted border. These two sets of vertices overlap completely, therefore, their Jaccard distance is 0.

eters of the generating model (probability p for the Erdös-Rényi model, and neighborhood and rewiring probability for the Watts-Strogatz model).

- Generate values for the attributes T by drawing from a normal distribution $T \sim \mathcal{N}(\mu, \sigma)$, where $\mu \sim \mathcal{U}(-5, 5)$ and $\sigma \sim \mathcal{U}(0, 3)$.

- Generate values for O using the conditional distribution

$$O \sim agg(\mathit{FriendsOfFriends}) + \epsilon \cdot \mathcal{N}(0, 1),$$

  where ϵ is the noise coefficient, *FriendsOfFriends* is the set of T values for the nodes that belong to the set of friends of friends (defined using two different semantics), and *agg* is an aggregation function applied to the values of that set. For this part, we ran two sets of experiments, one using *mean* as the aggregation function (i.e., the mean of the set of values in the sets *FriendsOfFriends*), and one using *degree* (i.e., the size of the set *FriendsOfFriends*).

- Learn two types of models from the generated data, one for each alternative definition of the set of friends of friends, and computed their log-likelihood.

The results for networks generated using the Erdös-Rényi model are shown in Figure 4.7 and for the Watts-Strogatz model in Figure 4.8. We use the term "strictly 2" to refer to the semantics that include only friends of friends who are not the central node's friends, i.e., node with a shortest path of length 2 from the central node. The term "2 and 1" refers to the semantics that would include friends in the set of friends of friends (nodes that have a path of length 2 from the central node, although the shortest path might be of length 1). In general, when the generating semantics match the semantics assumed for the analysis, the model has a higher log-likelihood. Moreover, when comparing the two cases where the true generating semantics do not match the applied semantics, it is generally the case that the model

Table 4.1: How different are the sets constructed using the two different semantics on some real-world network structures from the SNAP datesets.

	Experimental Results		Network statistics			
data set	mean Jaccard	max Jaccard	Nodes	Edges	Diameter	Avg. clustering coeff.
as-733	0.024	0.667	6,474	13,895	9	0.2522
ca-AstroPh	0.013	1.0	18,772	198,110	14	0.6306
ca-CondMat	0.033	1.0	23,133	93,497	14	0.6334
ca-GrQc	0.046	1.0	5,242	14,496	17	0.5296
ca-HepPh	0.018	1.0	12,008	118,521	13	0.6115
ca-HepTh	0.024	1.0	9,877	25,998	17	0.4714
com-Amazon	0.103	0.962	334,863	925,872	44	0.3967
com-DBLP	0.096	0.933	317,080	1,049,866	21	0.6324
email-Enron	0.068	1.0	36,692	183,831	11	0.4970
ego-Facebook	0.063	0.929	4,039	88,234	8	0.6055
loc-Gowalla	0.040	0.875	196,591	950,327	14	0.2367
regon-1 (010526)	0.001	0.4	11,174	23,409	9	0.2964
regon-2 (010526)	0.001	0.524	11,461	32,730	9	0.4943
roadNet-CA	0.047	1.0	1,965,206	2,766,607	849	0.0464
roadNet-PA	0.047	1.0	1,088,092	1,541,898	786	0.0465
roadNet-TX	0.047	1.0	1,379,917	1,921,660	1054	0.0470

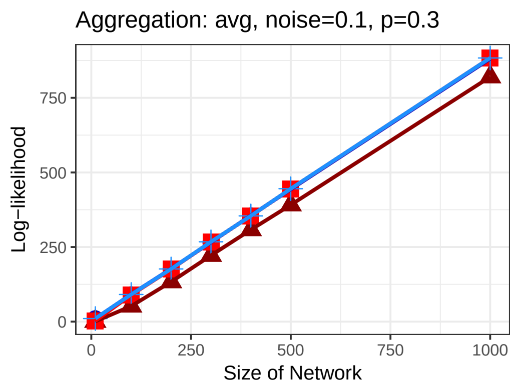
that assumed that friends of friends include friends, performs better. This is more pronounced for networks generated using the Erdös-Rényi model.

4.5 Experiments on Real Networks

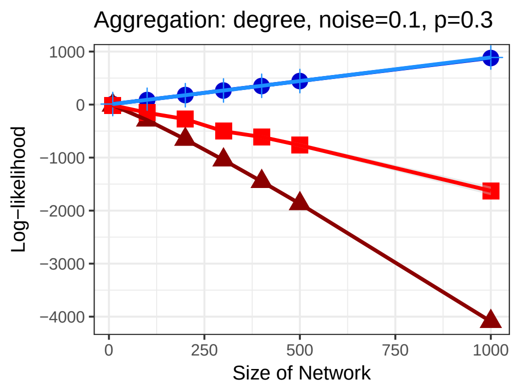
To provide a more realistic insight on the impact of using different semantics, we measured the Jaccard distance between the sets constructed under different semantics on real network structures from the SNAP data sets [47]. In this case, there are no attribute values on the nodes. The results are summarized in Table 4.1. These results indicate that there can be an average difference of up to 10% on the sets constructed under different semantics.

4.6 Concluding Remarks

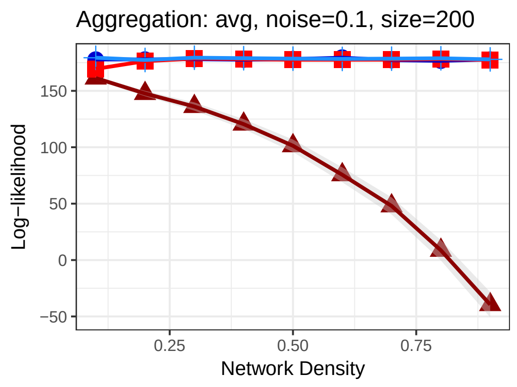
The proliferation of network data presents an opportunity to study phenomena involving network effects such as social or peer effects at a scale previously unthinkable. While there has been a large amount of work in that area, there still exists



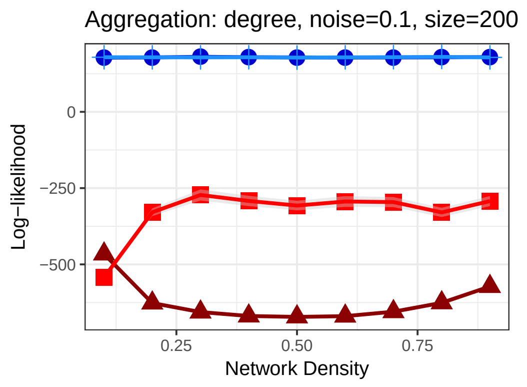
(a)



(b)



(c)



(d)

Figure 4.7: Log-likelihood of model for graphs generated using the Erdös-Rényi model. Results are shown for varying network size (4.7a, 4.7b), varying network density (4.7c, 4.7d), and different aggregation functions (average and degree). Blue corresponds to cases where the assumed semantics match the generating semantics, and red to cases where they do not.

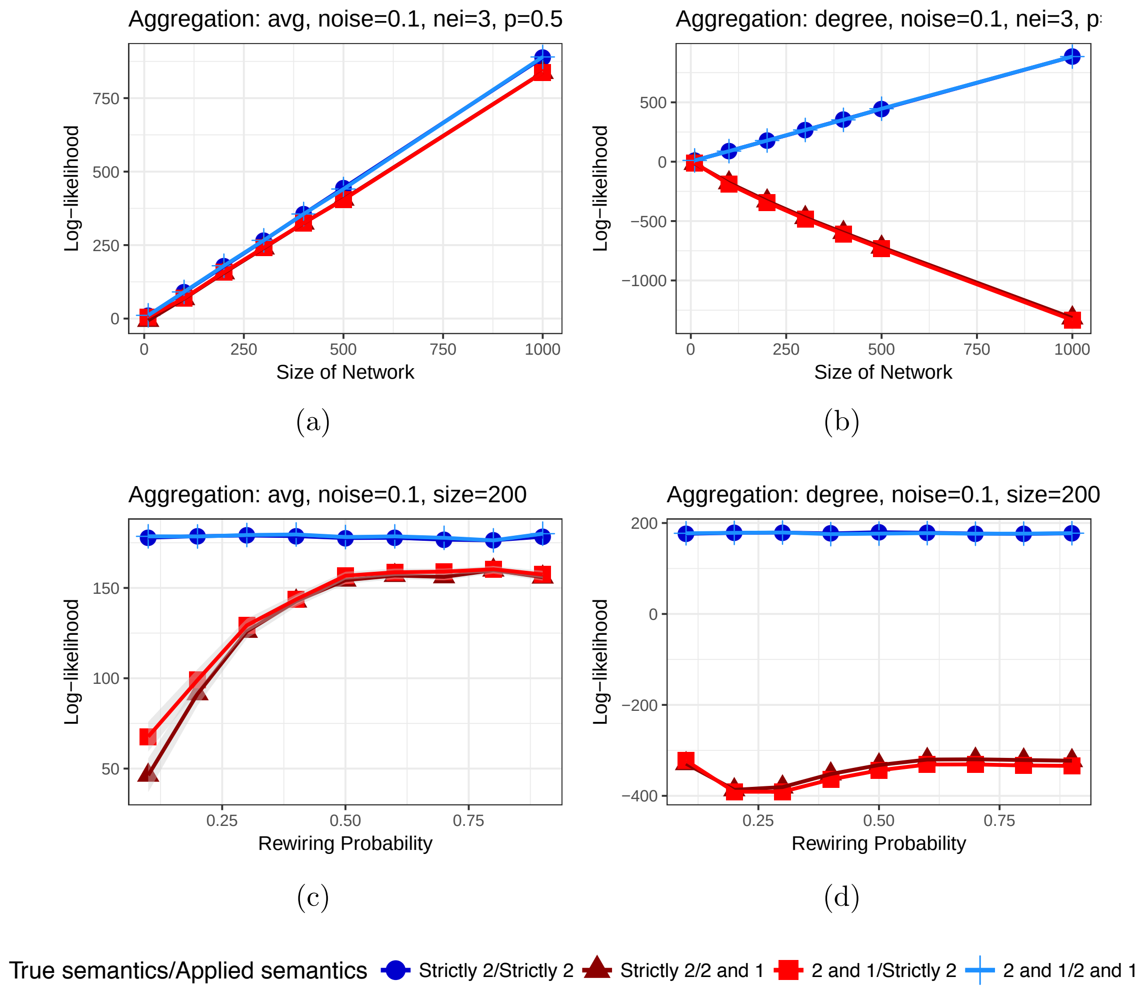


Figure 4.8: Log-likelihood of model for graphs generated using the Watts-Strogatz model. Results are shown for varying network size (4.8a, 4.8b), varying rewiring probability (4.8c, 4.8d), and different aggregation functions (average and degree). Blue corresponds to cases where the assumed semantics match the generating semantics, and red to cases where they do not.

a significant gap in terms of the exact semantics that can be used to describe the mechanism from which those phenomena arise. In this work we have looked at the semantics for constructing the set of peers in order to study a social or peer effect. While there is no definitive correct approach to construct these sets, the choice of strategy reflects assumptions about the underlying mechanism of influence. As both our synthetic and real-world experiments show, the difference is not merely theoretical; the use of one strategy over another can lead to significant differences in the aggregate values of network-based covariates. These results, when coupled with the importance of small-effect sizes in studies of large networks, demonstrate the necessity for clarity. Our work here represents a small fraction of the ambiguities that exist within the process of understanding phenomena in networks.

CHAPTER 5
CAUSAL ANALYSIS IN HETEROGENEOUS NETWORKS

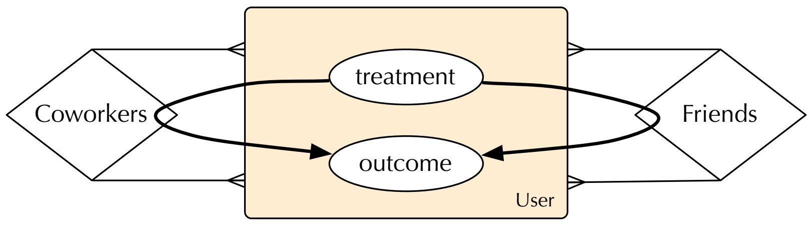
In the previous chapter, we studied the impact of alternative semantics for social networks with long-range dependencies. We focused on the tasks of feature construction and model selection. In this chapter, we adopt a causal oriented perspective. Specifically, we aim to investigate the impact of alternative grounding semantics to the estimation of causal effects for the case of social networks with multiple types of relationships.[1]

5.1 Motivation

Understanding how the behavior of an individual can be affected by the behavior of his or her peers in a network can provide valuable insight into many observed modern phenomena. For example, recent studies have focused on whether an individual's likelihood of adopting a particular product is influenced by messages from peers [1] and whether an individual's behavior can be influenced by their knowledge of their peers' behavior [4]. These studies attempt to establish the causes and effects of such behaviors within social networks, in contrast to studies that only examine correlations among such behaviors (e.g., [13, 21, 16]).

Increasing attention has been paid to developing reliable methods for causal inference in relational domains (c.f. [82], [80], [2]). While this work represents substantial

[1]Marazopoulou, Arbour, Jensen. On Causal Analysis for Heterogeneous Networks. KDD Workshops (2017) and in submission (2017) [53]



$$[User, Friends, User].treatment \rightarrow [User].outcome$$
$$[User, Coworkers, User].treatment \rightarrow [User].outcome$$

Figure 5.1: Relational model for a heterogeneous social network. Users are related to each other through types of relationships: friends and coworkers. There are two relational dependencies, one induced through a user's friends, and one through a user's coworkers.

progress, it overwhelmingly focuses on networks consisting of only one type of relationship among members of the network. However, social networks often arise through multiple types of relationships, such as friendship, kinship, and professional ties. Networks with multiple types of relationships are called *multi-relational* or *heterogeneous* networks.

As an example, consider a social network where people are related to other people through two different types of relationships: friend and coworker relationships.[2] Such a social network is shown in Figure 5.2. Every person has a set of friends and a set of coworkers, and these two sets are not necessarily distinct. In Figure 5.2, Bob and Ann are simultaneously friends and coworkers. Now consider the case where each type of relationship exerts a different type of influence on an individual, e.g., a person's political opinions might be more affected by friends than by coworkers. Naively assuming a single relationship type in this example would bias the estimated effect either upwards or downwards depending on whether there are more friend or coworker links present in the network.

[2]For clarity of exposition, we restrict our attention to two types of relationships, but the results generalize to arbitrary numbers of relationships.

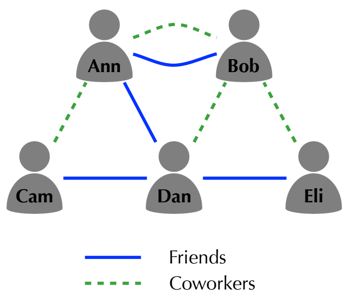


Figure 5.2: Example social network with two types of relationships between people. Solid blue lines denote friendship and dashed green lines professional connections.

Despite the possible impact of incorrectly specifying relational structure, causal inference in multi-relational domains has been largely neglected in the literature. Existing work is either predictive rather than causal (e.g., [28], [33], [68]), or focuses on asymptotic proofs rather than finite sample analysis (e.g., [50], [54]).

In this chapter, we study the impact of heterogeneous network structure on causal analysis. Toward this end, we provide:

1. An extension of existing theory for causal analysis of social networks to the case of heterogeneous networks.
2. An empirical characterization of how mis-specifying network structure in heterogeneous networks affects causal analysis.
3. An empirical quantification of how mis-specifying the relative importance of heterogeneous relationships affects causal estimation.

5.2 Related Work

Work relevant to our investigation of methods for inferring causal effects in heterogeneous networks falls into two basic categories. The first category develops novel algorithms for estimating causal effects in relational data. Ugander et al. [82] propose an experimental design for estimating the "global treatment" counterfactual by performing randomization with respect to an inferred graph clustering. Gui et al. [31] further this model by replacing the estimator of Ugander et al. with a linear model, achieving superior performance with respect to bias of the inferred average treatment effect. Toulis et al. [80] propose an experimental procedure for estimating the peer effect, i.e., the effect of only treating a peer group. Choi [11] analyzes the problem of estimating effects on networks experimentally under the assumption of monotonic effects. Arbour et al. [2] examine the problem of inferring causal effects from purely observational relational data. The work presented in this chapter may be seen as a complement to these papers, which explicitly consider single relation network settings.

The second category examines causality in domains with multiple types of entities and relationships. Arbour et al. [3] study the problem of inferring causal direction from observational multi-relational data. Maier et al. [50] extend the rules of *d*-separation to the relational setting. Maier et al. [49] leverage these new rules to present a constraint-based structure learning algorithm for relational models. Lee et al. [45] study causal inference using a different specification for grounding semantics. However, the three latter all assume a specific notion of grounding semantics, i.e., of how peers-sets are constructed.

5.3 Background

Several central concepts of causal inference are used throughout the remainder of the chapter. For clarity, we first introduce these concepts for the non-network or *propositional* setting, where instances are independent and identically distributed (i.i.d.), and then describe the extension to relational data.

5.3.1 Causal Effect Estimation

For estimation of causal effects, we use the framework of potential outcomes [72]. Consider a population of n individuals. Each individual i is characterized by the values of two random variables: a binary treatment T_i and an outcome of interest O_i. The task at hand is to estimate the causal effect of a specific treatment value on the outcome. We use the notation $O_i(T_i = 0)$ to denote the outcome of individual i that would be observed if no treatment was applied to that individual and similarly for $O_i(T_i = 1)$. More generally, $O_i(\cdot)$ is known as the *potential outcome function* or *response function*.

In the framework of potential outcomes, the causal effect of applying treatment to individual i can be formalized as a comparison between $O_i(T_i = 1)$ and $O_i(T_i = 0)$. A quantity of interest is the average treatment effect (ATE) across the entire population:

$$\tau = \frac{1}{n} \sum_{i=1}^{i=n} E[O_i(T_i = 1) - O_i(T_i = 0)].$$

In practice, this quantity cannot be computed exactly, since an individual will either receive treatment or not. However, a variety of techniques can be used to estimate ATE under the stable unit treatment value assumption (SUTVA). Under SUTVA an individual's outcome cannot depend on the treatment assignment of other individuals and there are no hidden treatment levels that can lead to different potential outcomes [37]. These techniques include methods that are appropriate for experimental data and methods appropriate for observational data (e.g., matching [78] and propensity scores [71]).

5.3.2 Causal Effect Estimation in Networks

We consider the more complex (and realistic) case where individuals in the population are not i.i.d. but instead form a social network in which nodes are influenced by their peers. Let $G = (V, E)$ be an undirected graph consisting of a set of n vertices V and a set of edges $E \subseteq V \times V$. Every node i corresponds to an individual and is characterized by two attributes, a binary treatment T_i and an outcome O_i. Let $\mathbf{T} \in \{0, 1\}^n$ denote the treatment assignment vector, i.e., a vector of length n where the i-th element corresponds to the treatment of node i, T_i. Let $O_i(\mathbf{T} = \mathbf{t})$ be the outcome of node i when the treatment assignment over the entire network is $\mathbf{T} = \mathbf{t}$. In this case, the outcome value for individual i depends on the treatment assignment over the entire population and not only on T_i. This is a violation of SUTVA. It is worth noting that there are other ways in which SUTVA might be violated, as, for example, in the case of *outcome interference*—the outcome of an individual's peers affecting her outcome—and *treatment contagion*—the treatment status of an individual's peers affecting her treatment status. While we focus on treatment interference theoretically, we provide an experimental evaluation on both treatment and outcome interference.[3]

Under our assumed setting,[4] our quantity of interest is the average treatment effect between global treatment (every node in the network is exposed to treatment) and global control (none of the nodes in the network are exposed to treatment), i.e.,

$$\tau(\mathbf{1}, \mathbf{0}) = \frac{1}{n} \sum_{i=1}^{n} E[O_i(\mathbf{T} = \mathbf{1}) - O_i(\mathbf{T} = \mathbf{0})].$$

Clearly, it is impossible to compute this quantity exactly, since for a given experiment on a network a node will be assigned either to the treatment or to the control group, but not to both.

In absence of exact estimation procedures, alternative approximate methods are necessary in order to approximate the global effect counterfactual. Such estimation procedures for experimental causal inference can be broken down to three components:

1. *Treatment assignment*—This consists of assigning individuals to the treated or the control group. For observational studies, there is an external mechanism that assigns the treatment status of individuals. For experimental studies, this is

[3]Treatment contagion is precluded by definition in the experimental setting.

[4]This is also the setting assumed by both Ugander, et al. [82], and Gui, et al. [31].

part of the experimental design. In the experimental setting, current state of the art for global treatment estimation in networks performs a graph clustering and then assigns treatment randomly with respect to each cluster of individuals [82, 31].

2. *Exposure model*—This determines when an individual is considered to be treated or not. For example, Ugander et al. [82] consider local exposure models, such as the full neighborhood exposure, where an individual is considered to be treated if all of her peers are treated. An alternative approach [31, 2] is to consider the fraction of treated peers in model estimation explicitly, leaving global effect estimate as an extrapolation performed in analysis.

3. *Analysis*—This part specifies how to estimate the causal quantity of interest, in this case the ATE. Possible approaches include inverse probability weighting [82], linear regression adjustment [31], and a general adjustment with non-parametric estimators [2].

In this work, we extend the fraction neighborhood exposure model with analysis using the linear regression adjustment proposed by Gui et al. [31] to the case of heterogeneous networks. The choice of this model is due to its flexibility and simple interpretation of coefficients.

In the fraction neighborhood exposure model, the response function depends on the individual's own treatment assignment and on the proportion of her peers that have been treated: $g(T_i, \lambda_i)$, where λ_i is the proportion of treated neighbors for node i. Consider the linear *response function*

$$g(T_i, \lambda_i) = \alpha + \beta T_i + \gamma \lambda_i$$

where β is the effect of the node's own treatment and γ is the effect from the node's peers. The ATE can be expressed as

$$\begin{aligned}\tau(\mathbf{1},\mathbf{0}) &= g(T_i = 1, \lambda_i = 1) - g(T_i = 0, \lambda_i = 0) \\ &= \beta + \gamma.\end{aligned}$$

The parameters β and γ can be estimated from data.

5.4 Heterogeneous networks

While existing frameworks provide a powerful mechanism for causal inference in single-relation networks, many networks observed in the real world contain rich relational structure that cannot be adequately described with a single type of relationship. Currently, these distinctions are largely ignored for causal inference and all edges are assumed to represent homogeneous relationships. As we show later, this can have a significant impact on the accuracy of estimates.

Consider a network with two distinct types of relationships (for convenience, called *friends* and *coworkers*). Such a network can be formalized as an undirected multigraph $G = \langle V, E \rangle$ in which the set of edges is partitioned into two disjoint sets E_F (friends) and E_C (coworkers):

$$E = E_F \cup E_C \text{ and } E_F \cap E_C = \emptyset.$$

In this network, every person has three non-overlapping (possibly empty) sets of peers: people that are only her friends, people that are only her coworkers, and people that are both. These disjoint sets of peers play a central role in our analysis for heterogeneous networks. Specifically, these sets are used to define alternative dependence models (i.e., ways in which a node is influenced by its peers), alternative exposure models, response functions and corresponding estimators for the ATE.

In the more general case of heterogeneous networks with k types of relationships, there are

$$\binom{k}{1} + \binom{k}{2} + \ldots + \binom{k}{k} = 2^k - 1$$

non-overlapping sets of peers. We will use $\mathcal{S}$ to denote the set of disjoint subsets that can be defined by k overlapping sets. Again, combining these sets in different ways leads to alternative definitions of models and estimators. Clearly, more types of relationships lead to a bigger space of potential models and corresponding estimators.

5.4.1 Estimation of ATE in Heterogeneous Networks

In heterogeneous networks, the response function might depend on multiple types of relationships. Consider, for example, the following response function for the case of a heterogeneous network with two types of relationships:

$$g_{f,c}(T_i, \lambda_i) = \alpha + \beta T_i + \gamma^f \lambda_i^f + \gamma^c \lambda_i^c$$

where λ_i^f is the proportion of treated friends for unit i and λ_i^c is the proportion of treated coworkers for unit i. In the general case of heterogeneous networks with k relationships, the response function might contain terms for every non-overlapping peer-set we can construct. In what follows, we present how to compute ATE for the fraction neighborhood exposure model for a heterogeneous network with k relationships.

Proposition 5.1. The ATE for the fraction neighborhood exposure model with response function $g(T_i, \lambda_i) = \alpha + \beta T_i + \sum_{j=1}^{|\mathcal{S}|} \gamma^j \lambda_i^j$ is given by

$$\tau_g = \beta + \sum_{j=1}^{|\mathcal{S}|} \gamma^j$$

where $\mathcal{S}$ is the set of disjoint subsets that can be defined by k overlapping sets, $j \in \{1, \dots, |\mathcal{S}|\}$ is the set of peers, and λ_i^j is the proportion of treated neighbors of the j-th set of peers for unit i.

Proof. The ATE is computed as:

$$\begin{aligned}
\tau_g &= \frac{1}{n}\sum_{i=1}^{n}(g(1,1) - g(0,0)) \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(\alpha + \beta + \sum_{j=1}^{|\mathcal{S}|}\gamma^j \lambda_i^j - \alpha\right) \\
&= \beta + \sum_{j=1}^{|\mathcal{S}|}\gamma^j
\end{aligned}$$

$\square$

5.4.2 Selecting a Response Function

As we have seen, more types of relationships result in a vast increase of the potential response functions that can be formulated. The question remains: Given a set of candidate relational semantics, is it possible to identify the true response function? We treat this as a model selection problem, and use BIC to evaluate alternative models. The BIC for a given model is defined as: ,

$$BIC = -2 \ln \hat{L} + k \ln(n),$$

where $\hat{L}$ is the maximum likelihood estimate of the likelihood function of the model, k is the number of free parameters of the model, and n is the number of data points. Intuitively, BIC captures how well the model fits the data, penalizing for the complexity of the model. Among a set of models, the one with the lower BIC is the preferred model.

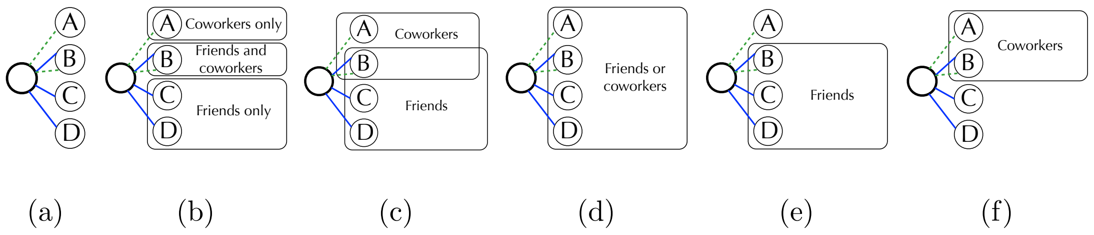


Figure 5.3: The peer-sets of interest we consider in this thesis. This figure depicts a small fragment of a social network, where the ego node (bold border) connects to four peers through two different types of relationships (friends in blue solid line, coworkers in green dashed line). The various sub-figures illustrate the five alternative peer-sets of interest we consider in this thesis.

5.4.3 Peer-sets of Interest in the Presence of Two Relationships

As we have seen, for a heterogeneous network with two relationships, there are three non-overlapping sets of peers for every node. In this chapter, we investigate the interplay of these disjoint peer-sets. and we focus on five alternative ways to combine information from these three sets. A schematic representation of these combinations is shown in Figure 5.3. For a more formal description of peer-sets of interest, let A denote the adjacency matrix of graph G, F the adjacency matrix corresponding to the friend-subgraph of G, and C the adjacency matrix of the coworker-subgraph of G. The five peer-sets of interest we are considering, are the following:

1. **Disjoint:** A node has three types of peers, those that can be reached only through edges of type E_F (i.e., peers that are only friends but not coworkers), those that can be reached only through edges of type E_C (i.e., peers that are only coworkers but not friends), and those that can be reached through both types of relationships (peers that are friends and coworkers at the same time), as shown in Figure 5.3b. For each of this peer-sets, we can define a subgraph of G with appropriate adjacency matrices.

$$F_o[i,j] = 1 \text{ iff } (V_i, V_j) \in E_F \text{ and } (V_i, V_j) \notin E_C$$

$$C_o[i,j] = 1 \text{ iff } (V_i, V_j) \in E_C \text{ and } (V_i, V_j) \notin E_F$$

$$F_\cap C[i,j] = 1 \text{ iff } (V_i, V_j) \in E_F \text{ and } (V_i, V_j) \in E_C$$

2. **Friends-coworkers:** A node has two types of peers, those that can be reached through edges of type E_F and those that can be reached through edges of type E_C (see Figure 5.3c).

$$F[i,j] = 1 \text{ iff } (V_i, V_j) \in E_F$$

$$C[i,j] = 1 \text{ iff } (V_i, V_j) \in E_C$$

3. **Friends or coworkers:** The peers of a node are all nodes it connects to either through edges of type E_F or of type E_C (see Figure 5.3d).

$$F_\cup C[i,j] = 1 \text{ iff } (V_i, V_j) \in E_F \text{ or } (V_i, V_j) \in E_C$$

   This corresponds to the case where the different types of edges are ignored, and the network is treated as a conventional (not a heterogeneous) network. In other words, the type of relationships is ignored and all peers affect the node in the same way.

4. **Friends:** A node is only influenced by peers along the friend-relationship (Figure 5.3e).

$$F[i,j] = 1 \text{ iff } (V_i, V_j) \in E_F$$

5. **Coworkers:** A node is only influenced by peers along the coworkers relationship (Figure 5.3f).

$$C[i, j] = 1 \text{ iff } (V_i, V_j) \in E_C$$

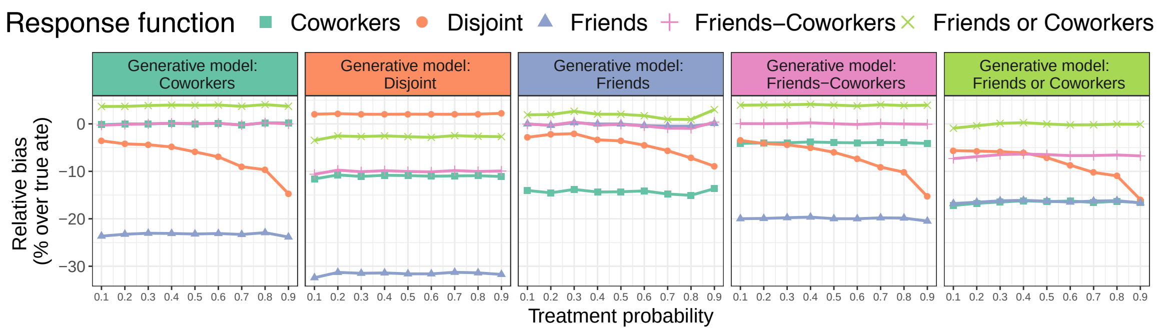
These alternative ways to construct sets of peers are not completely independent. An analyst could choose to use the more fine-grained approach (the one depicted in Figure 5.3b) and parameterize it appropriately to simulate other semantics. For example, by assigning 0 weight to the set "coworkers only" and equal weights to "friends and coworkers" and "friends only", we can effectively get the semantics depicted in Figure 5.3e. However, the analyst might not think or may not have the necessary information to do so.

Focusing on the fraction neighborhood exposure model, potential response functions that can be formulated with these peer-sets of interest are the following:

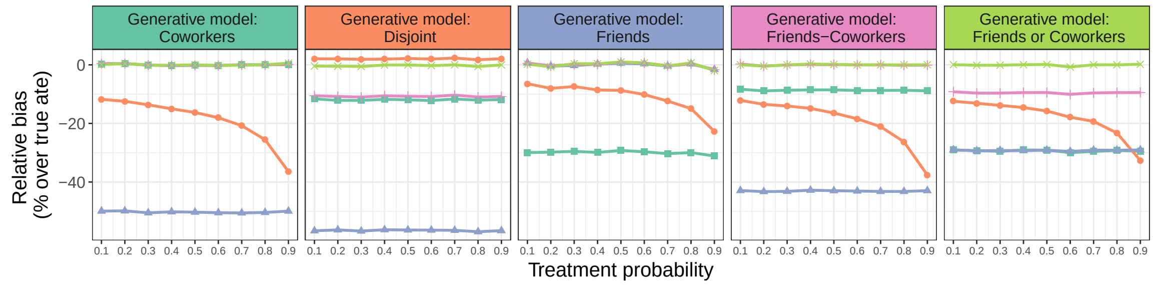
- Proportion of treated friends only, proportion of treated coworkers only, proportion of treated people that are both friends and coworkers. (Case 1)
- Proportion of treated friends and proportion of treated coworkers. (Case 2)
- Proportion of treated peers, regardless of their type. (Case 3)
- Proportion of treated friends. (Case 4)
- Proportion of treated coworkers. (Case 5)

5.5 Experiments on ATE Estimation

The first set of experiments we present focuses on estimation of ATE in heterogeneous networks using the fraction neighborhood exposure model as adapted for heterogeneous domains.



(a) Treatment assignment based on clustering on the entire graph (ignoring types of edges).



(b) Independent treatment assignment.

Figure 5.4: Relative bias of ATE estimator across 5 generative models (columns), using 5 different response functions and for increasing treatment probability. These results are on small-world networks with 1000 nodes. The noise coefficient was set to 0.5. The coefficients of the various relationships are according to configuration C2 of table 5.1. The dependence model is with peer treatment interference.

5.5.1 Generation of Synthetic Data

Experiments are performed on synthetically generated networks and synthetically generated data. In this section, we provide a detailed description of the data generation process we used.

5.5.1.1 Graph Generation

We generated two network structures with 1000 nodes, one for friends and one for coworkers, using the following network models:

- *Erdös-Rényi* [20]: Any two nodes are connected according to a given probability, p. Here we use $p = 0.2$.
- *Watts-Strogatz* [89]: We use a fixed size neighborhood $k = 3$ and rewiring probability $p = 0.2$.
- *Stochastic block* [35]: We use two communities with one of them including 200 and the other 800 nodes. The inter-community probability is set to 0.001 and the intra-community to 0.1 for the first block and to 0.05 for the second block.

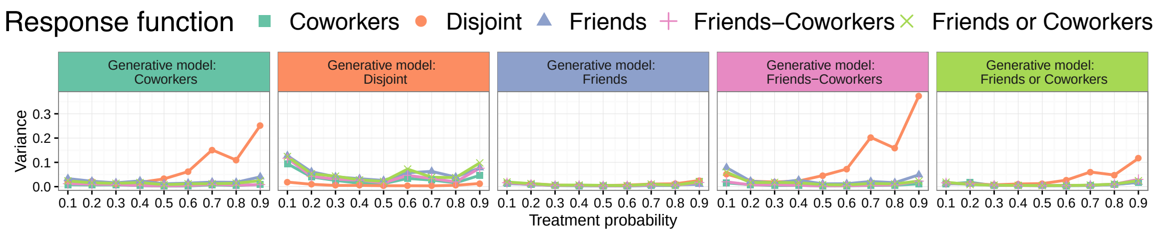
5.5.1.2 Generation of Treatment Values

To generate values for the treatment variable we used two approaches. First we generated values for every node independently (ignoring the underlying network structure) with probability p_t. The second approach is graph-cluster randomization [82]. In the presence of multiple types of edges clustering is not as straightforward. We considered three ways to create clusters (note that in each of these ways we clustered on a single-relationship network):

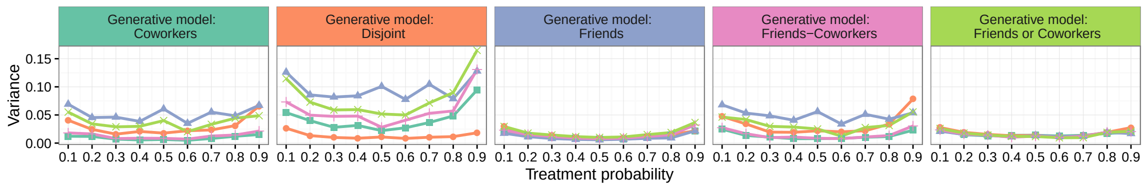
(i) considering only the friend relationship,

(ii) considering only the coworker relationship,

(iii) ignoring the relationship type.



(a) Treatment assignment based on clustering on the entire graph (ignoring types of edges).



(b) Independent treatment assignment.

Figure 5.5: Variance of ATE estimator across 5 generative models (columns), using 5 different response functions and for increasing treatment probability. These results are on small-world networks with 1000 nodes. The noise coefficient was set to 0.5. The dependence model is with peer treatment interference.

We then assigned every node in the same cluster to the treated condition with probability p_t or to control with probability $1 - p_t$.

5.5.1.3 Generation of Outcome Values

For this experiment, we considered two forms of dependence. First, we study the case where the outcome of an individual is influenced by their own treatment and the outcome of their peers:

$$O_{i,t+1} \sim w_0 + w_1 T_i + f(O_{peers\_of\_i,t}) + \epsilon$$

where f is a linear function of outcomes of peer-sets and the noise term is equal to $\epsilon = \beta_\epsilon \mathcal{N}(0, 1)$. The above defines a form of diffusion process, where the treatment of a node affects its own outcome in the first timestep and then the value of its neighbors and so on. This iterative process was repeated for 2 timesteps. This is similar to the data generation of Gui et al. [31], adapted for the case of heterogeneous networks. The adaptation has to do with the peer-sets we consider. To be more specific, we consider five different generative models, that match the peer-sets of interest we have described. In what follows D_i^j is the degree of node i with respect to relationships j.

1. **Disjoint**:

$$f(O_{peers\_of\_i,t}) = w_2^{fo} \frac{F_o[\cdot, i]^\top \mathbf{O_t}}{D_i^{F_o}} + w_2^{co} \frac{C_o[\cdot, i]^\top \mathbf{O_t}}{D_i^{C_o}} + w_2^{f\cap c} \frac{F_\cap C[\cdot, i]^\top \mathbf{O_t}}{D_i^{F_\cap C}}$$

2. **Friends-Coworkers**:

$$f(O_{peers\_of\_i,t}) = w_2^{f'} \frac{F[\cdot, i]^\top \mathbf{O_t}}{D_i^{F}} + w_2^{c'} \frac{C[\cdot, i]^\top \mathbf{O_t}}{D_i^{C}}$$

3. **Friends or Coworkers**:

$$f(O_{peers\_of\_i,t}) = w_2^{f\cup c} \frac{F_\cup C[\cdot, i]^\top \mathbf{O_t}}{D_i^{F_\cup C}}$$

4. **Friends**:

$$f(O_{peers\_of\_i,t}) = w_2^{f} \frac{F[\cdot, i]^\top \mathbf{O_t}}{D_i^{F}}$$

5. **Coworkers**:

$$f(O_{peers\_of\_i,t}) = w_2^{c} \frac{C[\cdot, i]^\top \mathbf{O_t}}{D_i^{C}}$$

Regarding the weights, we used four different configurations to vary the relative importance of each relationship. These are summarized in Table 5.1.

Table 5.1: Configurations of coefficients in the generating models used for the experiments. $w_0 = -1.5$ and $w_1 = 1$.

Setting	w_2^f	w_2^c	w_2^{fo}	w_2^{co}	$w_2^{f\cap c}$	$w_2^{f'}$	$w_2^{c'}$	$w_2^{f\cup c}$
C0	1	1	1	1	1	1	1	1
C1	1	2	1	2	5	1	2	1
C2	1	5	1	5	2	1	5	3
C3	5	1	5	1	2	5	1	3

The second form of dependence we consider is when the outcome of an individual is affected by her own treatment and the treatment of her peers:

$$O_i \sim w_0 + w_1 T_i + f(T_{peers\_of\_i}) + \epsilon$$

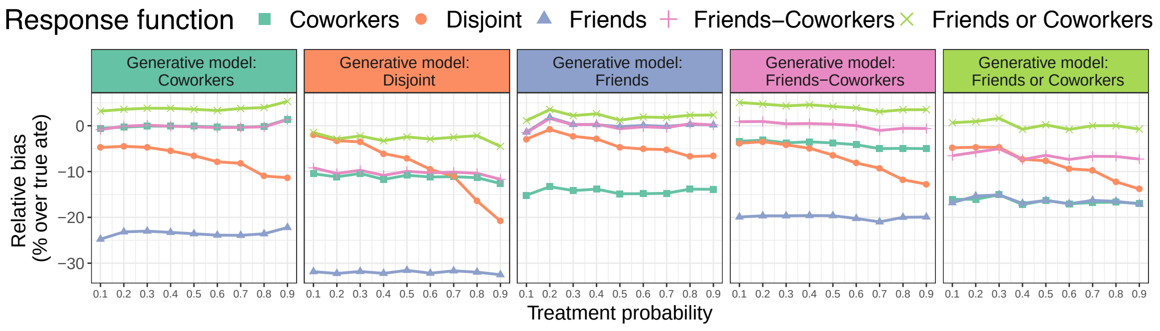
where f is a linear function of treatments of peer-sets. For the peer-sets we use the five alternative definitions, as in the previous dependence model, although in this case a diffusion process is not necessary.

5.5.2 Experimental Setup

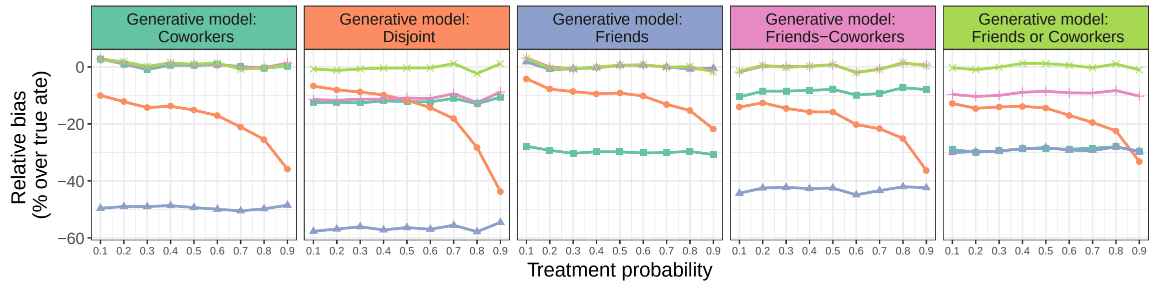
For the synthetic experiments, we randomly generated 100 network structures for friends and coworkers with 1000 nodes for each network type. We varied the treatment probability in the interval 0.1 to 0.9 by increments on 0.1, both for independent treatment assignment and clustered treatment assignment. We repeated this process for the two dependence models (outcome interference from peers and treatment interference from peers). For every combination of generating model and assumed model we estimated the ATE. We estimated the true ATE by performing a simulation where all nodes were treated and then all nodes were assigned to the control group.

5.5.3 Results

Figure 5.4 shows the relative bias for the dependence with peer treatment interference and for coefficient configuration C2. As expected, when the response function



(a) Treatment assignment based on clustering on the entire graph (ignoring types of edges).



(b) Independent treatment assignment.

Figure 5.6: Relative bias of ATE estimator across 5 generative models (columns), using 5 different exposure models and for increasing treatment probability. These results are on small-world networks with 1000 nodes. The noise coefficient was set to 0.5. The coefficients of the various relationships are according to configuration C2 of table 5.1. The dependence model is with peer outcome interference.

matches the generating model the bias is lower. The corresponding variance of the estimator is shown in Figure 5.5. As expected, in most cases the least amount of variance is observed when half the units are assigned to control (treatment probability equal to 0.5). Moreover, the variance decreases when treatment is assigned in clusters as opposed to independently, consistent with previous work [82]. Consistent across all of these results is the significant bias that can occur from mis-specification of the relational structure of dependence.

Figure 5.6 shows the bias for a similar configuration but with peer outcome interference. Again, the least amount of bias occurs in most cases when the assumed response function matches the generative model.

5.6 Experiments for Model Selection

Section 5.5 addressed the intricacies of effect estimation for causal inference in heterogeneous networks. The results indicate that when the true generating model does not match the assumed model, the ATE estimator exhibits bias. However, the question remains: given a set of candidate relational semantics, is it possible to identify the true generating model? We address this problem as one of model selection, as outlined in Section 5.4.2. Specifically, we perform a sensitivity analysis using the Bayesian Information Criterion (BIC) in order to choose the appropriate semantics. In this section, we provide experimental results that showcase the efficacy of this approach.

We assessed the efficacy of BIC for model selection in heterogeneous networks with a set of experiments on synthetic networks and data sets. We used the same settings for graph and data generation as the ones described in the previous experiment (Section 5.5). For every network type we ran 100 trials. For each network, we computed the BIC for each of the 5 possible models.

Figure 5.7 shows the average BIC score relative to the true model in the case of small-world networks for increasing noise coefficient. In every case, the average BIC score is lowest when the generative model matches the assumed model. The results are similar for graphs generated using the Erdös-Rényi and the stochastic block model. However, in the case of graphs generated using the Erdös-Rényi model, the BIC values for different models are not as diverse.

Figure 5.8 shows how often the correct generating model is selected for different types of network structure and as the noise increases. For comparison, the baseline for accuracy (selecting a model at random) is 20%. In all cases, the accuracy of model selection decreases as the noise coefficient increases. This is to be expected as the noise will "override" the influence from the different sets of peers. As the coefficients of different sets diverge (increasing weight specification), the accuracy increases. In other words, when the different types of peers influence the ego in significantly different ways, we can infer the correct model using BIC with higher accuracy. Moreover, the accuracy of model selection depends on the topology of the network. The results of this experiments suggest that model selection performs better for small-world networks generated using the Watts-Strogatz model. The least accurate case is that of random networks generated through the Erdös-Rényi model.

5.7 Experiments on Real-World Networks

To demonstrate the applicability of our proposed methods in real-world situations, we used real data from a study on the diffusion of microfinancing loans through various social networks [5, 6]. The data was collected through a survey conducted in 75 villages in southern India. The survey consists of a village-level survey and then a follow-up survey on a subsample of individuals for each village. From the individual surveys, 13 different types of social relationships were extracted, such as friends, relatives, helping with a decision, borrowing money from, going to temple

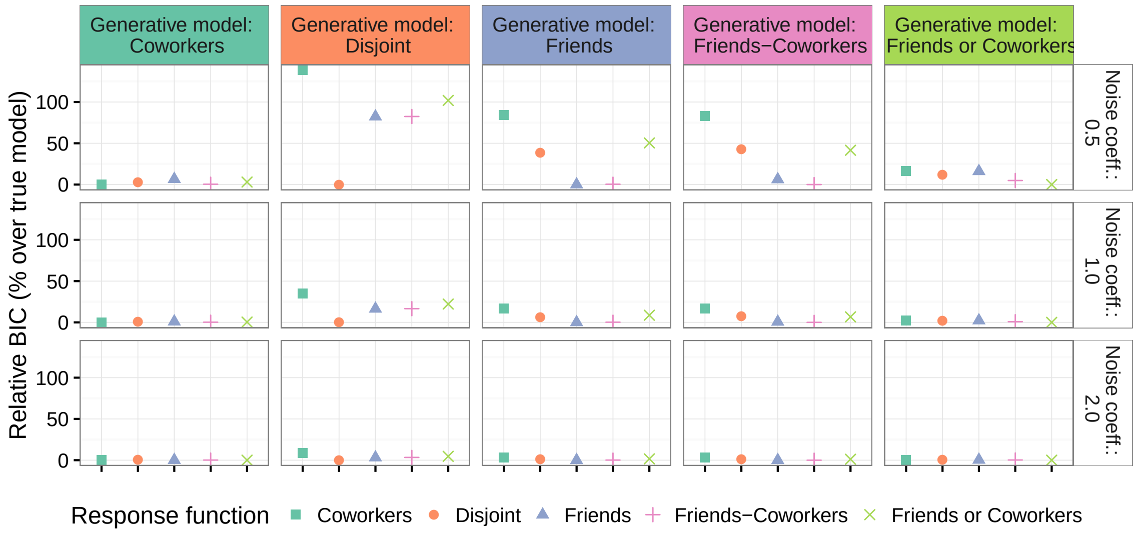


Figure 5.7: Average relative BIC (percent over true model) of assumed model for each of the true generative models (columns) and noise levels (rows). This is averaged across 100 trials for small-world networks with neighborhood size set to 3 and rewiring probability 0.2. In all cases, the average BIC score is lower when the assumed model matches the true generating model.

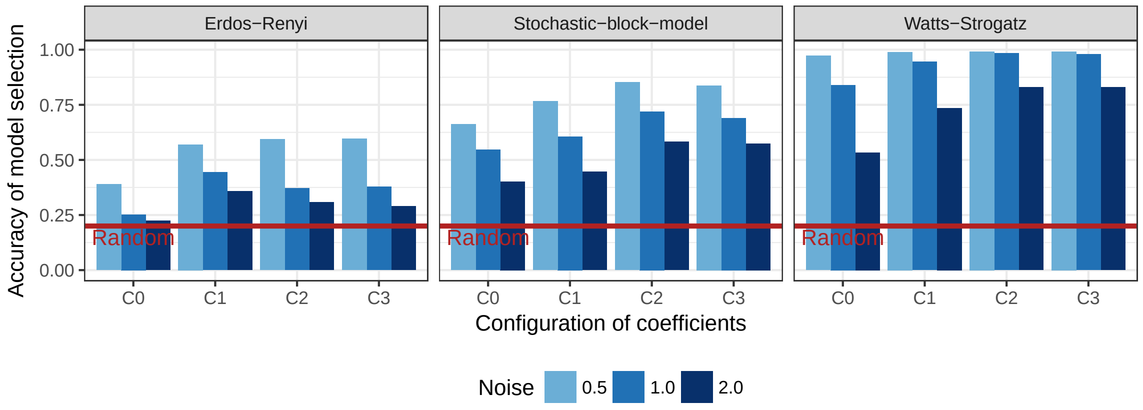


Figure 5.8: Accuracy of model selection for varying graph generating model, noise coefficient, and generative model specification. The baseline for accuracy is 20% (line shown in red).

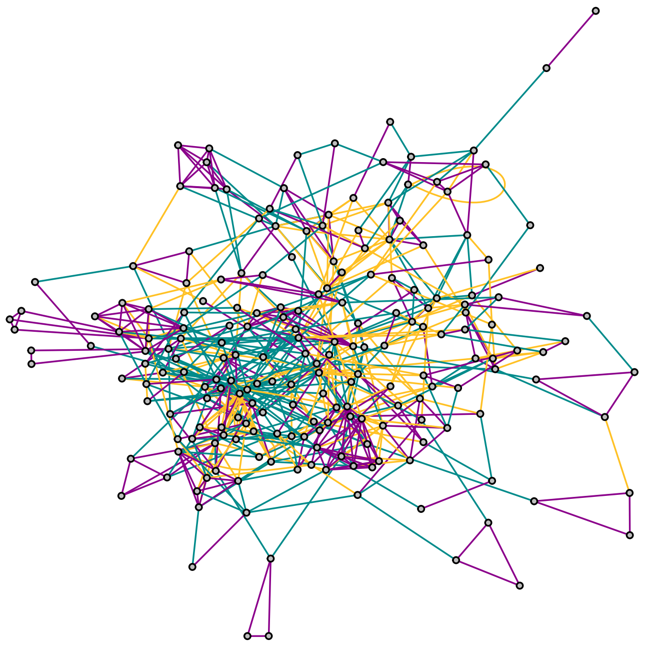


Figure 5.9: Social network for the individuals of village 1 for which there are survey data available. Green edges correspond to friends, yellow edges to relatives, and purple edges to connections that are both friends and relatives.

with. Individuals provided information on their age, gender, education level, work status, and more.

For our purposes, we restrict our analysis to people that completed the individual survey. While this introduces a selection bias to our estimates, it does not affect our main point: how different social relationships should be taken into account. But the reader should be cautious, the results for the real-world data set are not meant to be interpreted causally. Rather, they should be seen as a representative example displaying discrepancies that are exhibited in real-world data sets.

We consider several pairs of social relationships and combinations of treatment and outcome variables. The x-axis of Figure 5.10 shows the configurations we examined. One of the settings we considered focuses on friends and relatives and on the dependence: "how does gender affect working". Figure 5.9 shows the social network of the first village for these two relationships. It can be seen that there is non-negligible overlap between friends and relatives. Similar behavior is observed across villages and also across the pairs of relationships we considered.

For each of the aforementioned settings, we estimated the effect as described in section 5.4.1 under different response functions. The results are shown in Figure 5.10. For every estimate, we computed the standard error using 1000 bootstrapped samples. The estimated effect varies across the different response functions. In one of the cases we examined (first configuration), even the sign of the estimated effect changes for different response functions. These results provide evidence that difference response functions in heterogeneous networks do matter in estimation of effects.

Finally, we applied our proposed model selection technique in the aforementioned settings. In each case, the preferred model is the one where the two relationships are equally weighted (friends-coworkers).

5.8 Concluding Remarks

Causal inference is a fundamental tool for researchers and practitioners in a wide array of fields. The vast majority of tools that have been developed for causal inference rely on the assumption of i.i.d. data. Recent work has extended this to simple network structures, i.e., a single entity type and a single relationship type. In practice, many meaningful networks contain multiple types of relationships between individuals. In this chapter, we formally characterized the problem of causal effect estimation under multiple relationship types. Through evaluations on synthetic data, we have shown that significant bias can result from mis-specifying the relational structure of causal dependence. We also showed that for a single relational dependence the Bayesian information criterion can be used to distinguish between possible relational semantics. We showcased our findings on a real-world data set for the diffusion of micro-financing in indian villages.

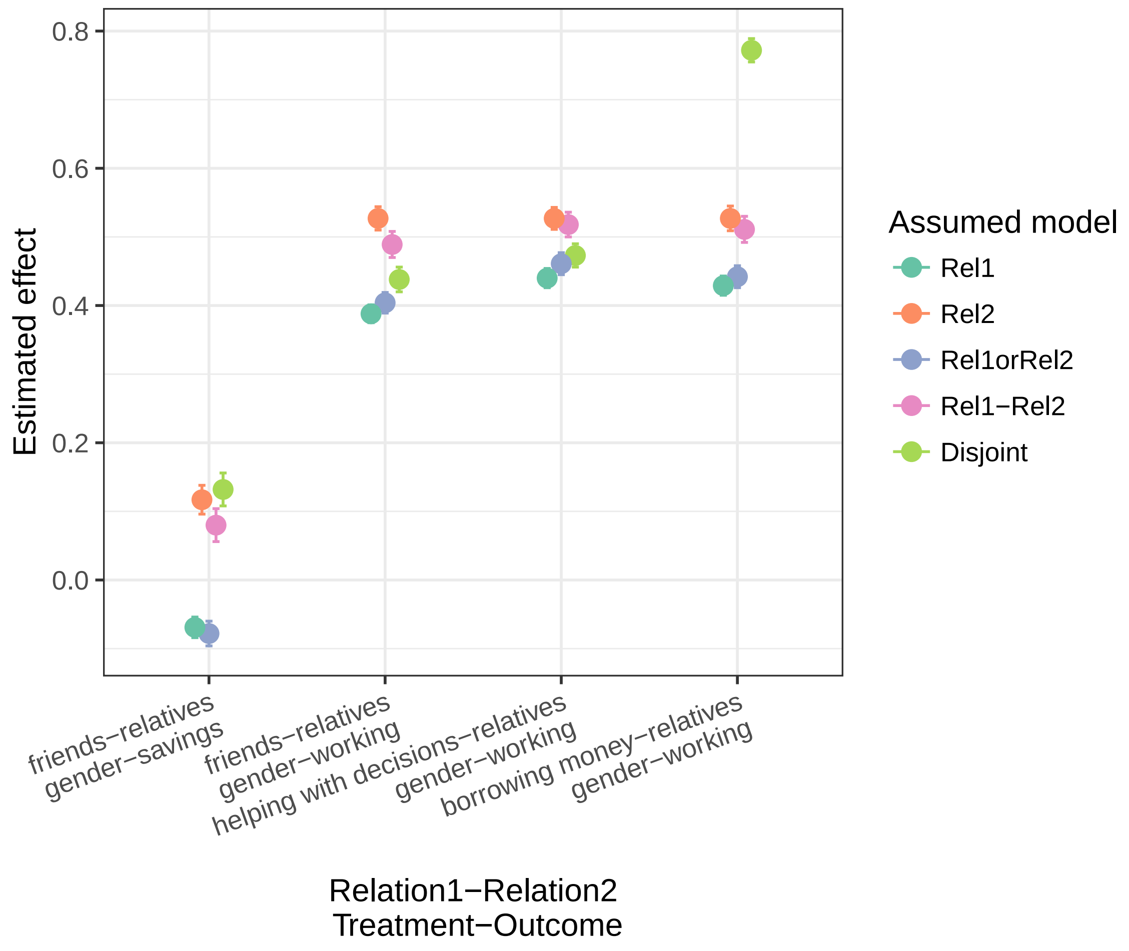


Figure 5.10: Estimated ATE (with standard error) for various pairs of relationships and pairs of treatment/outcome (shown on x-axis) for each of the five response functions (color).

CHAPTER 6
TEMPORAL RELATIONAL MODELS

So far, we have focused on relational models that do not explicitly model time. However, explicit modeling of time is critical, given that the world around us changes constantly and so do the systems we care to study. A system can change over time in many different ways. For example, consider the previous example of an academic domain. The relational skeleton might change over time: students are graduated, new students are admitted into a program, courses might be removed or a new course might be added. Moreover, a student that takes a course in the fall semester probably will not take the same course in the spring semester. Apart from the relational skeleton, the attributes of students and of courses are not necessarily static. For example, the GPA of students fluctuates over their course of study. Finally, the actual causal dependencies might change over time.

In this chapter, we extend the relational model described in Chapter 3 to explicitly model discrete time. This extension is similar to that of Bayesian networks to dynamic Bayesian networks. We then extend abstract ground graphs and the theory of relational d-separation for the class of temporal relational models.[1]

6.1 Related Work

The rest of this thesis focuses on extending the expressivity of current representations for causal graphical models and on presenting structure learning algorithms

[1]Marazopoulou, Maier, Jensen. Learning the Structure of Causal Models with Relational and Temporal Dependence. Proceedings of the 31st Conference in Artificial Intelligence (2015) [54]

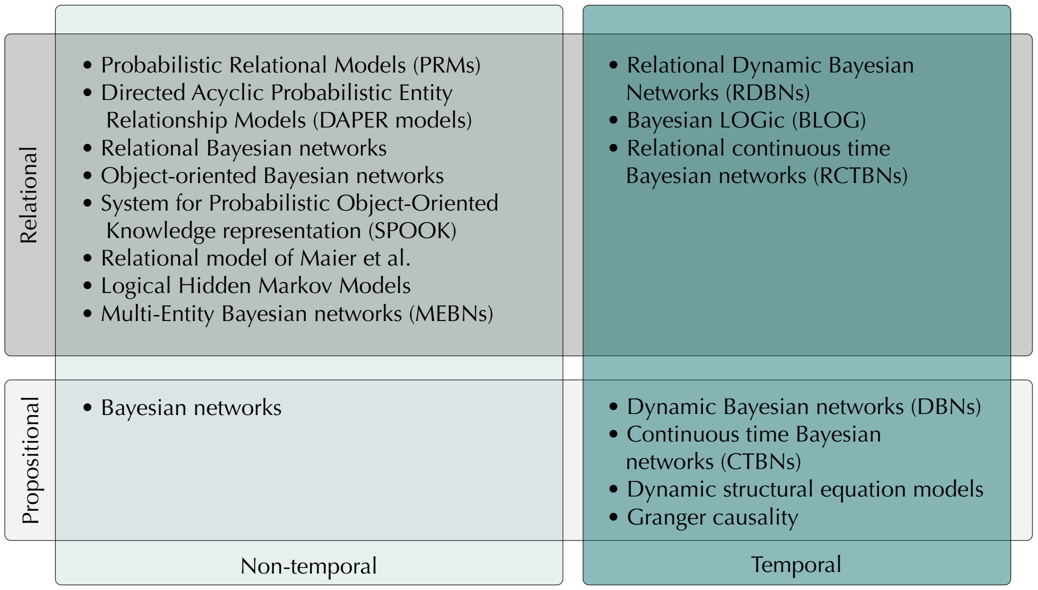

Relational
Propositional
Non-temporal
Temporal
• Probabilistic Relational Models (PRMs)
• Directed Acyclic Probabilistic Entity Relationship Models (DAPER models)
• Relational Bayesian networks
• Object-oriented Bayesian networks
• System for Probabilistic Object-Oriented Knowledge representation (SPOOK)
• Relational model of Maier et al.
• Logical Hidden Markov Models
• Multi-Entity Bayesian networks (MEBNs)
• Relational Dynamic Bayesian Networks (RDBNs)
• Bayesian LOGic (BLOG)
• Relational continuous time Bayesian networks (RCTBNs)
• Bayesian networks
• Dynamic Bayesian networks (DBNs)
• Continuous time Bayesian networks (CTBNs)
• Dynamic structural equation models
• Granger causality

Figure 6.1: Existing representations categorized by their expressiveness along the temporal and relational dimensions. The most expressive case is that of representations that are both temporal and relational (upper-right box). This is the area of our contributions.

for these expressive models. Relevant work falls into two main categories: representation and learning. Figure 6.1 visualizes the landscape of existing representations along the temporal and relational dimension. In what follows, we review existing representations and the corresponding learning algorithms.

6.1.1 Propositional

The standard probabilistic representation for propositional data is Bayesian networks (see section 2.1). There exists a plethora of algorithms for learning the structure of Bayesian networks from data, as described in section 2.4.

6.1.2 Temporal Propositional

Dynamic Bayesian networks (DBNs) [25, 58] extend Bayesian networks to include a temporal component. A DBN is essentially a set of Bayesian networks, one for each time point, where directed probabilistic dependencies are allowed between different time points.

There are several approaches for learning the structure of DBNs. Most of them are not constraint-based methods, but follow the score-based paradigm. Friedman et al. [25] present an algorithm to learn the structure of DBNs from complete data using a score-based algorithm, and from incomplete data using structural expectation-maximization (EM). Lähdesmäki et al. [43] use Bayesian methods to learn the structure of a DBN from steady state measurements or from time-series data and steady state measurements. Robinson et al. [70] present an MCMC algorithm to learn the structure of a DBN that changes over time. A constraint-based method for temporal propositional domains is presented by Entner et al. [19] who extend the FCI algorithm [76] to temporal domains. FCI is a constraint-based algorithm that relaxes the causal sufficiency assumption, i.e., it allows the presence of latent common causes.

One of the main limitations of dynamic Bayesian networks is that they model time as a discrete quantity. This raises the question, how to choose the appropriate granularity for the time points? The wrong choice for time granularity might lead to spurious dependencies (when undersampling) or computational inefficiencies (when oversampling). Ribeiro et al. [67] provide an analysis on how aggregating at a given time granularity affects the characterization of the underlying temporal process.

In practice, many temporal processes do not have a fixed and well-defined time granularity. In these cases, it is more intuitive to represent time as a continuous quantity. Continuous time Bayesian networks (CTBNs) [59] provide a framework to achieve this in the propositional setting, by allowing each variable to be modeled at a different time granularity. There are also approaches based on CTBNs that automatically select the right time granularity [73].

Structure learning for CTBNs is following the score-based paradigm [60]. Regarding the complexity of structure learning for CTBNs, Nodelman et al. [60] show that when a node is restricted to have at most k parents, searching for an optimal CTBN can be done in polynomial time with respect to the number of variables and the size of the data set. This is in contrast to score-based algorithms for Bayesian networks that are shown to be NP-hard [10].[2] Intuitively, the complexity of structure learning for Bayesian networks stems from the acyclicity constraint. The optimal parents of a node cannot be determined locally, without considering the rest of the graph. Choosing parents for a node affects the valid parents for other nodes due to the acyclicity constraint. However, in the case of CTBNs, the optimal parents can be chosen independently for every node.

A different approach that learns a causal temporal model from time series data is the difference-based causality learner [86]. This framework is based on dynamic Structural Equation Models.

Another widely used method for inferring causal relationships from temporal data is Granger causality [29]. The main idea underlying Granger causality is that a cause should improve the predictive accuracy of its effect, compared to predictions based solely on the effect's past values. There has been work in extending Granger causality for multivariate settings, as well as in combining Granger causality with graphical models [18]. Liu et al. [48] propose a regularized hidden Markov random field regression to learn the structure of a temporal causal graph from multivariate time-series data.

6.1.3 Relational

As far as relational models are concerned, there exist representations that extend Bayesian networks to the relational domain. Object-Oriented Bayesian Networks (OOBNs) [42] only support part-whole relations between objects and not arbitrary relations. The System for Probabilistic Object-Oriented Knowledge representation

[2]This also holds for dynamic Bayesian networks that allow for within-slice dependencies.

(SPOOK) [64] is an extension of OOBNs that is more general, but it only handles binary relations among objects. Probabilistic Relational Models (PRMs) [24] and Relational Bayesian networks [39] are more expressive than the aforementioned models. Structure learning in PRMs is achieved through score-based methods. Directed-Acyclic Entity-Relationship (DAPER) models [32] are a more expressive family of relational models and are a combination of ER diagrams and first-order logic. Finally, in prior work, Maier, Marazopoulou, and Jensen [50] introduce a relational model based on DAPER models by constraining the type of labels on dependencies. For that class of models, we provide a constraint-based algorithm for structure learning, the Relational Causal Discovery (RCD) algorithm [49]. To the best of our knowledge, this is the first sound and complete constraint-based method for learning the structure of relational models from data.

6.1.4 Temporal Relational

Regarding representations that are both temporal and relational, Manfredotti [51] introduces relational dynamic Bayesian networks (RDBNs), a first-order logic-based extension of dynamic Bayesian networks. RDBNs are similar to the temporal relational model we define later on; however, our model provides an explicit characterization for the space of relational features (specifically, the space of relational paths, defined later), and subsequently, the space of relational dependencies it considers. To be more specific, temporal relational paths in our framework are restricted to conjunctions of predicates that correspond to possible traversals of the relational schema. This restriction, together with the domain specific hop threshold, allows us to enumerate the space of potential dependencies to learn, thus rendering the learning task feasible.

Bayesian logic (BLOG) [56] is a probabilistic first-order language that allows for unknown objects in the domain. Multi-entity Bayesian networks (MEBNs) [44] use

MEBN fragments to represent probabilistic information about a set of related random variables. Those fragments can be combined to create a more complex network. Both BLOG and MEBNs can incorporate time by adding a variable in their first-order language that represents time.

The temporal relational model we propose bears similarity to the aforementioned formalisms. However, our temporal relational model is explicitly formalized as a graphical representation (as opposed to BLOG and MEBNs). More importantly, we leverage that graphical representation to introduce a temporal relational d-separation criterion which explicitly ties the structure of the model to a set of independencies that hold in the underlying distribution. Finally, all three of the representations mentioned above are first-order languages and, in the general case, learning such structures is intractable and no structure-learning algorithms have been provided. In this thesis, we provide a constraint-based structure learning algorithm for our temporal relational model.

Another line of work in learning temporal and relational models stems from combining first-order logic with probabilistic frameworks. Logical hidden Markov models (LHMMs) extend hidden Markov models to handle relational (non-flat data) [40]. Kersting et al. [41] provide an EM-based algorithm to learn the structure of LHMMs.

Focusing on social networks (a specific type of relational domains), modeling change in social networks using continuous time stochastic processes is not a new concept. It has been used by researchers in social mobility [8] as well as in social networks [88, 87]. Wasserman [87] introduced two simple models for studying the structural changes of social networks over time. These models are fairly simple and, as the author notes, they were a first step towards more complicated models that would fit the data better.

Regarding relational extensions of CTBNs, Fan et al. [22] provide such and extension for the case of social networks. Their work is limited in learning the parameters of a CTBN defined over a social network assuming a fixed graph structure. Finally, Yang et al. [91] provide a fully relational extension of CTBNs, RCTBNs (relational continuous-time Bayesian networks), and a non-parametric method to learn the structure and parameters of RCTBNs from data.

6.2 Temporal Relational Concepts

We now introduce the representation for our temporal relational graphical model. This extension is similar to the way in which dynamic Bayesian networks (DBNs) extend Bayesian networks [15, 58]. A temporal relational model can be thought of as a sequence of time points, each of which is associated with a (non-temporal) relational model, and a set of dependencies that cross time points. Because dependencies in this model have a causal interpretation, dependencies across time points are only directed from the past to the future.

In this section, we extend the relational concepts presented in Section 3.1 to include time. We assume the following:

- Time is discrete;
- The schema is static;
- Relational dependencies do not change over time (the model is stationary);
- The temporal relational skeleton is given *a priori*;
- The first-order Markov assumption holds (i.e., treatment and outcome can be at most one time point apart); and
- All entities participating in a relationship are contemporaneous with the relationship.

Under these assumptions, we only need to represent two consecutive time points. Every time point has the same relational schema. Therefore, a *temporal relational*

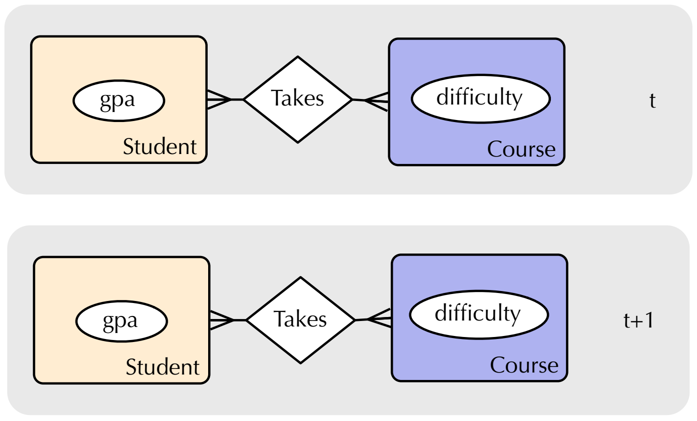


Figure 6.2: Example temporal schema for the academic domain. Notice that we only represent two consecutive time points and the schema is the same for every time point.

schema is defined in the same way as a relational schema, $\mathcal{S} = \langle \mathcal{E}, \mathcal{R}, \mathcal{A} \rangle$. We will use the notation I^t to denote item I in time point t. More generally, the temporal information will be denoted as a superscript. Graphically, we depict the temporal relational schema by replicating the schema in two consicutive time points t and $t + 1$, as shown in Figure 6.2.

Given a temporal relational schema, we can specify *temporal relational paths*, which capture possible ways to traverse the temporal schema across the two time points. Contrary to (non-temporal) relational paths, temporal relational paths can cross time points. A temporal relational path is a sequence of non-temporal relational paths (relational paths within a time point) and "jumps" between neighboring time points. These jumps can happen at both entities and relationships because each choice encodes a distinct semantics.

Definition 6.1. A *temporal relational path* P is a sequence of non-temporal relational paths $P_0^{t_0}, \ldots, P_k^{t_k}$ $(k \geq 0)$ such that for any two consecutive paths $P_i^{t_i}, P_j^{t_j}$ in P the following hold:

1. $|t_i - t_j| = 1$

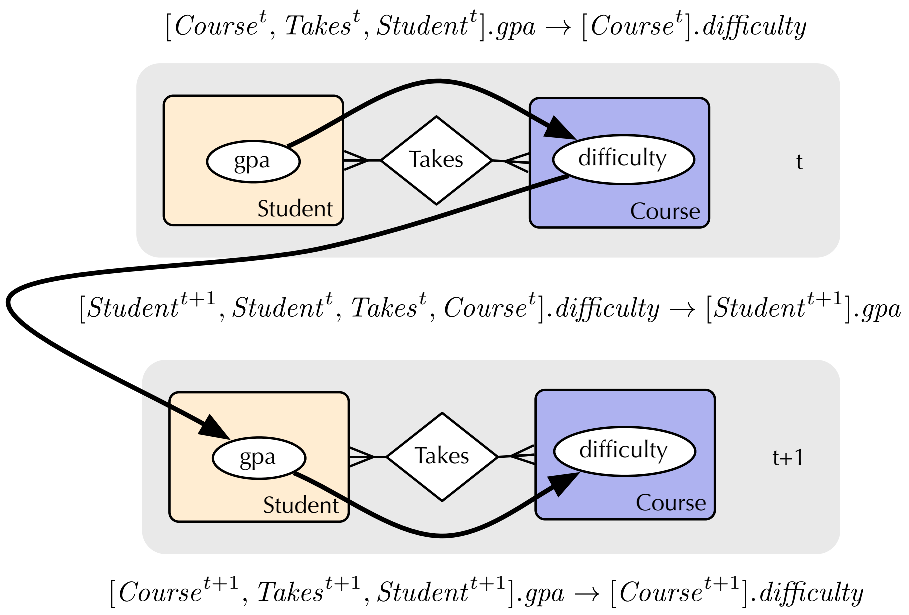

$[Course^t, Takes^t, Student^t].gpa \to [Course^t].difficulty$
gpa
Student
Takes
difficulty
Course
t
$[Student^{t+1}, Student^t, Takes^t, Course^t].difficulty \to [Student^{t+1}].gpa$
gpa
Student
Takes
difficulty
Course
t+1
$[Course^{t+1}, Takes^{t+1}, Student^{t+1}].gpa \to [Course^{t+1}].difficulty$

Figure 6.3: Example structure of a temporal relational model.

2. The last item class of $P_i^{t_i}$ is the same as the first item class of $P_j^{t_j}$.

3. No subpath of P is of the form $[I_k^t, \dots, I_k^t]$, where all relations in the subpath are one-to-one.

Condition 1 implies that relational paths can only "jump" one time point into the future or the past at a time. This does not limit the expressiveness, since those "atomic jumps" can be combined to reach time points further away. We use the notation $time(P)$ to denote the set of all time points that appear in path P.

For the temporal relational schema of Figure 6.2, an example temporal relational path is $[Student^{t+1}, Student^t, Takes^t, Course^t]$, which describes the classes that a student took in the previous semester. Regarding the difference between paths that "jump" at entities versus paths that "jump" in relationships, consider the following two paths, that start at the same item class, end at the same item class, but make temporal jumps at different item classes: $[Student^{t+1}, Student^t, Takes^t, Course^t]$ describes the courses that a student took in the previous semester, while the path

[*Student*$^{t+1}$, *Takes*$^{t+1}$, *Course*$^{t+1}$, *Course*^{t}] first finds the courses that a student is taking this semester, and then finds those courses in the previous semester.

Similar to non-temporal relational paths, temporal relational paths are templates. Instantiating a temporal relational path results in a set of items reachable from a given starting item along that path.

Temporal relational variables consist of a temporal relational path and an attribute that can be reached through that path.

Definition 6.2 (Temporal relational variables)**.** A temporal relational variable

$$[I_1^{t_1}, \ldots, I_k^{t_k}].A$$

consists of a temporal relational path $[I_1^{t_1}, \ldots, I_k^{t_k}]$ and an attribute class $A \in \mathcal{A}(I_k^{t_k})$.

The temporal relational variable [*Student*$^{t+1}$, *Student*^{t}, *Takes*^{t}, *Course*^{t}].*difficulty* corresponds to the set of *difficulty* attributes of the courses that a student takes. To be more specific, instantiating this temporal relational variable for a given student in time point $t+1$, results in a set of random variables of type *difficulty*, one for every course that the student took in the previous semester (at time point t).

Temporal relational dependencies define probabilistic dependencies between two temporal relational variables. Temporal probabilistic dependencies are never directed backwards in time. Therefore, at the model level, there are no dependencies going back in time. However, the temporal constraints associated with a dependency are also implicitly encoded by the temporal relational path that annotates the dependency. To account for this, we do not allow the temporal relational path of the treatment to go through any time points later than the time point of the outcome.

Definition 6.3 (Temporal relational dependency)**.** A *temporal relational dependency*, $[I_1^t, \ldots, I_k^{t'}].A_k \to [I_1^t].A_1$, consists of two temporal relational variables with a common base item such that

$$\max\Big(\mathit{time}\big([I_1^t, \ldots, I_k^{t'}]\big)\Big) \leq t$$

For example, the temporal relational dependency:

$$[\mathit{Student}^{t+1}, \mathit{Student}^{t}, \mathit{Takes}^{t}, \mathit{Course}^{t}].\mathit{difficulty} \to [\mathit{Student}^{t+1}].\mathit{gpa}$$

shows that the GPA of a student in the spring semester depends on the set of *difficulty* attributes of the courses that the student took in the fall semester. If, for example, a student takes many difficult classes, it is more likely that the student's GPA will drop in the next semester.

The first-order Markov assumption for temporal causal models implies that for every probabilistic dependency, if the treatment is in time point t, then the outcome is either in t or in $t+1$. In the case of relational domains, the temporal relational path of the treatment carries some temporal information since it can contain multiple time points. For the first-order Markov condition to hold, we require the relational path of the treatment to only go through the current and the next time points. More formally:

Definition 6.4 (Relational first-order Markov assumption)**.** A temporal relational dependency

$$[I_1^t, \ldots, I_k^{t'}].V_k \to [I_1^t].V_1$$

follows the first-order Markov assumption if the following two conditions hold:

$$t = t' \text{ or } t = t' + 1 \tag{6.1}$$

$$\mathit{time}([I_1^t, \ldots, I_k^{t'}]) \subseteq \{t, t-1\} \tag{6.2}$$

In other words, the outcome is at most one time step ahead of the treatment and the relational path of the treatment stays within the current and the next time point.

A temporal relational model can be represented by using only two consecutive time points (2-slice model). The structure of a 2-slice temporal relational model is defined as a set of temporal relational dependencies over a relational schema.

Definition 6.5 (Structure of temporal relational model)**.** The structure of a *2-slice temporal relational model* is a pair $\mathcal{M} = \langle \mathcal{S}, \mathcal{D}_T \rangle$, where $\mathcal{S}$ is a relational schema and $\mathcal{D}_T$ is a set of temporal relational dependencies defined over $\mathcal{S}$, which adhere to the first-order Markov assumption.

Figure 6.3 shows the structure of a 2-slice temporal relational model for the university domain. Within each time point the relational structure remains the same, i.e., Students take Courses. Moreover, the causal structure within a time point is the same across all points, and any two adjacent time points have the same temporal dependencies between them. That means that in every time point the gpa of the students that take a course affects the difficulty of the course. The temporal dependency shows that the difficulty of a course in the fall semester affects the spring GPA of the students that took this class in the fall.

6.3 Instantiation of Temporal Relational Concepts

The temporal relational concepts presented in Section 6.2 are defined on a template level. In this section, we formally specify how these concepts can be instantiated.

A *temporal relational skeleton* σ_T provides a partial instantiation of a temporal relational schema. It specifies the entity and relationship instances that exist in each time point. Note that different sets of entity and relationship instances may be present in each time step.

An example temporal relational skeleton is shown in Figure 6.4. In this case, the skeleton consists of three time points. The first two have the same set of entity

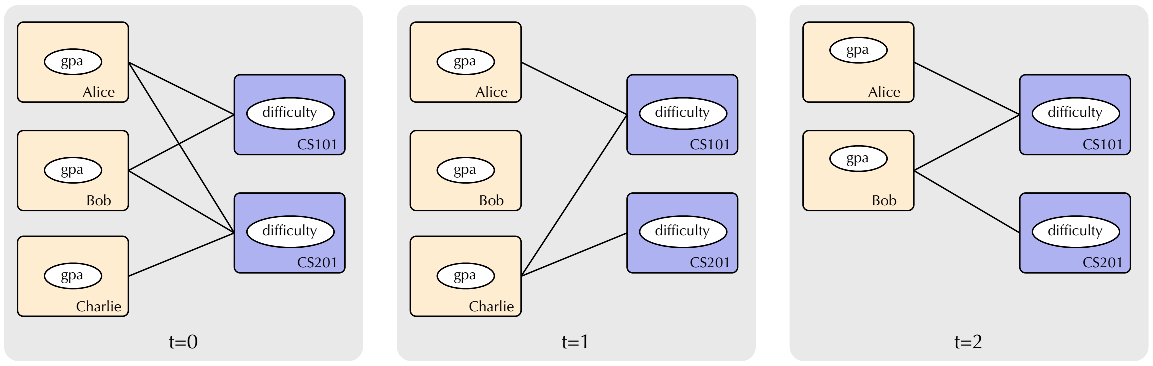


Figure 6.4: Example temporal relational skeleton for the schema shown in Figure 6.2.

instances (Alice, Bob and Charlie are instances of the *Student* entity, CS101 and CS201 are instances of the *Course* entity), while in $t = 2$ Charlie is no longer a student. The instances of the relationship *Takes* differ across time points. In $t = 0$, Alice and Bob take both classes. In $t = 1$, Alice takes CS101 and Bob takes no classes.

Given a temporal relational skeleton, temporal relational paths can be instantiated, starting from a specific item and reaching a set of items along the path.

Definition 6.6 (Terminal set of a temporal relational path)**.** Let σ_T be a temporal relational skeleton for temporal relational schema $\mathcal{S}$, $P = [I_1^{t_1}, \ldots, I_k^{t_k}]$ be a temporal relational path for $\mathcal{S}$, and $i_1^{t_1}$ an instance of item $I_1^{t_1}$ in the skeleton σ_T. The *terminal set* $P|_{i_1^{t_1}}$ is inductively defined as follows:

$$P^1|_{i_1} = [I_1^{t_1}]|_{i_1^{t_1}} = \{i_1^{t_1}\}$$

$$\vdots$$

$$P^k|_{i_1} = [I_1^{t_1}, \ldots, I_k^{t_k}]|_{i_1^{t_1}} = \bigcup_{i_m^{t_m} \in P^{k-1}|_{i_1 t_1}} \Big\{ i_k^{t_k} \mid \quad \big((i_{k-1}^{t_{k-1}} \in i_k^{t_k} \text{if } I_k^{t_k} \in \mathcal{R}) \\ \text{or } (i_k^{t_k} \in i_{k-1}^{t_{k-1}} \text{ if } I_k^{t_k} \in \mathcal{E}) \big) \\ \text{and } i_k \notin \bigcup_{j=1}^{k-1} P^j|_{i_1}^{t_1} \Big\}$$

This definition makes use of the bridge burning semantics, following the definition of relational paths for the non-temporal case (Definition 3.2).

In the example temporal skeleton of Figure 6.4, starting at *Alice* at $t+1$, the terminal sets for some example paths are given below:

$$[\mathit{Student}^{t+1}, \mathit{Student}^{t}, \mathit{Takes}^{t}, \mathit{Course}^{t}]|_{\mathit{Alice}^{t+1}} = \{\mathit{CS101}^{t}\}$$
$$[\mathit{Student}^{t+1}, \mathit{Takes}^{t+1}, \mathit{Course}^{t+1}, \mathit{Course}^{t}]|_{\mathit{Alice}^{t+1}} = \{\mathit{CS201}^{t}\}$$

Definition 6.7 (Temporal relational variable instance)**.** Let σ_T be a relational skeleton, $i_1^{t_1}$ an instance of item class $I_1^{t_1}$ in the skeleton σ_T, and $[I_1^{t_1}, \dots, I_k^{t_k}].A$ a temporal relational variable. An instance of the relational variable, denoted as $[I_1^{t_1}, \dots, I_k^{t_k}].A|_{i_1^{t_1}}$, is the set of attribute instances $\{i_k^{t_k}.A \mid A \in \mathcal{A}(i_k^{t_k}) \text{ and } i_k^{t_k} \in [I_1^{t_1}, \dots, I_k^{t_k}]|_{i_1^{t_1}}\}$.

Finally, given a temporal relational skeleton σ_T and a temporal relational model $\mathcal{M}$, we can construct a *temporal ground graph* $GG_{\mathcal{M}\sigma_T}$ in the same way as in the non-temporal case.

Definition 6.8 (Ground graph)**.** Let $\mathcal{M}_T = \langle \mathcal{S}, \mathcal{D}_T \rangle$ be a temporal relational model and σ_T a relational skeleton for that model. The ground graph $GG_{\mathcal{M}_T\sigma_T} = \langle V, E \rangle$ is a directed graph with nodes and edges defined as follows:

$$\begin{aligned}
V &= \{i^t.X \mid I^t \in \mathcal{E}_T \cup \mathcal{R}_T \text{ and } X \in \mathcal{A}(I^t)\} \\
E &= \{i_k^{t_k}.Y \to i_j^{t_j}.X \mid i_k^{t_k}.Y, i_j^{t_j}.X \in V \text{ and} \\
&\qquad\qquad i_k^{t_k}.Y \in [I_j^{t_j}, \ldots, I_k^{t_k}].Y|_{i_j^{t_j}} \text{ and} \\
&\qquad\qquad [I_j^{t_j}, \ldots, I_k^{t_k}].Y \to [I_j^{t_j}].X \in \mathcal{D}_T\}
\end{aligned}$$

Figure 6.5 shows the ground graph for the model of Figure 6.3 and the skeleton of 6.4.

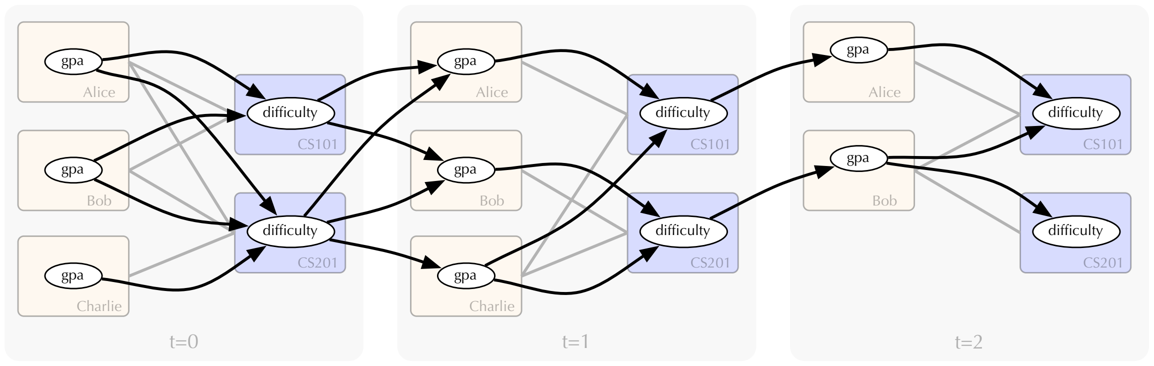


Figure 6.5: Temporal ground graph for the temporal relational model shown in Figure 6.3 applied on the temporal relational skeleton shown in Figure 6.4.

6.4 Expressiveness of Temporal Relational Models

The temporal relational model described so far subsumes propositional directed networks (Bayesian networks), propositional temporal directed networks (dynamic Bayesian networks), and relational non-temporal models. This added expressivity comes at the cost of increased complexity. This raises an obvious and important question: What is the value of this added expressivity?

As an example of the practical utility of this added expressivity, we consider a real data set and show how a path with temporal jumps leads to different terminal sets. Specifically, we used the Reality Mining data set [17]. The data set contains two entities, *Tower* and *Cellphone*, and one relationship, *Connects*. The relational schema for this domain is shown in Figure 6.6. For this data set, entity instances (i.e., the set of towers and cellphones) do not change over time. However, the set of relationship instances (the connections) are different at every time point. In total, there are 95 cellphones, 32,579 towers, and 3,308,709 connections. Every connection is time-stamped with precision of 1 second. For our work, the time granularity was coarsened to a day.

We computed the terminal sets for the following three paths:

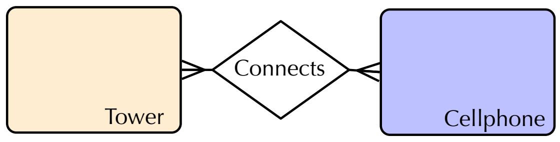


Figure 6.6: Relational schema for the Reality Mining data set.

$$P1 : [Tower^{t+1}, Tower^{t}, Connects^{t}, Cellphone^{t}]$$

$$P2 : [Tower^{t+1}, Connects^{t+1}, Connects^{t}, Cellphone^{t}]$$

$$P3 : [Tower^{t+1}, Connects^{t+1}, Cellphone^{t+1}, Cellphone^{t}]$$

The first path ($P1$) corresponds to the cellphones that were connected to a tower in the previous timestep. The second path ($P2$) corresponds to the cellphones that connected to a tower both in the current and in the previous timestep. The third path ($P3$) corresponds to the cellphones that connected to a tower in the current time step, and gets the state of those cellphones in the previous timestep.

We randomly selected 100 dates from the data set. For each of these dates and for each tower that was used in these dates, we computed the terminal sets for the above paths. For a given tower and date, we computed the Jaccard distance between the different terminal sets. The Jaccard distance between two sets A and B is defined as

$$J(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|}.$$

Intuitively, this quantifies the overlap of two sets, while accounting for the size of both. Table 6.1 shows the average Jaccard distance between the terminal sets, averaged across the dates and the towers. The results indicate that, on average, the terminal sets reached through the three paths will be different; therefore, this more expressive representation could be of use in real data.

Table 6.1: Jaccard distance between the terminal sets of the different paths for the reality mining data set (100 sample dates, distance is averaged across dates and across towers).

Jaccard distance	P1 vs. P3	P1 vs. P2	P2 vs. P3
mean	0.47	0.31	0.31
min	0	0	0
max	1	1	1
median	0.5	0	0

6.5 Temporal Abstract Ground Graphs

In this chapter we turn our attention to properties of temporal relational models that enable reasoning. Specifically, we focus on d-separation properties. As we have shown in previous work [50], the rules of traditional d-separation cannot be naively applied to the structure of relational models in order to infer conditional independencies. This is true for the case of the temporal relational models as well. To resolve that, we introduced an auxiliary representation, abstract ground graphs (AGGs), that abstract the independencies that hold over all possible ground graphs. Then we provided a modified version of d-separation, *relational d-separation*, that can be applied on abstract ground graphs. In what follows, we extend the notions of abstract ground graphs and relational d-separation for the case of a 2-slice temporal relational models.

An abstract ground graph is a representation that abstracts paths of dependence over all possible ground graphs for a given relational model [50]. Abstract ground graphs are shown to be sound and complete in the sense that every edge in the abstract ground graph corresponds to an edge in some ground graph, and every edge in an arbitrary ground graph is represented by an edge in an abstract ground graph. Here, we adapt the definition of abstract ground graphs for the case of a 2-slice temporal relational model.

Definition 6.9. A *temporal abstract ground graph* $tAGG_{\mathcal{M}Bh} = \langle V, E \rangle$ for a 2-slice temporal relational model $\mathcal{M} = \langle \mathcal{S}, \mathcal{D}_T \rangle$, perspective $B \in \mathcal{E} \cup \mathcal{R}$, and hop threshold

$h \in \mathbb{N}^0$ is an abstraction of the dependencies $\mathcal{D}_T$ for all possible ground graphs $GG_{\mathcal{M}\sigma_T}$ of $\mathcal{M}$ on arbitrary temporal skeletons σ_T. The temporal abstract ground graph is a directed graph with the following nodes and edges:

1. $V = RV \cup IV$, where
   (a) RV is the set of *temporal relational variables* with a path of length at most $h+1$.
   $$RV = \{[B^t, \dots, I_j^{t'}].V \mid \mathit{length}([B^t, \dots, I_j^{t'}]) \le h+1\}$$
   (b) IV are *intersection variables* between pairs of temporal relational variables that could intersect.[3]
   $$IV = \{X \cap Y \mid X, Y \in RV \text{ and } X = [B^t, \dots, I_k^{t'}, \dots, I_j^{t''}].V \\ \text{and } Y = [B^t, \dots, I_l^{t'''}, \dots, I_j^{t''}].V \text{ and } I_k^{t'} \ne I_l^{t'''}\}$$

2. $E = RVE \cup IVE$, where
   (a) $RVE \subset RV \times RV$ are the *relational variable edges*:
   $$RVE = \{[B^t, \dots, I_k^{t'}].V_k \to [B^t, \dots, I_j^{t''}].V_j \mid \\ [I_j^{t''}, \dots, I_k^{t'}].V_k \to [I_j^{t''}].V_j \in \mathcal{D}_T \text{ and} \\ [B^t, \dots, I_k^{t'}] \in \mathit{extend}([B^t, \dots, I_j^{t''}], [I_j^{t''}, \dots, I_k^{t'}])\}$$
   (b) $IVE \subset \big(IV \times RV\big) \cup \big(RV \times IV\big)$ are the *intersection variable edges.* This is the set of edges that intersection variables "inherit" from the relational variables that they were created from.

[3]Only relational variables in the same time point can intersect.

The *extend* method converts dependencies of the model, specified in the canonical form, into dependencies from the perspective of the abstract ground graph. An example temporal abstract ground graph for a model on the student-courses domain with the dependency $[\mathit{Student}^{t+1}, \mathit{Student}^{t}, \mathit{Takes}^{t}, \mathit{Course}^{t}].\mathit{difficulty} \rightarrow [\mathit{Student}^{t+1}].\mathit{gpa}$ is shown in Figure 6.7b. This abstract ground graph is from the perspective of *Student* and for hop threshold $h = 4$. Disconnected nodes are omitted.

Definition 6.10 (Extend). Let $P_{orig} = [I_1, \ldots, I_j]$ and let $P_{ext} = [I_j, \ldots, I_k]$ be two temporal relational paths for schema $\mathcal{S}$. The following three functions extend P_{orig} with P_{ext}:

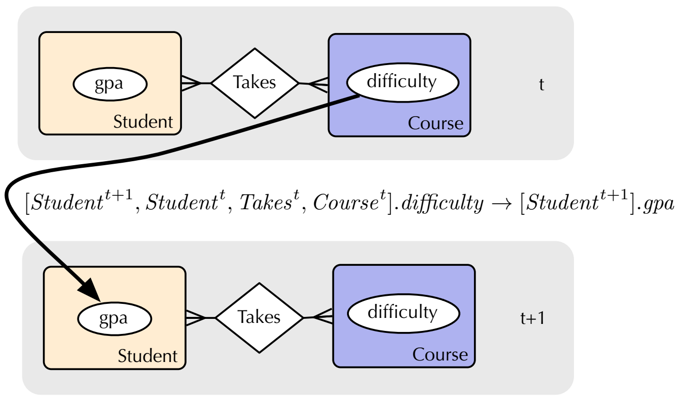
$$\begin{aligned}
\mathit{extend}(P_{orig}, P_{ext}) = \Big\{ & P = \mathit{concat}\big(P_{orig}[0 : \mathit{length}(P_{orig}) - i + 1], P_{ext}[i : \mathit{length}(P_{ext})]\big) \mid \\
& i \in \mathit{pivots}\big(\mathit{reverse}(P_{orig}), P_{ext}\big) \ \wedge \ \mathit{isValid}(P) \Big\} \\
\mathit{pivots}(P_1, P_2) = {} & \{ i \mid P_1[0 : i] = P_2[0 : i] \} \\
\mathit{isValid}(P) = {} & \begin{cases} \text{True} & \text{if } P \text{ does not violate Definition 6.1} \\ \text{False} & \text{otherwise} \end{cases}
\end{aligned}$$

where *concat*, *length*, *reverse*, and $[i : j]$ inclusive-exclusive sublist are standard list functions.

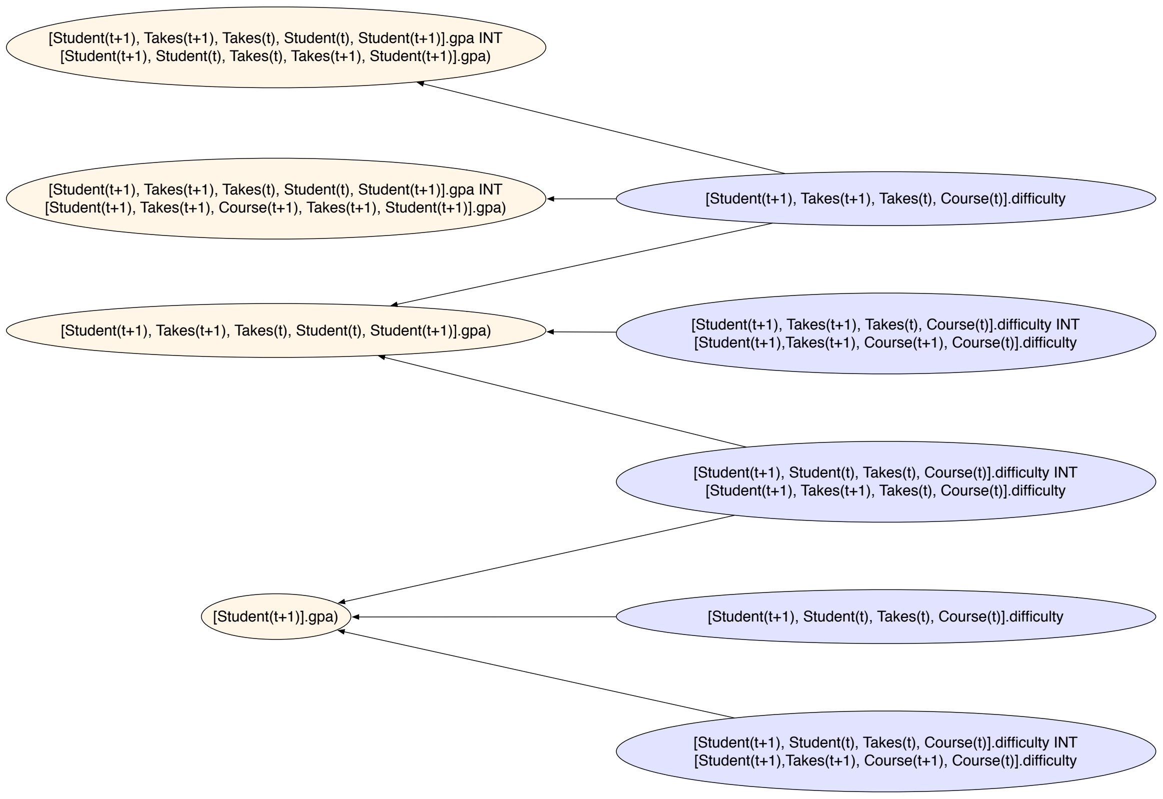
Temporal abstract ground graphs can be shown to be a correct abstraction over all possible temporal ground graphs. The proof follows the one provided for the non-temporal abstract ground graphs, as presented by Maier, Marazopoulou, and Jensen [50].

6.6 Temporal Relational *d*-separation

The rules of *d*-separation provide a graphical criterion that specifies whether two sets of variables in a directed acyclic graph are conditionally independent given a


gpa
Student
Takes
difficulty
Course
t
$[Student^{t+1}, Student^{t}, Takes^{t}, Course^{t}].difficulty \rightarrow [Student^{t+1}].gpa$
gpa
Student
Takes
difficulty
Course
t+1

(a) Temporal model with one dependency.


[Student(t+1), Takes(t+1), Takes(t), Student(t), Student(t+1)].gpa INT [Student(t+1), Student(t), Takes(t), Takes(t+1), Student(t+1)].gpa)
[Student(t+1), Takes(t+1), Takes(t), Student(t), Student(t+1)].gpa INT [Student(t+1), Takes(t+1), Course(t+1), Takes(t+1), Student(t+1)].gpa)
[Student(t+1), Takes(t+1), Takes(t), Course(t)].difficulty
[Student(t+1), Takes(t+1), Takes(t), Student(t), Student(t+1)].gpa)
[Student(t+1), Takes(t+1), Takes(t), Course(t)].difficulty INT [Student(t+1),Takes(t+1), Course(t+1), Course(t)].difficulty
[Student(t+1), Student(t), Takes(t), Course(t)].difficulty INT [Student(t+1), Takes(t+1), Takes(t), Course(t)].difficulty
[Student(t+1)].gpa)
[Student(t+1), Student(t), Takes(t), Course(t)].difficulty
[Student(t+1), Student(t), Takes(t), Course(t)].difficulty INT [Student(t+1),Takes(t+1), Course(t+1), Course(t)].difficulty

(b) Temporal abstract ground graph.

Figure 6.7: Temporal model (6.7a) and the corresponding temporal abstract ground graph (6.7b) from the perspective of *Student* and hop threshold $h = 4$ for the model shown in 6.7a. INT denotes intersection variables between pairs of temporal relational variables.

third set of variables. The graphical criterion of d-separation provides a connection between the structure of a graphical model and the independencies that hold in the distribution it represents. In other words, d-separation enables reasoning from observed conditional independencies in the data to possible causal models that could produce those.

Ground graphs (and temporal ground graphs) are directed acyclic graphs; therefore, the rules of d-separation can be applied to them. In the case of domains where the first-order Markov assumption holds, a d-separating set for variables in the same time point can be found by examining only one time point in the past.

However, temporal ground graphs have a very important property if they are interpreted causally: dependencies never go towards a previous time point. In what follows we are investigating properties of d-connecting paths in temporal domains. Assume that we are given a temporal ground graph, and a conditional independence query involving variables of that graph. We are interested in answering the following question: is there a fragment of the ground graph we can use to answer the query? It can be shown that there is no d-connecting path that goes through time slices in the future of the maximum time point of the query.

Proposition 6.11. Let G be a directed acyclic temporal ground graph, $X^{t_x} \perp\!\!\!\perp Y^{t_y} \mid Z^{t_z}$ a conditional independence query where X, Y, Z are variables of G, and $t_{max} = \max\{t_x, t_y, t_z\}$. There is no d-connecting path between X^{t_x} and Y^{t_y} that goes through a time slice $t > t_{max}$.

Proof. Assume for contradiction that there exists a d-connecting path P_a that goes through some variable W^{t_a} with $t_a > t_{max}$. Without loss of generality, let t_a be the maximum time that appears in P_a. Probabilistic dependencies always go from the past to the future, therefore, P_a must have a collider at t_a. This collider is not conditioned on, since all conditioning variables appear at or before t_{max}. Therefore, P_a is not d-connecting, which is a contradiction. □

We are interested in finding if two variables in adjacent time points are d-separated, when possible conditioning variables are only within those two time points. The following proposition states that if two variables in the same time point $t+1$ are conditionally independent given some set, then there exists a separating set that contains only variables in t or $t+1$. Moreover, if two variables in adjacent time points, t and $t+1$ are conditionally independent given a set, then there exists again a separating set that contains only variables in t or $t+1$.

Proposition 6.12. Let G be a temporal directed acyclic graph that follows the first-order Markov assumption.

1. If X^{t+1} and Y^{t+1} are conditionally independent given some set $\mathbf{Z}$, then there exists a separating set $\mathbf{W}$ such that $\mathit{time}(\mathbf{W}) \subseteq \{t, t+1\}$.

2. Similarly, if X^{t} and Y^{t+1} are conditionally independent given some set $\mathbf{Z}$, then there exists a separating set $\mathbf{W}$ such that $\mathit{time}(\mathbf{W}) \subseteq \{t, t+1\}$.

Proof. For each case, we will construct an appropriate separating set:

1. Let $\mathbf{W} = Pa(X^{t+1}) \cup Pa(Y^{t+1})$. Because of the first-order Markov assumption, $\mathit{time}(\mathbf{W}) \subseteq \{t, t+1\}$. Moreover, conditioning of $\mathbf{W}$ renders X^{t+1}, Y^{t+1} independent because of the local Markov property (either X^{t+1} is a descendant of Y^{t+1} or vice versa, or none of them is a descendant of the either).

2. In this case, let $\mathbf{W} = Pa(Y^{t+1})$. Because of the temporal semantics, X^{t} is a non-descendant of Y^{t+1}; therefore, Y^{t+1} is conditionally independent of its non-descendants (that include X^{t}) given $\mathbf{W}$.

□

The significance of the above proposition is that, given a model where the first-order Markov assumption holds, we could infer conditional independencies (and use them to learn the structure of the model) by only considering consecutive time points.

As shown by Maier, Marazopoulou, and Jensen [50], d-separation cannot be applied directly to a relational model. To correct for that, we introduced *relational d-separation*, a graphical criterion that can be applied to abstract ground graphs and used to infer conditional independencies that hold across all possible ground graphs of the model. Here, we show that the notion of relational d-separation can be generalized for temporal abstract ground graphs as well. In the following definition, $X|_b$ denotes the terminal set of X starting at b, i.e., the set of X instances that can be reached if we start from instance b and follow the relational path of X on the relational skeleton.

Definition 6.13 (Temporal relational d-separation)**.** Let $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ be three disjoint sets of temporal relational variables from perspective B for a 2-slice temporal relational model $\mathcal{M}$ such that not both $\mathbf{X}$ and $\mathbf{Y}$ contain variables in t. Then $\mathbf{X}$ and $\mathbf{Y}$ are d-separated by $\mathbf{Z}$ if and only if, for any temporal skeleton σ_T, $\mathbf{X}|_b$ and $\mathbf{Y}|_b$ are d-separated by $\mathbf{Z}|_b$ in ground graph $GG_{\mathcal{M}\sigma_T}$ for all $b \in \sigma_T(B)$.

The following theorem shows that temporal relational d-separation is sound and complete up to a specified hop threshold. We use the notation $\bar{\mathbf{X}}$ to denote the set of relational variables $\mathbf{X}$ augmented with the set of intersection variables they participate in.

Theorem 6.14. Let $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ be three disjoint sets of temporal relational variables for perspective B such that not both $\mathbf{X}$ and $\mathbf{Y}$ contain variables in t. Then, for any temporal skeleton σ_T and for all $b \in \sigma(B)$, $\mathbf{X}|_b$ and $\mathbf{Y}|_b$ are d-separated by $\mathbf{Z}|_b$ up to h in ground graph $GG_{\mathcal{M}\sigma_T}$ if and only if $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ are d-separated by $\bar{\mathbf{Z}}$ on the abstract ground graph $tAGG_{\mathcal{M}Bh}$.

Proof. We prove this by defining a non-temporal relational model that is equivalent to the given temporal model. Since d-separation is sound and complete for non-temporal relational models, the result extends to the equivalent model, and, therefore, the temporal relational model. Given a 2-slice temporal relational model $\mathcal{M}_T = \langle \mathcal{S}, \mathcal{D}_T \rangle$, construct a relational model $\mathcal{M} = \langle \mathcal{S}', \mathcal{D} \rangle$ as follows:

- For every item in $\mathcal{S}$, add an item in $\mathcal{S}'$ with superscript t and one with superscript $t+1$: $\mathcal{S}' = \{I^t, I^{t+1} \mid I \in \mathcal{S}\}$.

- For every entity $E \in \mathcal{S}$, add in $\mathcal{S}'$ a 1-1 relation between E^t and E^{t+1}. This part allows the "temporal jumps" between entities.

- For every relation $R \in \mathcal{S}$, add in $\mathcal{S}'$ a path from R^t to R^{t+1} as follows; Add two "dummy" entities E_d^t, E_d^{t+1} and 1-1 relations between R^t and R^{t+1}. This part allows the "temporal jumps" between relations. Note that this transformation changes the arity of the original relation R, but preserves the path semantics we are trying to capture.

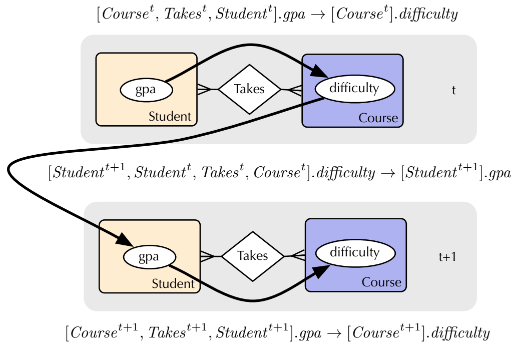
- The set of relational dependencies is the set of temporal relational dependencies: $\mathcal{D} = \mathcal{D}_T$.

Figure 6.8 shows the process for transforming a temporal model to an equivalent non-temporal model. From the definition of temporal relational paths, it can be shown that these two models are equivalent in the sense that there is a one-to-one correspondence between valid paths in $\mathcal{M}_T$ and $\mathcal{M}$. Leveraging the temporal constraints of d-separation described by Proposition 6.12 and the soundness and completeness of d-separation for abstract ground graphs, we conclude that d-separation is sound and complete (up to a hop threshold) in temporal abstract ground graphs.

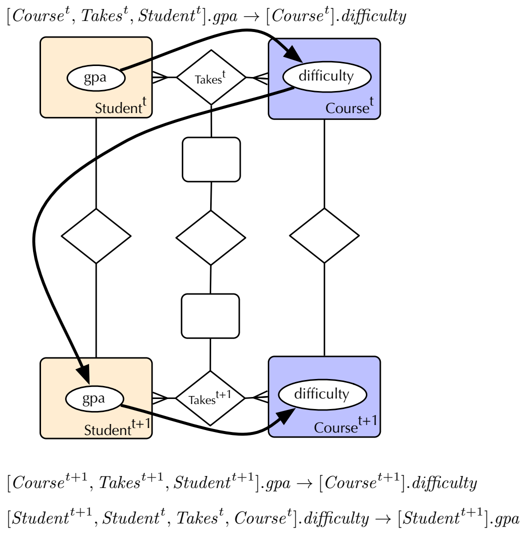
$\square$

6.7 Concluding Remarks

In this chapter, we presented a formalization of temporal relational models. Our representation models time as a discrete quantity and can express temporal relational


$[Course^t, Takes^t, Student^t].gpa \rightarrow [Course^t].difficulty$
gpa
Student
Takes
difficulty
Course
t
$[Student^{t+1}, Student^t, Takes^t, Course^t].difficulty \rightarrow [Student^{t+1}].gpa$
gpa
Student
Takes
difficulty
Course
t+1
$[Course^{t+1}, Takes^{t+1}, Student^{t+1}].gpa \rightarrow [Course^{t+1}].difficulty$

(a) Temporal model $\mathcal{M}_T$


$[Course^t, Takes^t, Student^t].gpa \rightarrow [Course^t].difficulty$
gpa
$Student^t$
$Takes^t$
difficulty
$Course^t$
gpa
$Student^{t+1}$
$Takes^{t+1}$
difficulty
$Course^{t+1}$
$[Course^{t+1}, Takes^{t+1}, Student^{t+1}].gpa \rightarrow [Course^{t+1}].difficulty$
$[Student^{t+1}, Student^t, Takes^t, Course^t].difficulty \rightarrow [Student^{t+1}].gpa$

(b) Transformed non-temporal model $\mathcal{M}$

Figure 6.8: Temporal relational model 6.8a and equivalent non-temporal relational model 6.8b. The transformed model contains dummy entity and relationship classes in order to represent the temporal paths as defined for the temporal relational schema.

dependencies across time points. This is an expressive unifying representation that subsumes Bayesian networks, dynamic Bayesian networks, and the non-temporal relational model. We modified the definition of abstract ground graphs for the case of temporal relational models. Moreover, we showed that the theory of relational d-separation can be extended for the class of temporal relational models. The theory of temporal relational d-separation together with the framework of temporal abstract ground graphs lay the groundwork for estimating causal effects and designing experiments in temporal relational domains.

CHAPTER 7

LEARNING THE STRUCTURE OF TEMPORAL RELATIONAL MODELS

The theory of temporal relational d-separation allows us to derive all conditional independence facts that are consistent with the structure of a temporal relational model, the same way that d-separation connects the structure of a Bayesian network and the conditional independencies of the underlying distribution. This is precisely the connection that constraint-based algorithms leverage in order to learn the structure of models. Thus, temporal relational d-separation (and temporal abstract ground graphs) enable a constraint-based algorithm, TRCD, that learns the structure of temporal and relational causal models from data.[1]

In this chapter, we first describe RCD, the first sound and complete constraint-based algorithm to learn the structure of relational models, that serves as the basis for TRCD. We then present the TRCD algorithm along with an experimental evaluation of its performance in synthetically generated domains.

7.1 Relational Causal Discovery Algorithm

The relational causal discovery algorithm (RCD) takes as input a relational schema, a populated relational database consistent with that schema, and a domain specific threshold for the maximum length of relational dependencies to consider. The output of RCD is a set of relational dependencies. RCD is a constraint-based algorithm and

[1]Marazopoulou, Maier, Jensen. Learning the Structure of Causal Models with Relational and Temporal Dependence. Proceedings of the 31st Conference in Artificial Intelligence (2015) [54]

operates in two phases. The first phase learns a set of unoriented dependencies. The second phase leverages a set of orientation rules to orient as many of the dependencies as possible.

7.1.1 Phase I of RCD

RCD starts by constructing the set of all potential relational dependencies in canonical form with length up to the specified threshold. For every relational dependency in the set, its reverse is also included. Therefore, this set effectively contains unoriented dependencies. In the relational setting, reversing a dependency in canonical form boils down to reversing the path of the treatment relational variable. Consider, for example, the following relational dependency and its reverse:

$$[I_1, \ldots, I_n].A_n \rightarrow [I_1].A_1$$
$$[I_n, \ldots, I_1].A_1 \rightarrow [I_n].A_n$$

For every relational dependency in the set of potential dependencies, a series of conditional independence tests is performed with increasing size of conditioning set. The goal is to find a set of relational variables that will make the treatment and outcome relational variables of the dependency independent. If two relational variables are found to be conditionally independent, the separating set is recorded and the relational dependency is removed from the set of potential dependencies along with its reverse.

For example, for the relational dependency $[I_1, \ldots, I_n].A_n \rightarrow [I_1].A_1$ the conditional independence queries are of the form:

$$[I_1, \ldots, I_n].A_n \perp\!\!\!\perp [I_1].A_1 \mid \mathbf{Z}$$

where $\mathbf{Z}$ is a set of relational variables with base item I_1. Note that a conditional independence query in the relational setting is among relational variables with a common base item. Specifically for the purposes of RCD, at least one of the treatment/outcome query variables has a path of length one, since RCD is operating over dependencies in canonical form.

7.1.2 Phase II of RCD

The second phase of RCD starts with the output of phase I, i.e., a set of pairs of relational dependencies (a relational dependence and its reverse). It employs a set of deterministic orientation rules to orient dependencies. The orientation rules operate on the level of abstract ground graphs. Therefore, phase II of RCD starts by constructing all abstract ground graphs (from all perspectives) for the set of dependencies produced by phase I. Whenever a relational dependency is oriented in an abstract ground graph, the orientation is propagated within that abstract ground graph and across all abstract ground graphs.

The orientation rules are similar to the ones used in the propositional case, with the addition of one new rule, unique to the relational case: Relational bivariate orientation (RBO). A schematic representation of the orientation rules is shown in Figure 7.1. These orientation rules are sound and complete assuming the set of dependencies they operate on is correct. However, since the rules are deterministic, if the set of dependencies is mis-specified, errors may be introduced and propagated. This is a characteristic of all constraint-based algorithms and is one reason that hybrid algorithms where introduced.

7.1.3 Properties of RCD

The outcome of RCD is a partially oriented set of dependencies (i.e., a partially oriented model). This corresponds to the relational Markov equivalence class of the true model. Note that even though this is the relational Markov equivalence class, no characterization is yet known with respect to the structure of the model.

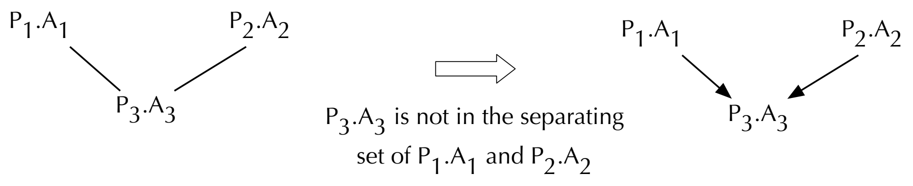
Maier, Marazopoulou, and Jensen [49] showed that RCD is sound and complete assuming perfect conditional independence tests, infinite threshold, the causal Markov condition holds, faithfulness, and causal sufficiency for relational models. The correctness of phase I stems from the correctness of abstract ground graphs and of the theory of relational d-separation. The correctness of phase II, is due to the soundness and completeness of the orientation rules.

RCD is the relational counterpart of PC, a propositional constraint-based algorithm. A major difference of RCD compared to PC is that RCD operates over *a set* of abstract ground graphs. In other words, RCD is reasoning simultaneously on abstract ground graphs constructed from all possible perspectives.

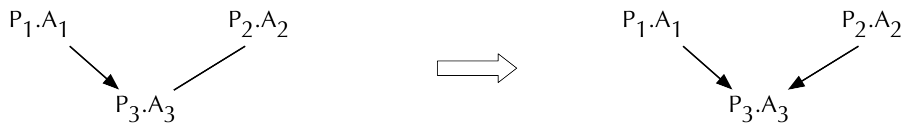
7.2 Temporal Relational Causal Discovery Algorithm

TRCD extends RCD [49] to operate over a 2-slice temporal relational model. More specifically, it constructs a set of temporal abstract ground graphs, one from the perspective of each item class, and uses the theory of temporal relational d-separation on temporal abstract ground graphs to decide conditional independence facts. TRCD uses the temporal abstract ground graphs to determine which conditional independence facts should be checked in the data and which dependencies (edges) in the temporal relational model are implied by those facts. As in the case of RCD, for practical reasons, the space of potential dependencies is limited by a domain-specific hop threshold.

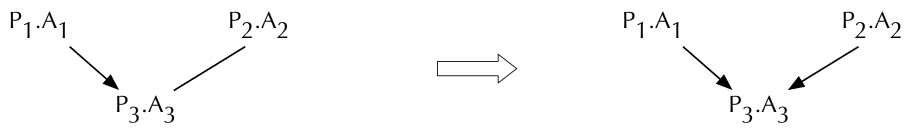
TRCD operates in two phases. Phase I learns a set of undirected dependencies and Phase II employs a set of orientation rules to orient those dependencies. Phase II of TRCD uses the same orientation rules as RCD—collider detection, known noncolliders (KNC), cycle avoidance (CA), Meek rule 3 (MR3), and relational bivariate



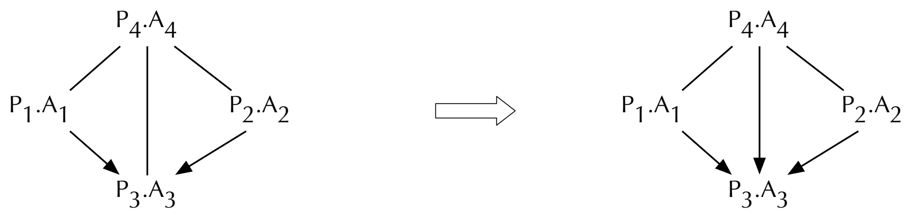
(a) Collider detection.



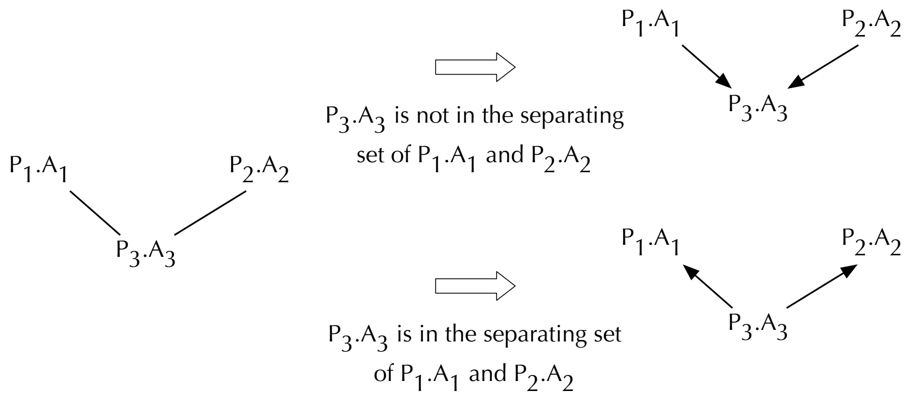
(b) Known non-colliders (KNC).



(c) Cycle avoidance (CA).



(d) Meek rule 3 (MR3).


P1.A1
P2.A2
P3.A3
P3.A3 is not in the separating set of P1.A1 and P2.A2
P1.A1
P2.A2
P3.A3
P3.A3 is in the separating set of P1.A1 and P2.A2
P1.A1
P2.A2
P3.A3

(e) Bivariate edge orientation (BEO).

Figure 7.1: The orientation rules used for RCD (and TRCD). $P_i.A_i$, $i \in \{1, \dots, 4\}$ are relational variables with a common base item.

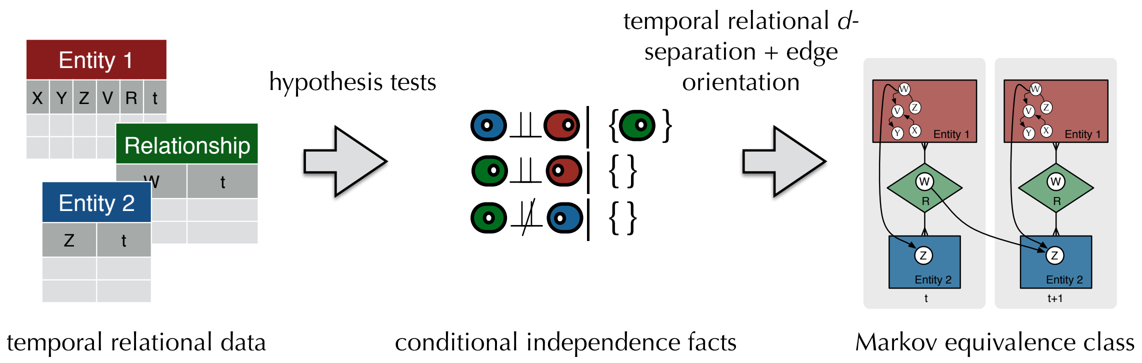


Figure 7.2: Schematic representation of how TRCD works. The input of the algorithm is a data set in the form of a relational database with timestamps. Through hypothesis tests TRCD infers a set of conditional independence facts that hold in the data. Then, through the use of temporal relational d-separation and orientation rules, the algorithm learns the structure of the temporal relational model, up to the (temporal relational) Markov equivalence class.

orientation (RBO). However, TRCD applies these rules on temporal abstract ground graphs, as opposed to abstract ground graphs in the case of RCD. Additionally, TRCD orients dependencies that cross time points from the past to the future. A schematic representation of how the algorithm works is shown in Figure 7.2.

RCD was shown to be sound and complete in the sample limit (i.e., with perfect tests of conditional independence) and for infinite hop threshold, under the standard assumptions of the causal Markov condition, faithfulness, and causal sufficiency for relational domains. By leveraging the soundness and completeness of temporal relational d-separation, TRCD can be shown to be sound and complete (under the same assumptions).

7.3 Experimental Evaluation of TRCD

In order to evaluate the performance of TRCD, we performed experiments on synthetically generated domains. The first set of experiments relies on an oracle for conditional independence. This simulates the behaviour of the algorithm with perfect tests of conditional independence. In the second set of experiments, actual conditional independence tests are performed.

7.3.1 Oracle Experiments

The goal of this set of experiments is twofold. First, we evaluate the theoretical performance of TRCD in the large sample limit. Towards that end, we employ an oracle for answering d-separation queries in the place of statistical tests of independence. Second, these experiments provide experimental evidence about the correctness of temporal relational d-separation.

We generated synthetic schemas with number of entities varying from 1 to 3, number of relationships fixed to one less than the number of entities, and number of attributes for each item drawn from $Poisson(\lambda = 1) + 1$. The cardinalities were uniformly selected at random. For each number of entities specified, we generated 500 different schemas. This generating process yielded 1,500 different schemas. For a given schema, we generated 10 different models with number of dependencies ranging from 1 to 10. Specifically, we generated all possible dependencies, up to hop-threshold $h = 3$. Then, we chose the desired number of dependencies from that space at random, subject to the following constraints: Each relational variable has at most 3 parents and the generated model contains no cycles. Moreover, every model must contain at least one dependency with a temporal relational path that contains more than one time point. This procedure resulted in a total of 15,000 models.

We ran TRCD with a d-separation oracle for each model. We evaluate the performance of the algorithm by computing precision and recall, both after Phase I (on the unoriented model) and after Phase II (on the partially oriented model). Specifically, the metrics are computed as follows:

$$\begin{aligned}
skeletonPrecision &= \frac{\text{number of true edges learned}}{\text{number of learned edges}} \\
skeletonRecall &= \frac{\text{number of true edges learned}}{\text{number of true edges}} \\
orientedPrecision &= \frac{\text{number of learned orientations that match the true orientations}}{\text{number of learned orientations}} \\
orientedRecall &= \frac{\text{number of learned orientations that match the true orientations}}{\text{number of true orientations}}
\end{aligned}$$

Figure 7.3 shows average precision and recall after Phase I (unoriented dependencies) and after Phase II (partially oriented dependencies). As expected, given that these experiments use an oracle, the algorithm always learns the correct set of unoriented dependencies (unoriented precision and recall are always 1). The algorithm makes no mistakes in the orientation (oriented precision is always 1); however, it is not possible to orient all dependencies (oriented recall is lower than 1). This is to be expected since TRCD recovers the structure up to the relational Markov equivalence class. Therefore, some edges are expected to be left unoriented. Since this experiment characterizes the theoretical performance of TRCD with perfect conditional independence tests, the line corresponding to oriented recall implicitly describes the size of the Markov equivalence class for temporal relational models.

Note that comparing to an oracle version of RCD is not straightforward. The oracle requires a true model that is fully directed. Converting the true temporal model to a non-temporal one would often result in cycles or undirected edges, and the relational d-separation oracle cannot be used.

Finally, we measured how often each of the orientation rules shown in Figure 7.1 is used in Phase II of TRCD. The results are shown in Table 7.1. Collider detection, known non-colliders (KNC), and relational bivariate orientation (RBO) are orienting the majority of the edges. As expected, in the case of propositional models (one entity), the relational bivariate orientation rule, a rule for edge orientation that is unique to relational data, is never used. We observe that the other two rules—

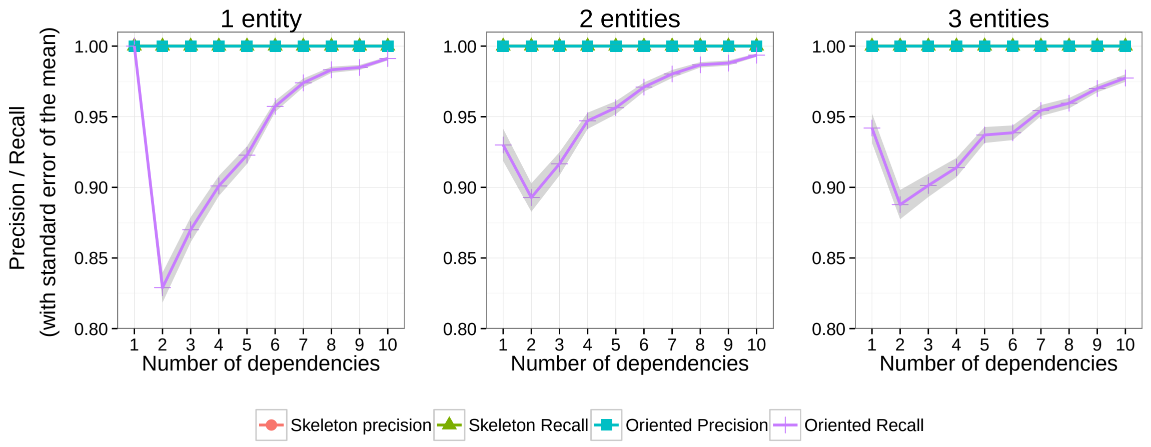


Figure 7.3: Precision and recall for TRCD after Phase I (unoriented) and after Phase II (oriented) when using an oracle for answering d-separation queries. Note that the y-axis values start at 0.8.

cycle avoidance and Meek's rule 3—do not fire often. That can be explained by the fact that those rules would never fire in the presence of a temporal edge. Consider cycle avoidance in the propositional case: If the pattern $X \rightarrow Y \rightarrow Z$ and $X - Y$ is encountered, then cycle avoidance orients $X \rightarrow Y$. If any of the dependencies $X \rightarrow Y \rightarrow Z$ crosses time points, then X and Y would be in different time points and the edge between them would be oriented based on temporal precedence. A similar argument can be made for the case of Meek's rule 3.

7.3.2 Experiments on Synthetic Data

In the previous set of experiments, TRCD was given access to an oracle for the conditional independence tests and the experimental results demonstrate the behavior of the algorithm under perfect tests of conditional independence. However, in practice, conditional independence tests are imperfect and often conclude dependence when there is none (type I errors) or the reverse (type II errors). This set of experiments showcases the use of TRCD on a more realistic setting, i.e., using synthetic data,

Table 7.1: Frequency of the most commonly used orientation rules during Phase II of TRCD for the oracle experiments. For the rest of the rules, the frequency was less than 1% and they are omitted from this table. Temporal dependencies are not explicitly oriented by the orientation rules (they are always oriented based on temporal information: from the past to the future).

Number of entities	Collider detection	KNC	RBO	Percent of temporal dependencies
1	71%	28%	0%	66%
2	66%	11%	23%	68%
3	53%	11%	36%	65%

without the use of an oracle to decide conditional independence. In this case, actual—and therefore imperfect—tests of conditional independence are performed on the data.

The experiments were performed on synthetic models and synthetic data generated from these models. The use of synthetic data allows us to have access to the ground truth, so we can measure the accuracy of the learned models. The data-generating process is described in detail below.

Using the same process as described in 7.3.1, we generated 5 synthetic schemas with 2 entities and 5 synthetic schemas with 3 entities. For each schema, we generated 10 models with number of dependencies ranging from 1 to 10. This resulted in 100 different models. For each model we created 3 different relational skeletons over 300 timepoints. The number of entity instances at each time point was drawn from $Poisson(\lambda) + 1$, where $\lambda \sim \mathcal{U}(5, 10)$. The degree distribution for the relationship instances was drawn from $Poisson(\lambda) + 1$, where $\lambda \sim \mathcal{U}(1, 5)$. Regarding the parameters of the graphical model, the marginals were parameterized as normal distributions $\mathcal{N}(\mu, \sigma)$, where $\mu \sim \mathcal{U}(0, 5)$ and $\sigma \sim \mathcal{U}(0, 1)$. The conditional distribution for a relational variable X was

$$X \sim \sum_{Y \in Pa(X)} \big(avg(Y)\big) + 0.1 * \mathcal{N}(0, 1).$$

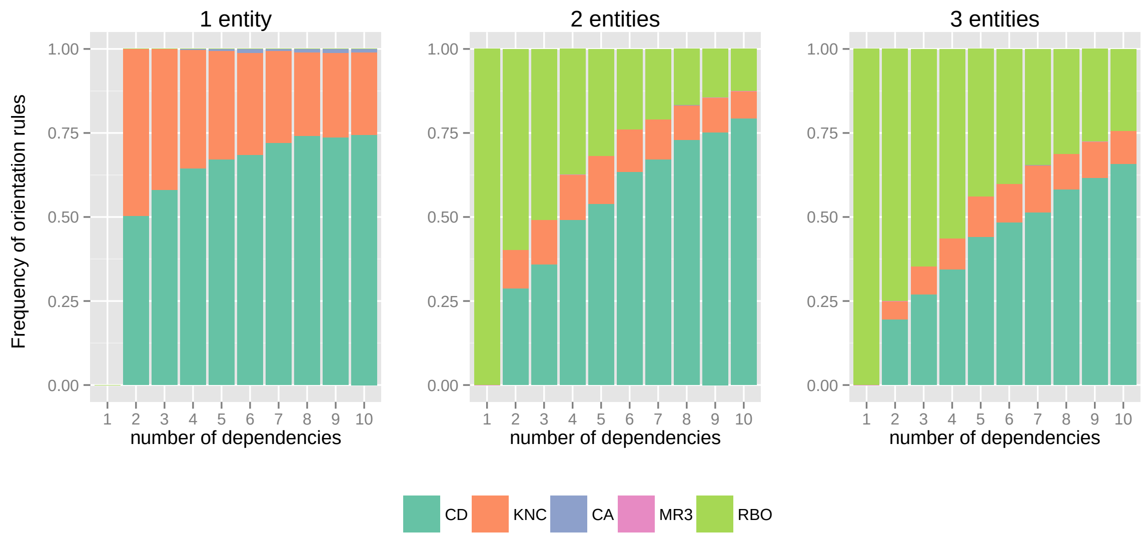


Figure 7.4: Frequency of orientation rules of TRCd for synthetic domains of varying complexity.

This resulted in 300 data sets, each over 300 time points.

In order to assess statistical independence, we fitted a standard linear least-squares regression equation to the outcome variable using the treatment and the variables in the conditioning set as covariates. For relational variables in the conditioning set, we used the average as the aggregation function. Then, we directly used the t-test of the coefficient of the treatment variable to assess independence ($p > 0.05$ or effect size < 0.01). Figure 7.5 shows average precision and recall of TRCD after Phase I and Phase II, when applied to the synthetic data sets. While precision after Phase I is more than 0.75 in most cases, the recall after Phase I is relatively low. That implies that we concluded independence (and therefore we removed an edge) more often than we should. This corresponds to Type II errors and can be attributed to the lack of good conditional independence tests for relational data and/or low power of the conditional independence tests.

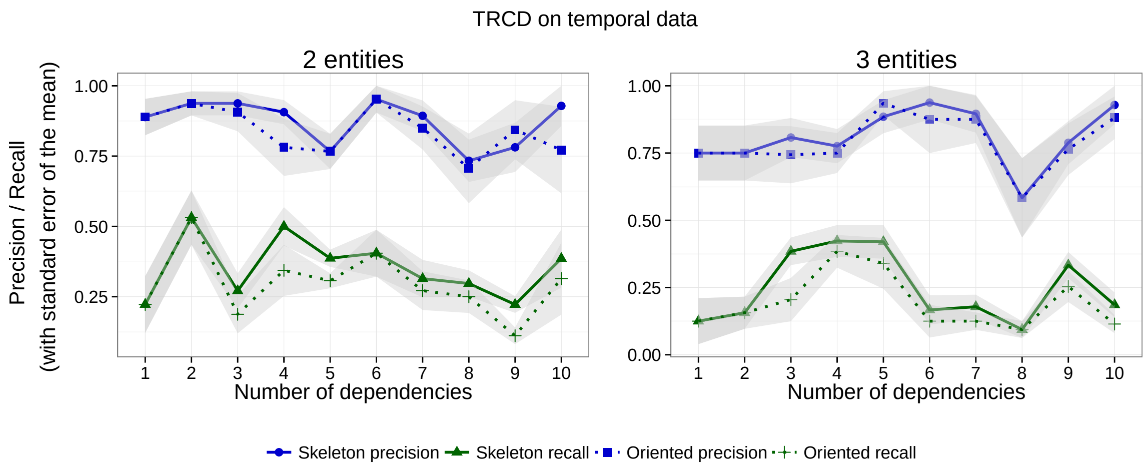


Figure 7.5: TRCD on synthetic temporal data.

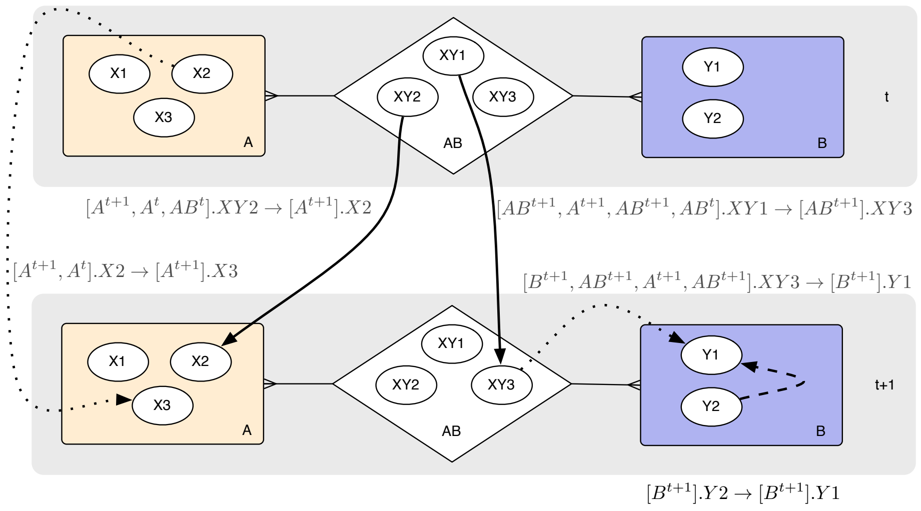
7.3.3 Comparing TRCD to RCD

Finally, to demonstrate the difference in expressiveness between TRCD and RCD (the only constraint-based algorithm for relational data), we ran RCD on a "temporally flattened" version of the synthetic data. Specifically, RCD is ignoring temporal information and an instance is uniquely identified by instance id and time point.

The true model and the model that TRCD learned are shown in Figure 7.6. The model learned by RCD is shown in Figure 7.7. TRCD correctly learns and orients two of the edges, with the correct path specification. Those dependencies cannot even be expressed in the space of dependencies for RCD.

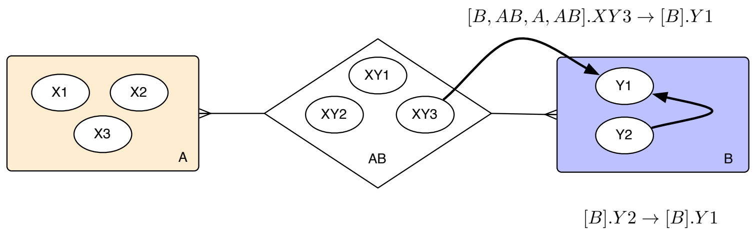
7.4 Concluding remarks

In this chapter, we presented a constraint-based algorithm, TRCD, that learns the structure of temporal relational models from data. TRCD leverages the theory of temporal relational d-separation and the framework of temporal abstract ground graphs. We showed that the algorithm is sound and complete under certain standard assumptions (perfect conditional independence test, infinite threshold for the length of dependencies, causal Markov condition, causal sufficiency, faithfulness). We



X1
X2
X3
A
XY1
XY2
XY3
AB
Y1
Y2
B
t
$[A^{t+1}, A^{t}, AB^{t}].XY2 \rightarrow [A^{t+1}].X2$
$[AB^{t+1}, A^{t+1}, AB^{t+1}, AB^{t}].XY1 \rightarrow [AB^{t+1}].XY3$
$[A^{t+1}, A^{t}].X2 \rightarrow [A^{t+1}].X3$
$[B^{t+1}, AB^{t+1}, A^{t+1}, AB^{t+1}].XY3 \rightarrow [B^{t+1}].Y1$
X1
X2
X3
A
XY1
XY2
XY3
AB
Y1
Y2
B
t+1
$[B^{t+1}].Y2 \rightarrow [B^{t+1}].Y1$

Figure 7.6: True temporal relational model. Solid edges were successfully learned by TRCD, dotted edges were not successfully learned, and the dashed edge was left unoriented.



$[B, AB, A, AB].XY3 \rightarrow [B].Y1$
X1
X2
X3
A
XY1
XY2
XY3
AB
Y1
Y2
B
$[B].Y2 \rightarrow [B].Y1$

Figure 7.7: Model learned by RCD for data generated from the temporal model of Figure 7.6. Some of the dependencies of the true temporal model cannot be expressed in the space of dependencies for RCD.

provided experimental evidence that showcases the correctness of TRCD when the aforementioned assumptions are met. For this set of experiments, TRCD was given access to an oracle for conditional independence testing. Moreover, we evaluated TRCD in synthetically generated data, using conditional independence tests. This set of results suggests there is room for improvement, especially regarding the use of independence tests in temporal/relational settings. Finally, we showed the improvement that TRCD achieves compared to RCD when applied to domains with a temporal component.

CHAPTER 8
CONCLUSIONS AND FUTURE WORK

Causality is a fundamental tool for researchers and practitioners in a wide array of fields, ranging from engineering to social and political sciences. The vast majority of tools that have been developed for causal inference and causal discovery rely on the assumption of i.i.d. data. However, many real-world domains are inherently more complex, consisting of multiple interacting entities (for example social networks). Moreover, the behavior and the structure of such systems evolves over time.

In this thesis, we focus on developing graphical models for temporal and relational domains and studying their properties. These expressive classes of models allow us to represent complex systems in a more accurate and intuitive way. Moreover, they enable reasoning about complex processes such as interventions in dynamic networks, and lay the groundwork for experimental designs in dynamic relational domains.

8.1 Summary of Contributions

The contributions of this thesis can be divided into two categories. First, an in-depth investigation of the semantics of relational models. Specifically, we show the impact that alternative grounding semantics have for various tasks of interest (feature construction, model fit, estimation of causal effects) under two different scenarios. Chapter 4 focuses on the case of networks with one type of relationship with probabilistic dependencies between a node and nodes in its extended neighborhood [52]. Chapter 5, focuses on the case of heterogeneous networks with two types of relationships and probabilistic dependencies from a node's immediate peers [53]. In both scenarios, grounding semantics significantly affect terminal sets (i.e., feature construction). For the second case, our results suggest that model mis-specification may significantly bias the estimated average treatment effect. Finally, model selection techniques can be used to learn the semantics that are a better fit for the given data.

The second set of contributions is centered around extending the expressiveness of existing relational graphical models to include a temporal dimension [54]. In chapter 6, we present a formalization of (discrete time) temporal relational models, an expressive class of models that can capture probabilistic dependencies between variables on different types of entities within and across time points. We extend the notion of abstract ground graphs [50]—a lifted representation that allows reasoning about the conditional independencies implied by a relational model—to the case of temporal relational models, and we show that temporal abstract ground graphs are a sound and complete abstraction for ground graphs of temporal relational models. Temporal abstract ground graphs can be used to answer d-separation queries for temporal relational models. Finally, in Chapter 7, we extend an existing constraint-based algorithm for inferring causal dependence in relational data—the relational causal discovery (RCD) algorithm—to incorporate time, thus providing a constraint-based method that learns causal models from temporal relational data. We show that the temporal relational causal discovery (TRCD) algorithm is sound and complete under certain standard assumptions.

8.2 Future Work

There are several promising directions for future work. These include extensions of the work presented in this thesis, such as lifting some of the simplifying assumptions. Moreover, the contributions of this thesis provide a first step towards some more long-term research goals, such as automated causal discovery in complex systems. In what follows, we describe interesting directions for future work, grouped thematically.

There is a number of interesting open problems with respect to properties of the relational model described in Chapter 3. These include the following:

- *Formalizing the notion of relational Markov equivalence classes.* Lee et al. [45] provided a characterization of Markov equivalence classes for relational models under path semantics. No characterization of relational Markov equivalence classes is known under the bridge burning semantics, i.e., for the relational model as described in Chapter 3. This leaves a couple of questions unanswered: Is the characterization of relational Markov equivalence class under bridge burning semantics identical to the one under path semantics? More generally, do relational Markov equivalence classes depend on the semantics assumed for relational dependence?

- *Characterizing the identifiability of relational models.* Arbour, Marazopoulou, and Jensen [3] have shown that the direction of relational dependence can be identified in the bivariate relational case for deterministic (noiseless) dependence and for regular graphs. This is an indication that relational models provide more information (in a sense, they are more identifiable, or they form smaller equivalence classes) compared to propositional models. Future work could extend the proofs of Arbour et al. to fully relational domains, with arbitrary skeletons, and noisy dependence.

- *Formalizing do-calculus for relational models.* The theory of relational d-separation together with the representation of abstract ground graphs provide the necessary machinery for defining the equivalent of *do*-calculus in relational settings. This would provide a principled framework to reason about the effects of interventions in relational domains and to characterize causal effects that are identifiable from observational data in relational settings. Arbour et al. [2] make a first step towards that direction. Specifically, they introduce relational covariate adjustment (RCA), a method that leverages the theory of relational d-separation to identify adjustments sets for estimation of causal effects in relational domains. Their method achieves accuracy similar to that of experimental methods.

All of the above directions can also be applied to the more expressive class of temporal relational models. Specific to the temporal relational model, interesting directions for future work include:

- *Characterizing temporal relational interventions.* A perfect intervention in the propositional setting is well-defined. It consists of changing a variable and setting its value to some constant. In the temporal and relational case, interventions are more complex and, to the best of our knowledge, not fully explored or formally defined. For example, consider a dynamic system that changes over time. What consists an intervention in this setting? Setting a variable to a constant value forever? Or for a certain amount of time? These are both valid and potentially useful interpretations of a temporal intervention, depending on the domain of interest.

With respect to constraint-based structure learning algorithms:

- *Improving test of conditional independence for temporal/relational data.* All constraint-based algorithms rely heavily on tests of conditional independence. In this thesis, we use a naive implementation of conditional independence tests for temporal relational domains and the relatively poor performance of TRCD on synthetic data reflects this choice. Conditional independence testing for non-i.i.d. data is an active area of research. To the best of our knowledge, there does not exists a general test of conditional independence that can handle an arbitrary relational structure and temporal dependence. Zhang et al. [93] extend the Hilbert-Schidt independence criterion (HSIC) [30] to structured domains that can be decomposed in cliques. This, however, cannot be applied to networks with a lattice structure and is a test of marginal, not conditional, dependence. Similarly, Chwialkowski et al. [14] extend HSIC for random processes, again in the marginal setting. Moneta et al. [57] provide an conditional independence test for time series data modeled as a VAR (vector autoregressive) process. This test accounts for temporal dependence, but cannot account for arbitrary relational structure. Finally, Flaxman et al. [23] propose a conditional independence test based on Gaussian process regression.

- *Developing better evaluation methods for causal structure learning algorithms.* For evaluation of structure learning algorithms the standard practice is to use synthetic models. The output of the structure learning algorithm, i.e., the learned model, is compared to the true synthetic model and various accuracy metrics can be computed on the model structure. These metrics include precision and recall (as we do in this thesis), structural Hamming distance [81], and structural intervention distance [63]. However, Garant and Jensen [26] showed that these metrics are not necessarily optimal for evaluating the accuracy of estimated causal effects. Instead, they propose an evaluation method based on interventional distributions and their discrepancies, taking into account in the evaluation both the quality of the structure and of the parameters. Their methodology is applicable to DAGs and they provide benchmarks for empirical testing of causal discovery algorithms on real propositional data. Their work could be extended to the case of temporal relational models.

TRCD makes certain simplifying assumptions. Future work could focus on relaxing some of those assumptions.

- *Relaxing the stationarity assumption.* TRCD assumes that the structure and parameters of the model are stationary and do not change over time. A common way around this is to use change point detection to infer when the model structure or model parameters have changed enough. This will require change-point detection techniques for relational data.

- *Relaxing the assumption of causal sufficiency.* Another avenue for future research is relaxing the causal sufficiency assumption by employing techniques such as blocking [66] for temporal relational domains. A different way to relax causal sufficiency would be to significantly extend the representation of relational models. This would be equivalent to the extension of DAGs to maximal ancestral graphs (MAGs) [69] in the propositional setting, a graphical model that is closed under marginalization and conditioning. For propositional data, there exists a constraint-based algorithm, FCI algorithm [76], that learns the Markov equivalence class of MAGs from data.

BIBLIOGRAPHY

[1] Aral, Sinan, and Walker, Dylan. Identifying influential and susceptible members of social networks. *Science 337*, 6092 (2012), 337–341.

[2] Arbour, David, Garant, Dan, and Jensen, David. Inferring network effects from observational data. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2016), pp. 715–724.

[3] Arbour, David, Marazopoulou, Katerina, and Jensen, David. Inferring causal direction from relational data. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence* (2016).

[4] Bakshy, Eytan, Rosenn, Itamar, Marlow, Cameron, and Adamic, Lada. The role of social networks in information diffusion. In *Proceedings of the 21st international conference on World Wide Web* (2012), ACM, pp. 519–528.

[5] Banerjee, Abhijit, Chandrasekhar, Arun G., Duflo, Esther, and Jackson, Matthew O. The diffusion of microfinance. *Science 341*, 6144 (2013).

[6] Banerjee, Abhijit, Chandrasekhar, Arun G., Duflo, Esther, and Jackson, Matthew O. The diffusion of microfinance. `http://hdl.handle.net/1902.1/21538`, 2014.

[7] Barabási, Albert-László, and Albert, Réka. Emergence of scaling in random networks. *Science 286*, 5439 (1999), 509–512.

[8] Blumen, Isadore, Kogan, Marvin, and McCarthy, Philip J. The industrial mobility of labor as a probability process. *Cornell Studies of Industrial and Labor Relations 6* (1955).

[9] Chickering, David Maxwell. Optimal structure identification with greedy search. *Journal of Machine Learning Research 3* (2003), 507–554.

[10] Chickering, David Maxwell, Geiger, Dan, and Heckerman, David. Learning bayesian networks is NP-hard (technical report MSR-TR-94-17). *Redmond, WA: Microsoft Research* (1994).

[11] Choi, David S. Estimation of monotone treatment effects in network experiments. *arXiv preprint arXiv:1408.4102* (2014).

[12] Christakis, Nicholas, and Fowler, James. *Connected: The surprising power of our social networks and how they shape our lives.* Hachette Digital, Inc., 2009.

[13] Christakis, Nicholas A., and Fowler, James H. The spread of obesity in a large social network over 32 years. *New England Journal of Medicine 357*, 4 (2007), 370–379.

[14] Chwialkowski, Kacper, and Gretton, Arthur. A kernel independence test for random processes. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)* (2014), pp. 1422–1430.

[15] Dean, Thomas, and Kanazawa, Keiji. A model for reasoning about persistence and causation. *Computational Intelligence 5*, 2 (1989), 142–150.

[16] Dong, Yuxiao, Yang, Yang, Tang, Jie, Yang, Yang, and Chawla, Nitesh V. Inferring user demographics and social strategies in mobile social networks. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2014), pp. 15–24.

[17] Eagle, Nathan, and Pentland, Alex (Sandy). Reality mining: Sensing complex social systems. *Personal and Ubiquitous Computing 10*, 4 (Mar. 2006), 255–268.

[18] Eichler, Michael. Graphical modelling of multivariate time series. *Probability Theory and Related Fields 153*, 1-2 (2012), 233–268.

[19] Entner, Doris, and Hoyer, Patrik. On causal discovery from time series data using FCI. In *Proceedings of the 5th European Workshop on Probabilistic Graphical Models (PGM-2010)* (2010), pp. 121–128.

[20] Erdös, Paul, and Rényi, Alfréd. On random graphs, I. *Publicationes Mathematicae (Debrecen) 6* (1959), 290–297.

[21] Fakhraei, Shobeir, Foulds, James, Shashanka, Madhusudana, and Getoor, Lise. Collective spammer detection in evolving multi-relational social networks. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2015), KDD '15, pp. 1769–1778.

[22] Fan, Yu, and Shelton, Christian R. Learning continuous-time social network dynamics. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence* (2009), pp. 161–168.

[23] Flaxman, Seth R., Neill, Daniel B., and Smola, Alexander J. Gaussian processes for independence tests with non-iid data in causal inference. *ACM TIST 7*, 2 (2016), 22–1.

[24] Friedman, N, Getoor, L, Koller, D, and Pfeffer, A. Learning probabilistic relational models. *International Joint Conference on Artificial Intelligence 16* (1999), 1300–1309.

[25] Friedman, Nir, Murphy, Kevin, and Russell, Stuart. Learning the structure of dynamic probabilistic networks. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence* (1998), pp. 139–147.

[26] Garant, Dan, and Jensen, David. Evaluating causal models by comparing interventional distributions. *The 2016 ACM SIGKDD Workshop on Causal Discovery* (2016).

[27] 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.

[28] Getoor, Lise, Friedman, Nir, Koller, Daphne, Pfeffer, Avi, and Taskar, Ben. Probabilistic relational models. In *Introduction to Statistical Relational Learning.* The MIT Press, 2007, ch. 5, pp. 129–174.

[29] Granger, Clive W. J. Investigating Causal Relations by Econometric Models and Cross-Spectral Methods. *Econometrica, Econometric Society 37*, 3 (July 1969), 424–38.

[30] Gretton, Arthur, Bousquet, Olivier, Smola, Alex, and Scholkopf, Bernhard. Measuring statistical dependence with Hilbert-Schmidt norms. In *ALT* (2005), vol. 16, Springer, pp. 63–78.

[31] Gui, Huan, Xu, Ya, Bhasin, Anmol, and Han, Jiawei. Network a/b testing: From sampling to estimation. In *Proceedings of the 24th International World Wide Web Conference* (2015), ACM, pp. 399–409.

[32] Heckerman, David, Meek, Chris, and Koller, Daphne. Probabilistic models for relational data. Tech. rep., Microsoft Research and Stanford University, 2004.

[33] 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, 2007.

[34] Hernán, Miguel A., and Robins, James M. Instruments for causal inference: An epidemiologist's dream? *Epidemiology 17*, 4 (2006), 360–372.

[35] Holland, Paul W., Laskey, Kathryn Blackmond, and Leinhardt, Samuel. Stochastic blockmodels: First steps. *Social Networks 5*, 2 (1983), 109–137.

[36] 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* (2009), pp. 689–696.

[37] Imbens, Guido W., and Rubin, Donald B. *Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction.* Cambridge University Press, New York, NY, USA, 2015.

[38] Ishak, Mouna Ben, Leray, Philippe, and Amor, Nahla Ben. A hybrid approach for probabilistic relational models structure learning. In *International Symposium on Intelligent Data Analysis* (2016), Springer, pp. 38–49.

[39] Jaeger, Manfred. Relational Bayesian Networks. *Proceedings of the thirteenth Conference on Uncertainty in artificial intelligence* (1997), 266–273.

[40] Kersting, Kristian, Raedt, Luc De, and Raiko, Tapani. Logical hidden Markov models. *Journal of Artificial Intelligence Research 25* (2006), 425–456.

[41] Kersting, Kristian, and Raiko, Tapani. "Say EM" for selecting probabilistic models for logical sequences. In *Proceedings of the Twenty-First Conference in Uncertainty in Artificial Intelligence* (2005), pp. 300–307.

[42] Koller, Daphne, and Pfeffer, Avi. Object-Oriented Bayesian Networks. *Proceedings of the thirteenth Conference on Uncertainty in Artificial Intelligence* (1997), 302–313.

[43] Lähdesmäki, Harri, and Shmulevich, Ilya. Learning the structure of dynamic bayesian networks from time series and steady state measurements. *Machine Learning 71*, 2-3 (June 2008), 185–217.

[44] Laskey, Kathryn Blackmond. MEBN: A language for first-order Bayesian knowledge bases. *Artificial Intelligence 172* (2008), 140–178.

[45] Lee, Sanghack, and Honavar, Vasant. A characterization of markov equivalence classes of relational causal models under path semantics. In *Proc. Conf. Uncertainty Artif. Intell* (2016), pp. 387–396.

[46] Lee, Sanghack, and Honavar, Vasant. On learning causal models from relational data. In *AAAI* (2016), pp. 3263–3270.

[47] Leskovec, Jure, and Krevl, Andrej. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

[48] Liu, Yan, Niculescu-Mizil, Alexandru, Lozano, Aurelie C., and Lu, Yong. Learning temporal causal graphs for relational time-series analysis. In *Proceedings of the Twenty-Seventh International Conference on Machine Learning* (2010), pp. 687–694.

[49] 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.

[50] Maier, Marc, Marazopoulou, Katerina, and Jensen, David. Reasoning about Independence in Probabilistic Models of Relational Data. *arXiv preprint arXiv:1302.4381* (2013).

[51] Manfredotti, Cristina E. *Modeling and Inference with Relational Dynamic Bayesian Networks*. PhD thesis, University of Milano, 2009.

[52] Marazopoulou, Katerina, Arbour, David, and Jensen, David. Refining the semantics of social influence. *Networks: From Graphs to Rich Data, NIPS Workshop* (2014).

[53] Marazopoulou, Katerina, Arbour, David, and Jensen, David. On causal analysis for heterogeneous networks. *The 2017 ACM SIGKDD Workshop on Causal Discovery* (2017).

[54] Marazopoulou, Katerina, Maier, Marc, and Jensen, David. Learning the structure of causal models with relational and temporal dependence. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence* (2015).

[55] Margaritis, Dimitris. *Learning Bayesian network model structure from data.* PhD thesis, US Army, 2003.

[56] Milch, Brian, Marthi, Bhaskara, Russell, Stuart, Sontag, David, Ong, Daniel L., and Kolobov, Andrey. BLOG: Probabilistic Models with Unknown Objects. *Statistical Relational Learning* (2007), 373.

[57] Moneta, Alessio, Chlaß, Nadine, Entner, Doris, and Hoyer, Patrik. Causal search in structural vector autoregressive models. In *NIPS Mini-Symposium on Causality in Time Series* (2011), pp. 95–114.

[58] Murphy, Kevin Patrick. *Dynamic Bayesian Networks: Representation, Inference and Learning.* PhD thesis, University of California, Berkeley, 2002.

[59] Nodelman, Uri, Shelton, Christian R., and Koller, Daphne. Continuous time Bayesian networks. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence* (2002), pp. 378–387.

[60] Nodelman, Uri, Shelton, Christian R., and Koller, Daphne. Learning continuous time Bayesian networks. In *Proceedings of the Nineteenth International Conference on Uncertainty in Artificial Intelligence* (2003), pp. 451–458.

[61] Pearl, Judea. Causal diagrams for empirical research. *Biometrika 82*, 4 (1995), 669–688.

[62] Pearl, Judea, and Verma, Thomas. A theory of inferred causation. In *Principles of Knowledge Representation and Reasoning: Proceeding of the Second International Conference*, J. Allen, R. Fikes, and E. Sandewall, Eds. Morgan Kaufmann, 1991, pp. 441–452.

[63] Peters, Jonas, and Bühlmann, Peter. Structural intervention distance for evaluating causal graphs. *Neural Computation* (2015).

[64] Pfeffer, Avi, Koller, Daphne, Milch, Brian, and Takusagawa, Ken T. SPOOK: A system for probabilistic object-oriented knowledge representation. *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence* (1999), 541–550.

[65] Ramakrishnan, Raghu, and Gehrke, Johannes. *Database Management Systems*. McGraw-Hill international editions: Computer science series. McGraw-Hill, 2003, ch. 2.

[66] Rattigan, Matthew J.H., Maier, Marc, and Jensen, David. Relational blocking for causal discovery. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence* (2011), pp. 145–151.

[67] Ribeiro, Bruno F., Perra, Nicola, and Baronchelli, Andrea. Quantifying the effect of temporal resolution in time-varying network. *CoRR abs/1211.7052* (2012).

[68] Richardson, Matthew, and Domingos, Pedro. Markov logic networks. *Machine Learning 62*, 1-2 (2006), 107–136.

[69] Richardson, Thomas, and Spirtes, Peter. Ancestral graph markov models. *Annals of Statistics* (2002), 962–1030.

[70] Robinson, Joshua W., and Hartemink, Alexander J. Learning non-stationary dynamic bayesian networks. *Journal of Machine Learning Research 11* (Dec. 2010), 3647–3680.

[71] 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.

[72] Rubin, Donald B. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology 66*, 5 (1974), 688.

[73] Saria, Suchi, Nodelman, Uri, and Koller, Daphne. Reasoning at the right time granularity. In *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence, Vancouver, BC, Canada, July 19-22, 2007* (2007), pp. 326–334.

[74] 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*, Oct (2006), 2003–2030.

[75] Shpitser, Ilya, and Pearl, Judea. Dormant independence. In *AAAI* (2008), pp. 1081–1087.

[76] Spirtes, Peter, Glymour, Clark, and Scheines, Richard. *Causation, Prediction and Search*, 2nd ed. MIT Press, Cambridge, MA, 2000.

[77] Spirtes, Peter, and Zhang, Kun. Causal discovery and inference: concepts and recent methodological advances. In *Applied Informatics* (2016), vol. 3, Springer Berlin Heidelberg, p. 3.

[78] Stuart, Elizabeth A. Matching methods for causal inference: A review and a look forward. *Statistical Science 25*, 1 (2010), 1.

[79] Tian, Jin, and Pearl, Judea. On the testable implications of causal models with hidden variables. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence* (2002), Morgan Kaufmann Publishers Inc., pp. 519–527.

[80] Toulis, Panos, and Kao, Edward. Estimation of causal peer influence effects. In *Proceedings of The 30th International Conference on Machine Learning* (2013), pp. 1489–1497.

[81] Tsamardinos, Ioannis, Brown, Laura E., and Aliferis, Constantin F. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning 65*, 1 (2006), 31–78.

[82] Ugander, Johan, Karrer, Brian, Backstrom, Lars, and Kleinberg, Jon. Graph cluster randomization: Network exposure to multiple universes. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2013), ACM, pp. 329–337.

[83] Ugander, Johan, Karrer, Brian, Backstrom, Lars, and Marlow, Cameron. The anatomy of the facebook social graph. *arXiv preprint arXiv:1111.4503* (2011).

[84] 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.

[85] Verma, Thomas, and Pearl, Judea. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence* (New York, NY, USA, 1991), UAI '90, Elsevier Science Inc., pp. 255–270.

[86] Voortman, Mark, Dash, Denver, and Druzdzel, Marek J. Learning why things change: The difference-based causality learner. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence* (2010), pp. 641–650.

[87] Wasserman, Stanley. Analyzing social networks as stochastic processes. *Journal of the American Statistical Association 75*, 370 (1980), 280–294.

[88] Wasserman, Stanley. A stochastic model for directed graphs with transition rates determined by reciprocity. *Sociological Methodology 11* (1980), 392–412.

[89] Watts, Duncan J., and Strogatz, Steven H. Collective dynamics of small-world networks. *Nature 393*, 6684 (1998), 440–442.

[90] Xiang, Rongjing, and Neville, Jennifer. Relational learning with one network: An asymptotic analysis. In *AISTATS* (2011), pp. 779–788.

[91] Yang, Shuo, Khot, Tushar, Kersting, Kristian, and Natarajan, Sriraam. Learning continuous-time bayesian networks in relational domains: A non-parametric approach. In *AAAI* (2016), pp. 2265–2271.

[92] 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), AUAI Press, pp. 647–655.

[93] 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* (2009), pp. 1937–1944.