Reasoning about Independence in Probabilistic Models of Relational Data

Marc Maier **Katerina Marazopoulou** **David Jensen**
School of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003
{maier, kmarazo, jensen}@cs.umass.edu

Abstract

The rules of d-separation provide a theoretical and algorithmic framework for deriving conditional independence facts from model structure. However, this theory only applies to Bayesian networks. Many real-world systems are characterized by interacting heterogeneous entities and probabilistic dependencies that cross the boundaries of entities. Consequently, researchers have developed extensions to Bayesian networks that can represent these relational dependencies. We show that the theory of d-separation inaccurately infers conditional independence when applied directly to the structure of probabilistic models of relational data. We introduce relational d-separation, a theory for deriving conditional independence facts from relational models, and we provide a new representation, the *abstract ground graph*, that enables a sound, complete, and computationally efficient method for answering d-separation queries about relational models.

1 INTRODUCTION

The rules of d-separation are the foundation for algorithmic derivation of the conditional independence facts implied by the structure of Bayesian networks (Geiger et al., 1990). Accurate reasoning about conditional independence facts is the basis for constraint-based algorithms, such as PC and FCI (Spirtes et al., 2000), and hybrid approaches, such as MMHC (Tsamardinos et al., 2006), that learn the structure of Bayesian networks. When interpreting a Bayesian network causally, the causal Markov condition (variables are independent of their non-effects given their direct causes) and d-separation connect the causal structure and conditional independence (Scheines, 1997).

Bayesian networks assume that data instances are independent and identically distributed (i.i.d.), but many real-world systems are characterized by interacting, heterogeneous entities. For example, citation data involve researchers collaborating and authoring scholarly papers that cite prior work. Over the past 15 years, researchers in statistics and computer science have devised more expressive classes of directed graphical models, such as probabilistic relational models (PRMs), which remove the assumptions of i.i.d. data (Getoor and Taskar, 2007). Relational models generalize other classes of models that incorporate interference, spillover effects, or violations of the stable unit treatment value assumption (SUTVA) (Hudgens and Halloran, 2008; Tchetgen and VanderWeele, 2012) and multilevel or hierarchical models (Gelman and Hill, 2007). Many applications have benefited from learning and reasoning with relational models, such as the analysis of gene regulatory interactions (Segal et al., 2001), scholarly citations (Taskar et al., 2001), and biological cellular networks (Friedman, 2004).

In this paper, we show that d-separation does not correctly produce conditional independence facts when applied directly to relational models. We introduce an alternative representation, the *abstract ground graph*, that enables algorithmic derivation of conditional independence facts from relational models. We show that this algorithm is sound, complete, and computationally efficient. Proofs and empirical results can be found in an extended version (Maier et al., 2013).

1.1 WHY D-SEPARATION IS USEFUL

A Bayesian network, as a model of a joint probability distribution, enables a wide array of useful tasks by supporting inference over an entire system of variables. Naïvely specifying a joint distribution by hand requires an exponential number of states; however, Bayesian networks leverage the Markov condition to represent conditional independencies in order to compactly specify a joint probability distribution.

Alternative to the Markov condition, but equivalent in its implications (Neapolitan, 2004), d-separation provides an algorithmic framework to derive the conditional independencies encoded by the model. These conditional independence facts are guaranteed to hold in every faithful distribution the model represents and, consequently, any sampled data instance. The semantics of holding across all distributions is the main reason why d-separation is a useful theory.

Causal discovery, the task of learning generative models of observational data, superficially appears to be a futile endeavor. Yet learning and reasoning about the causal structure of real domains is a primary goal for many researchers. Fortunately, d-separation offers a connection between causal structure and conditional independence. The theory of d-separation can be leveraged to constrain the hypothesis space by eliminating models that are inconsistent with observed conditional independence facts. While many distributions do not lead to uniquely identifiable models, this approach (under simple assumptions) frequently discovers useful causal knowledge for domains that can be represented as a Bayesian network.

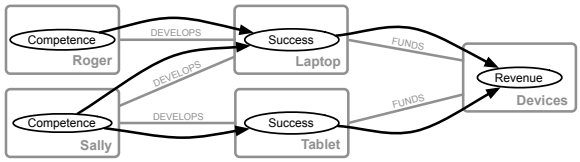
As described above, relational models more closely represent the real-world domains that many social scientists and other researchers investigate. To successfully learn causal models from observational data of relational domains, we need a similar theory for deriving conditional independence from relational models. In this paper, we formalize the theory of relational d-separation, providing a theoretical framework for algorithms that learn causal models of relational domains.

2 EXAMPLE

Consider a corporate analyst who was hired to identify which products and employees are effective and productive for some organization. If the company is structured as a pure project-based organization, the analyst may collect data as described by the relational schema in Figure 1(a) (without the dependencies). The schema denotes that employees can collaborate and work on multiple products, each of which is funded by a specific business unit. The analyst has also obtained variables on each entity—competence of employees, the success of each product, and the revenue of business units. In this example, the organization consists of two employees, two products, and a single business unit, shown in the relational skeleton (in gray) in Figure 1(b).

The analyst may believe that the organization operates under the model depicted in Figure 1(a). The competence of an employee affects the success of products they develop, and the revenue of a business unit

(a) Example relational model. Competence of employees cause the success of products they develop, which in turn influences the revenue received by the business unit funding the product. The dependencies are specified by relational paths, listed below the graphical model.



(b) Example fragment of a ground graph. The success of Laptop is influenced by the competence of both Roger and Sally. The revenue of Devices is caused by the success of all its funded products—Laptop and Tablet.

Figure 1: An example relational model and ground graph for the organization domain.

is influenced by the success of products that it funds. The analyst then needs to verify the model structure in order to accurately advise executive decisions, such as determining which business units should have increased funding. Perhaps the analyst has experience in graphical models and decides to check that the conditional independencies encoded by the model are reflected in the data, assuming the faithfulness condition. The analyst then naïvely applies d-separation to the model structure in an attempt to derive these conditional independencies. However, applying d-separation directly to relational models does not correctly derive the set of conditional independencies. In other words, d-separation applied directly to relational models is not equivalent to the Markov condition.

Naïvely applying d-separation to the model in Figure 1(a) suggests that employee competence is conditionally independent of the revenue of business units given the success of products. To see why this approach is flawed, we must consider the *ground graph*. A necessary precondition for inference is to apply a model to a data instantiation, which yields a ground graph to which d-separation can be applied. For a Bayesian network, a ground graph consists of replicates of the model structure for each data instance. In contrast, a relational model defines a template that results in ground graphs with varying structure depending on the data instantiation.

Figure 1(b) shows the ground graph for the relational model in Figure 1(a) applied to the relational skeleton corresponding to this small company. This ground graph illustrates that conditioning on product success

activates a path through the competence of other employees who develop the same products—we call this a *relational d-connecting path.*[1] Checking d-separation on the ground graph indicates that to d-separate employee competence from business unit revenue, we cannot condition only on the success of developed products, but should also condition on the competence of other employees who work on those products, e.g.,

$$\text{Roger}.\mathit{Competence} \perp\!\!\!\perp \text{Devices}.\mathit{Revenue} \mid \{\text{Laptop}.\mathit{Success}, \text{Sally}.\mathit{Competence}\}.$$

In fact, this is not the only conditional independence fact for which d-separation produces an incorrect result for this example model. Only 25% of the pairs of relational variables can be described by direct inspection of the model, and of those (such as the above example), 75% yield an incorrect conclusion.

It might appear that, since the standard rules of d-separation apply to Bayesian networks and the ground graphs of relational models are also Bayesian networks, that applying d-separation to relational models is a non-issue. However, applying d-separation to a single ground graph may result in excessively long runtime if the instantiation is large—especially given that relational databases can be arbitrarily large. Further, and more importantly, the semantics of d-separation require that conditional independencies hold across all possible model instantiations. Although d-separation can apply directly to a ground graph, this semantics prohibits reasoning about a single ground graph.

The conditional independence facts derived from d-separation hold for all faithful distributions represented by a Bayesian network. Therefore, the implications of relational d-separation should analogously hold for all faithful distributions of variables for the space of all possible ground graphs. It is simple to show that d-separation holds for any ground graph of a Bayesian network—every ground graph consists of a set of disconnected subgraphs, each of which has a structure identical to that of the model. However, relational models produce ground graphs that vary with the relational structure of the underlying data (e.g., different products are developed by varying numbers of employees). As a result, relational d-separation queries must be answered without respect to ground graphs. Additionally, the example illustrates how relational dependencies can exhibit d-connecting paths that are only manifest in ground graphs, not the model representation. In Section 4, we describe a new representation, the abstract ground graph, that can be used to reason about d-separation for relational models.

[1]The indirect effect attributed to a relational d-connecting path is often referred to as interference, a spillover effect, or a violation of SUTVA because the treatment of one instance affects the outcome of another.

3 CONCEPTS OF RELATIONAL DATA AND MODELS

In this section, we formally define the concepts of relational data and models, providing the basis for the theory of relational d-separation. Note that the relational representation is strictly more expressive than the representation assumed by Bayesian networks.

A relational schema is a top-level description of what data exist in a particular domain. Specifically (adapted from Heckerman et al., 2007):

Definition 1 (Relational schema) A *relational schema* $\mathcal{S} = (\mathcal{E}, \mathcal{R}, \mathcal{A})$ consists of a set of entity classes $\mathcal{E} = \{E_1, \ldots, E_m\}$; relationship classes $\mathcal{R} = \{R_1, \ldots, R_n\}$, where each $R_i = \{E_1, \ldots, E_j\}$ with $E_i, E_j \in \mathcal{E}$ and $E_i \neq E_j$; attribute classes $\mathcal{A}(I)$ for each item $I \in \mathcal{E} \cup \mathcal{R}$; and cardinality function $card(R, E) = \{\text{ONE, MANY}\}$ for each $R \in \mathcal{R}$ and $E \in R$.

A relational schema can be represented graphically with an entity-relationship (ER) diagram. An example relational schema is shown in Figure 1(a), with the exception of the model's dependencies (directed arrows). Entities are rectangular boxes, relationships are diamonds connected to their associated entities, attributes are ovals residing on entities and relationships, and cardinalities are represented with crow's foot notation.

A relational schema is a template for a relational skeleton—an instantiation of entities and relationships. Specifically (adapted from Heckerman et al., 2007):

Definition 2 (Relational skeleton) A *relational skeleton* σ is an instantiation of entity sets $\sigma(E)$ for each $E \in \mathcal{E}$ and relationship sets $\sigma(R)$ for each $R \in \mathcal{R}$, adhering to its cardinality constraints. Let $r \in \sigma(R)$ where $R = \{E_1, \ldots, E_j\}$ be denoted as $r(e_1, \ldots, e_j)$ where $e_i \in \sigma(E_i)$ and $E_i \in \mathcal{E}$.

An example of a relational skeleton can be seen (in gray) underlying the ground graph of Figure 1(b).

In order to specify a model over a relational domain, we must define a space of possible variables and dependencies. For relational data, the variable space includes intrinsic entity and relationship attributes, and also the variables on other entities and relationships that are reachable by paths along the relational skeleton. As above, paths in a relational skeleton are instantiations of path templates on a relational schema.

Definition 3 (Relational path) A *relational path* $[I_1, \ldots, I_k]$ for relational schema $\mathcal{S}$ is an alternating sequence of entity and relationship classes $I_1, \ldots, I_k \in$

$\mathcal{E} \cup \mathcal{R}$ such that for all $j > 1$: (1) if $I_j \in \mathcal{E}$, then $I_{j-1} \in \mathcal{R}$ and I_j participates in I_{j-1} ($I_j \in I_{j-1}$), (2) if $I_j \in \mathcal{R}$, then $I_{j-1} \in \mathcal{E}$ and I_{j-1} participates in I_j ($I_{j-1} \in I_j$), and (3) for each ordered triple $\langle I_{j-1}, I_j, I_{j+1} \rangle$ in $[I_1, \ldots, I_k]$, if $I_j \in \mathcal{R}$, then $I_{j-1} \neq I_{j+1}$; otherwise, if $I_j \in \mathcal{E}$, then if $I_{j-1} = I_{j+1}$ then $card(I_{j-1}, I_j) = \text{MANY}$. I_1 is called the *base item*, or *perspective*, of the relational path.

Definition 3 generalizes the notion of "slot chains" from PRMs (Getoor et al., 2007) by including cardinality constraints and formally describing the semantics under which repeated item classes may appear on a path. Condition (3) in the definition removes paths that would invariably reach an empty terminal set (see Definition 4). Also, since relational paths may become arbitrarily long, the path length is ordinarily limited by a user-specified, domain-specific hop threshold.

An instantiated relational path produces a set of traversals on a relational skeleton. However, the quantity of interest is not the paths themselves, but the set of reachable item instances:

Definition 4 (Terminal set: relational path) For any skeleton σ and any $i_1 \in \sigma(I_1)$, a *terminal set* $P|_{i_1}$ for relational path $P = [I_1, \ldots, I_k]$ can be defined inductively as

$$[I_1]|_{i_1} = \{i_1\}$$

$$[I_1, \ldots, I_{k-1}, I_k]|_{i_1} = \bigcup_{i_{k-1} \in [I_1, \ldots, I_{k-1}]|_{i_1}} \{i_k \mid ((i_{k-1} \in i_k \text{ if } I_k \in \mathcal{R}) \vee (i_k \in i_{k-1} \text{ if } I_k \in \mathcal{E})) \wedge i_k \notin [I_1, \ldots, I_j]|_{i_1} \text{ for } j = 1 \text{ to } k-1\}$$

A terminal set of a relational path consists of reachable instances of class I_k, the terminal item on the path. Conceptually, a terminal set is produced by traversing the skeleton beginning at a single base item $i_1 \in \sigma(I_1)$, following instances of the items in the relational path, and reaching a target set of I_k instances. The definition implies a "bridge burning" semantics under which no instantiated items are revisited.[2]

Example 1 The set of relational paths for the schema in Figure 1(a) from the EMPLOYEE perspective with hop threshold $h = 4$ includes the following: [EMPLOYEE] (employees, 0 hops), [EMPLOYEE, DEVELOPS, PRODUCT] (products developed by employees, 2 hops), and [EMPLOYEE, DEVELOPS, PRODUCT, DEVELOPS, EMPLOYEE] (other employees developing the same products, 4 hops). Let Sally be a base item instance. Then terminal sets for the previous relational paths are: $[\text{EMPLOYEE}]|_{\text{Sally}} = \{\text{Sally}\}$, $[\text{EMPLOYEE, DEVELOPS, PRODUCT}]|_{\text{Sally}} = \{\text{Laptop, Tablet}\}$, and $[\text{EMPLOYEE, DEVELOPS, PRODUCT, DEVELOPS, EMPLOYEE}]|_{\text{Sally}} = \{\text{Roger}\}$. The bridge burning semantics enforces that Sally is not also included in this last terminal set. □

Most relational paths start and end with different item classes. However, there are pairs of distinct relational paths that start and end with the same item classes. For these pairs, it is possible that their terminal sets, when originating at the same base item instance, will have items in common. The following lemma states that if two relational paths with the same base and target items diverge in the middle of the path, then for some relational skeleton, their terminal sets will have a non-empty intersection. This property is important to consider for relational d-separation, and this is the only form for which non-empty intersection can occur.

Lemma 1 *For any schema $\mathcal{S}$ and any two relational paths $P_1 = [I_1, \ldots, I_m, \ldots, I_k]$ and $P_2 = [I_1, \ldots, I_n, \ldots, I_k]$ with $I_m \neq I_n$, there exists a skeleton σ such that $P_1|_{i_1} \cap P_2|_{i_1} \neq \emptyset$ for some $i_1 \in \sigma(I_1)$.*

Example 2 Let Path_1 = [EMPLOYEE, DEVELOPS, PRODUCT, DEVELOPS, EMPLOYEE, DEVELOPS, PRODUCT], for which terminal sets yield other products developed by collaborating employees. Let Path_2 = [EMPLOYEE, DEVELOPS, PRODUCT, FUNDS, BUSINESS-UNIT, FUNDS, PRODUCT], for which terminal sets consist of other products funded by the business units funding products developed by a given employee. For base item instance Roger: $\text{Path}_1|_{\text{Roger}} = \{\text{Tablet}\}$ and $\text{Path}_2|_{\text{Roger}} = \{\text{Tablet}\}$. □

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

Definition 5 (Relational variable) A *relational variable* $[I_1, \ldots, I_k].V$ consists of a relational path $[I_1, \ldots, I_k]$ and an attribute class $V \in \mathcal{A}(I_k)$.

Definition 6 (Terminal set: relational variable) For any skeleton σ and $i_1 \in \sigma(I_1)$, a *terminal set* $P.V|_{i_1}$ for relational variable $P.V = [I_1, \ldots, I_k].V$ is the set of variable instances $\{i_k.V \mid V \in \mathcal{A}(i_k) \wedge i_k \in P|_{i_1}\}$.

As a notational convenience, if $\mathbf{X}$ is a set of relational variables, all from a common perspective I_1, then we say that $\mathbf{X}|_{i_1}$ for some item $i_1 \in \sigma(I_1)$ is the union of all terminal sets, $\{x \mid x \in X|_{i_1} \wedge X \in \mathbf{X}\}$. Given the formal definitions for relational variables, we can now define relational dependencies.

Definition 7 (Relational dependency) A *relational dependency* $[I_1, \ldots, I_k].V_1 \rightarrow [I_1].V_2$ consists of

[2] The bridge burning semantics yields terminal sets that are necessarily subsets of terminal sets which would otherwise be produced without bridge burning. Although this appears to be limiting, it actually enables a strictly more expressive class of relational models.

two relational variables with a common base item and corresponds to a directed probabilistic dependence from $[I_1, \ldots, I_k].V_1$ to $[I_1].V_2$.

Depending on the context, $[I_1, \ldots, I_k].V_1$ and $[I_1].V_2$ can be referred to as *treatment* and *outcome*, *cause* and *effect*, or *parent* and *child*. Without loss of generality, Definition 7 provides a canonical specification for dependencies, with the child relational variable restricted to singleton paths, thus ensuring that terminal sets of child relational variables consist of single values.

Example 3 The dependencies in the relational model displayed in Figure 1(a) can be specified as: [PRODUCT, DEVELOPS, EMPLOYEE].*Competence* → [PRODUCT].*Success* (product success is influenced by the competence of the employees developing the product), and [BUSINESS-UNIT, FUNDS, PRODUCT].*Success* → [BUSINESS-UNIT].*Revenue* (the success of the products funded by a business unit influences that unit's revenue). □

A relational model is a schema paired with a collection of relational dependencies, defined as:

Definition 8 (Relational model) The structure of a *relational model* $\mathcal{M} = (\mathcal{S}, \mathcal{D})$ consists of a schema $\mathcal{S}$ and a set of relational dependencies $\mathcal{D}$ defined over $\mathcal{S}$.

A relational model can be represented graphically by superimposing dependencies on the ER diagram of a relational schema (see Figure 1(a) for an example). This definition of relational models is consistent with and yields structures expressible as DAPER models (Heckerman et al., 2007). These relational models are also equivalent to PRMs, but we generalize slot chains as relational paths and provide a formal semantics for their instantiation. These models also generalize plate models because dependencies can be specified with arbitrary relational paths as opposed to simple intersections among plates (Buntine, 1994; Gilks et al., 1994).

Relational models only define coherent joint probability distributions if they produce acyclic model instantiations (i.e., ground graphs, defined below). A useful construct for checking model acyclicity is the class dependency graph—a directed graph with nodes for each attribute of each item class and edges between pairs of attributes supported by relational dependencies in the model (Getoor et al., 2007). If the relational dependencies form an acyclic class dependency graph, then every possible ground graph of that model is acyclic as well. All future references to acyclic relational models refer to relational models having dependency structures that form acyclic class dependency graphs.

A parameterized relational model contains conditional probability distributions for every attribute class $\mathcal{A}(I)$ for each $I \in \mathcal{E} \cup \mathcal{R}$ in order to represent a joint probability distribution. Similar to Bayesian networks, the joint distribution factorizes according to the conditional distributions given a relational skeleton σ as

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

where $GG_{\mathcal{M}\sigma}$ is the ground graph of the model and skeleton, defined below. Note that without a generative model of relational skeletons, these relational models are not truly generative as the skeleton must be generated prior to the attributes. However, the same issue occurs for Bayesian networks—relational skeletons consist of disconnected entity instances, but the number of such instances to create is not described by the model. We choose simple processes for generating skeletons and focus on relational models of attributes, leaving structural causes and effects as future work.

Just as the relational schema is a template for skeletons, a relational model is a template for ground graphs—dependencies applied to skeletons.

Definition 9 (Ground graph) The *ground graph* $GG_{\mathcal{M}\sigma} = (V, E)$ for relational model $\mathcal{M} = (\mathcal{S}, \mathcal{D})$ and skeleton σ is a directed graph with nodes $V = \mathcal{A}(\sigma) = \{i.X \mid I \in \mathcal{E} \cup \mathcal{R} \ \wedge \ X \in \mathcal{A}(I) \ \wedge \ i \in \sigma(I)\}$ and edges $E = \{i_k.Y \rightarrow i_j.X \mid i_k.Y, i_j.X \in V \ \wedge \ i_k.Y \in [I_j, \ldots, I_k].Y|_{i_j} \ \wedge \ [I_j, \ldots, I_k].Y \rightarrow [I_j].X \in \mathcal{D}\}$.

A ground graph is a directed graph, with a node for every variable of every entity and relationship instance in a skeleton, and an edge between pairs of variable instances belonging to the terminal sets of the relational variables of all dependencies in a model. In fact, given an acyclic relational model, the ground graph has the same semantics as a Bayesian network (Getoor, 2001; Heckerman et al., 2007). Figure 1(b) displays an example ground graph superimposed on a skeleton.

By Lemma 1 and Definition 9, the same canonical dependency can connect many relational variables. If the terminal sets involve $i_k.Y$ and $i_j.X$, then there is a dependency between all relational variables for which $i_k.Y$ and $i_j.X$ are elements. These implied dependencies form part of the challenge of identifying independence in relational models. Additionally, the intersection between the terminal sets of two relational paths is crucial for reasoning about independence. Since d-separation only guarantees independence when there are no d-connecting paths, we must consider all possible paths among variable instances, any one of which may be a member of multiple relational variables. In Section 4, we define relational d-separation and provide an appropriate representation, the abstract ground graph, that enables straightforward reasoning about d-separation.

4 RELATIONAL D-SEPARATION

Conditional independence facts are entailed by the rules of d-separation, but only for simple directed acyclic graphs. Recall that every ground graph of a Bayesian network consists of a set of identical copies of the model structure. Thus, the implications of d-separation on Bayesian networks hold for every ground graph. In contrast, relational models are templates for ground graphs that vary with underlying skeletons. That is, the set of distributions represented by a relational model is not only parameterized by conditional probabilities, but also by the space of valid skeletons given by the schema. Since conditional independence facts are only useful when they hold across all possible model instantiations, reasoning about d-separation for relational models is inherently more challenging and leads to the following definition:

Definition 10 (Relational d-separation) Let $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ be three distinct sets of relational variables for perspective $B \in \mathcal{E} \cup \mathcal{R}$ defined over relational schema $\mathcal{S}$. Then, for relational model $\mathcal{M}$, $\mathbf{X}$ and $\mathbf{Y}$ are d-separated by $\mathbf{Z}$ if and only if, for any skeleton σ, $\mathbf{X}|_b$ and $\mathbf{Y}|_b$ are d-separated by $\mathbf{Z}|_b$ in ground graph $GG_{\mathcal{M}\sigma}$ for all $b \in \sigma(B)$.

In other words, for $\mathbf{X}$ and $\mathbf{Y}$ to be d-separated by $\mathbf{Z}$ for relational model $\mathcal{M}$, d-separation must hold for all instantiations of those relational variables for any possible skeleton. This is a conservative definition, but it is consistent with the semantics of d-separation on Bayesian networks—it guarantees independence, but it does not guarantee dependence. If there exists even one valid skeleton and faithful distribution represented by the relational model for which $\mathbf{X} \not\perp\!\!\!\perp \mathbf{Y} \mid \mathbf{Z}$, then $\mathbf{X}$ and $\mathbf{Y}$ are not d-separated by $\mathbf{Z}$.

Given the semantics specified in Definition 10, answering relational d-separation queries is challenging for the following reasons:

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

Relational models are templates: Relational models appear to be directed acyclic graphs, but they are templates for constructing ground graphs. The rules of d-separation do not directly apply to relational models, only to their ground graphs. Applying the rules of d-separation to a relational model frequently leads to incorrect conclusions because of confounding paths that are only manifest in ground graphs.

Terminal sets of relational variables may intersect: The terminal sets of two different relational variables may have non-empty intersections, as described by Lemma 1. Consequently, there exist non-intuitive implications of dependencies that d-separation must account for, such as the relational d-connecting paths in the example in Section 2.

Relational dependency specification: Relational models are defined with canonical dependencies, each specified from a single perspective. However, variables in a ground graph may contribute to the terminal sets of *multiple* relational variables, each defined from *different* perspectives. Thus, we need methods to translate canonical dependencies to produce the implied dependencies between arbitrary relational variables.

4.1 ABSTRACT GROUND GRAPHS

The definition of relational d-separation and its challenges suggest a solution that abstracts over all possible ground graphs and explicitly represents the potential intersection between the terminal sets of pairs of relational variables. We introduce a new lifted representation, called the *abstract ground graph*, that captures all dependencies among arbitrary relational variables for any ground graph, using knowledge of only the schema and the model.

Definition 11 (Abstract ground graph) An *abstract ground graph* $AGG_{\mathcal{M}Bh} = (V, E)$ for relational model $\mathcal{M} = (\mathcal{S}, \mathcal{D})$, perspective $B \in \mathcal{E} \cup \mathcal{R}$, and hop threshold $h \in \mathbb{N}^0$ is an abstraction of the dependencies $\mathcal{D}$ for all possible ground graphs $GG_{\mathcal{M}\sigma}$ of $\mathcal{M}$ on arbitrary skeletons σ.

The set of nodes in $AGG_{\mathcal{M}Bh}$, $V = RV \cup IV$, is the union of all relational variables $RV = \{[B, \ldots, I_j].V \mid length([B, \ldots, I_j]) \leq h + 1\}$ and the intersection between pairs of relational variables that may intersect $IV = \{X \cap Y \mid X, Y \in RV \wedge X = [B, \ldots, I_k, \ldots, I_j].V \wedge Y = [B, \ldots, I_l, \ldots, I_j].V \wedge I_k \neq I_l\}$.

The set of edges in $AGG_{\mathcal{M}Bh}$ is $E = RVE \cup IVE$, where $RVE \subset RV \times RV$ and $IVE \subset IV \times RV \cup RV \times IV$. RVE is the set of edges between pairs of relational variables: $RVE = \{[B, \ldots, I_k].V_1 \rightarrow [B, \ldots, I_j].V_2 \mid [I_j, \ldots, I_k].V_1 \rightarrow [I_j].V_2 \in \mathcal{D} \wedge [B, \ldots, I_k] \in extend([B, \ldots, I_j], [I_j, \ldots, I_k])\}$.

IVE is the set of edges inherited from both relational variable sources of every intersection variable: $IVE = \{X \rightarrow [B, \ldots, I_j].V_2 \mid X = P_1.V_1 \cap P_2.V_1 \in$

$IV \wedge (P_1.V1 \rightarrow [B, \ldots, I_j].V_2 \in RVE \vee P_2.V_1 \rightarrow [B, \ldots, I_j].V_2 \in RVE)\} \cup \{[B, \ldots, I_j].V_1 \rightarrow X \mid X = P_1.V_2 \cap P_2.V_2 \in IV \wedge ([B, \ldots, I_j].V_1 \rightarrow P_1.V1 \in RVE \vee [B, \ldots, I_j].V_1 \rightarrow P_2.V_1 \in RVE)\}$.

The *extend* method is described extensively in Maier et al. (2013). Informally, this method translates dependencies specified in the model into dependencies in the abstract ground graph. The construction of an abstract ground graph for relational model $\mathcal{M}$, perspective B, and hop threshold h follows three simple steps: (1) Add a node for all relational variables, with relational path length limited by h. (2) Insert edges for the direct causes of every relational variable. (3) For each pair of potentially intersecting relational variables, add a new "intersection" node that inherits the direct causes and effects from both of its sources. Then, to answer queries of the form "Are $\mathbf{X}$ and $\mathbf{Y}$ d-separated by $\mathbf{Z}$?", simply (1) augment $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ with their corresponding intersection variables and (2) apply the rules of d-separation on the abstract ground graph for the common perspective of $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$.

Example 4 Figure 2 shows the abstract ground graph $AGG_{\mathcal{M},\text{EMPLOYEE},6}$ from the EMPLOYEE perspective with $h = 6$. As in Section 2, we are interested in d-separating individual employee competence ($\mathbf{X} = \{$[EMPLOYEE].*Competence*$\}$) from the revenue of the employee's funding business units ($\mathbf{Y} = \{$[EMPLOYEE, DEVELOPS, PRODUCT, FUNDS, BUSINESS-UNIT].*Revenue*$\}$). Applying the rules of d-separation to the abstract ground graph, we see that the conditioning set $\mathbf{Z}$ must include both product success ([EMPLOYEE, DEVELOPS, PRODUCT].*Success*) and the competence of other employees developing the same products ([EMPLOYEE, DEVELOPS, PRODUCT, DEVELOPS, EMPLOYEE].*Competence*). For $h = 6$, augmenting $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ with their corresponding intersection variables does not result in any changes. For $h = 8$, the abstract ground graph includes a node for relational variable [EMPLOYEE, DEVELOPS, PRODUCT, DEVELOPS, EMPLOYEE, DEVELOPS, PRODUCT, FUNDS, BUSINESS-UNIT].*Revenue* (the revenue of the business units funding the other products of collaborating employees) which, by Lemma 1, could have a non-empty intersection with [EMPLOYEE, DEVELOPS, PRODUCT, FUNDS, BUSINESS-UNIT].*Revenue*. Therefore, $\mathbf{Y}$ would also include the intersection with this other relational variable. However, for this query, the conditioning set $\mathbf{Z}$ for $h = 6$ happens to also d-separate at $h = 8$ (and any $h \in \mathbb{N}^0$). □

Using the algorithm devised by Geiger et al. (1990), relational d-separation queries can be answered in $O(|E|)$ time with respect to the number of edges in the abstract ground graph. In practice, the size of an abstract ground graph depends on properties of the relational schema and model (e.g., the number of entities, the types of cardinalties, the number of dependencies), as well as the hop threshold. For the example in Figure 2, the abstract ground graph has 7 nodes and 7 edges (including 1 intersection node with 2 edges); for $h = 8$, it would have 13 nodes and 21 edges (including 4 intersection nodes with 13 edges). Abstract ground graphs are invariant to the size of ground graphs, even though ground graphs can be arbitrarily large—that is, relational databases have no maximum size.

4.2 PROOF OF CORRECTNESS

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

Theorem 1 *The rules of d-separation are sound and complete for directed acyclic graphs.*

This result is due to Verma and Pearl (1988) for soundness and Geiger and Pearl (1988) for completeness. Theorem 1 implies that (1) all conditional independence facts derived by d-separation on a Bayesian network hold in any faithful distribution represented by that model (soundness) and (2) all conditional independence facts that hold in a faithful distribution can be inferred from d-separation applied to the Bayesian network encoding that distribution (completeness).

Lemma 2 *For any acyclic relational model $\mathcal{M}$ and skeleton σ, the ground graph $GG_{\mathcal{M}\sigma}$ is a directed acyclic graph.*

This result is due to both Heckerman et al. (2007) for DAPER models and Getoor (2001) for PRMs. Lemma 2 states that any ground graph of an acyclic relational model is just a Bayesian network. By Theorem 1, d-separation is sound and complete when applied directly to a ground graph. However, Definition 10 explicitly states that relational d-separation must hold across *all possible* ground graphs, which is the reason for constructing the abstract ground graph representation. Next, we introduce the notion of (B, h)-reachability, which describes the conditions under which we can expect an edge in a ground graph to be represented in an abstract ground graph.

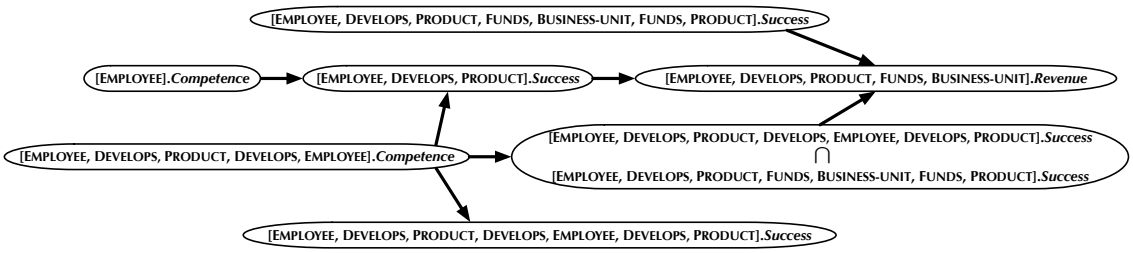


Figure 2: The abstract ground graph for the organization domain model in Figure 1(a) from the EMPLOYEE perspective with hop threshold $h = 6$. This abstract ground graph includes one intersection node.

Definition 12 ((B,h)-reachability) Let $GG_{\mathcal{M}\sigma}$ be the ground graph for some relational model $\mathcal{M}$ and skeleton σ. Then, $i_k.V_1 \rightarrow i_j.V_2 \in GG_{\mathcal{M}\sigma}$ is (B,h)-*reachable* for perspective B and hop threshold h if there exist relational variables $P_k.V_1 = [B,\ldots,I_k].V_1$ and $P_j.V_2 = [B,\ldots,I_j].V_2$ such that $length(P_k) \leq h+1$, $length(P_j) \leq h+1$, and there exists a $b \in \sigma(B)$ where $i_k \in P_k|_b$ and $i_j \in P_j|_b$.

Example 5 For the ground graph in Figure 1(b), let B = EMPLOYEE, $h = 6$, and let $i_k.V_1 \rightarrow i_j.V_2$ be the edge Laptop.*Success* $\rightarrow$ Devices.*Revenue* in the ground graph. This edge is (B,h)-reachable: Set $P_k.V_1$ = [EMPLOYEE, DEVELOPS, PRODUCT].*Success*, $P_j.V_2$ = [EMPLOYEE, DEVELOPS, PRODUCT, FUNDS, BUSINESS-UNIT].*Revenue*, and let b = Sally. We have $length(P_k) < 7$, $length(P_j) < 7$, Laptop $\in P_k|_{\text{Sally}}$, and Devices $\in P_j|_{\text{Sally}}$. □

Since Definition 12 pertains to edges reachable via a particular perspective B and hop threshold h, it relates to the reachability of edges in abstract ground graphs. Specifically, Definition 12 implies that (1) for an edge in any $GG_{\mathcal{M}\sigma}$, we can derive a set of abstract ground graphs for which that edge is (B,h)-reachable, and (2) for any $AGG_{\mathcal{M}Bh}$, we can derive the set of (B,h)-reachable edges for a given ground graph. Given (B,h)-reachability, we can now express the soundness and completeness of abstract ground graphs.

Theorem 2 *For any acyclic relational model $\mathcal{M}$, perspective $B \in \mathcal{E} \cup \mathcal{R}$, and hop threshold $h \in \mathbb{N}^0$, the abstract ground graph $AGG_{\mathcal{M}Bh}$ is (B,h)-reachably sound and complete for any ground graph $GG_{\mathcal{M}\sigma}$ for all skeletons σ.*

Theorem 2 guarantees that, up to the hop threshold for a given perspective, abstract ground graphs capture all possible paths of dependence between any pair of variables in any ground graph.

Theorem 3 *For any acyclic relational model $\mathcal{M}$, perspective $B \in \mathcal{E} \cup \mathcal{R}$, and hop threshold $h \in \mathbb{N}^0$, the abstract ground graph $AGG_{\mathcal{M}Bh}$ is directed and acyclic.*

Theorem 3 ensures that the standard rules of d-separation can apply directly to abstract ground graphs because they are acyclic given an acyclic model. In the following theorem, we define $\bar{\mathbf{W}}$ as the set of nodes augmented with their corresponding intersection nodes for the set of relational variables $\mathbf{W}$: $\bar{\mathbf{W}} = \mathbf{W} \cup \bigcup_{W \in \mathbf{W}} \{W \cap W' \mid W \cap W'$ is an intersection node in $AGG_{\mathcal{M}Bh}\}$. We also say that d-separation holds up to a hop threshold h if there are no d-connecting paths involving a relational variable with path length greater than $h+1$.

Theorem 4 *Relational d-separation is sound and complete for abstract ground graphs up to a specified hop threshold. Let $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$ be three distinct sets of relational variables for perspective $B \in \mathcal{E} \cup \mathcal{R}$ defined over relational schema $\mathcal{S}$. Then, for any skeleton σ and for all $b \in \sigma(B)$, $\mathbf{X}|_b$ and $\mathbf{Y}|_b$ are d-separated by $\mathbf{Z}|_b$ up to hop threshold h in ground graph $GG_{\mathcal{M}\sigma}$ if and only if $\bar{\mathbf{X}}$ and $\bar{\mathbf{Y}}$ are d-separated by $\bar{\mathbf{Z}}$ on the abstract ground graph $AGG_{\mathcal{M}Bh}$.*

Theorem 4 states that d-separation on abstract ground graphs is a sound and complete solution to identifying independence in relational models. Given Theorem 1, the set of conditional independence facts derived from d-separation on abstract ground graphs is identical to (up to a specified hop threshold) the set of conditional independencies in common with all faithful distributions represented by all possible ground graphs.

5 EXPERIMENTS

To complement the theoretical results, Maier et al. (2013) presented a series of four experiments to demonstrate the necessity of the abstract ground graph representation and the feasibility of applying relational d-separation in practice. Here, we summarize the main conclusions of these four experiments.

First, we tested how frequently d-separation applied directly to the model structure derived incorrect conditional independencies. We found that 56% of all queries could not even be represented, and of those

that could be represented and required a non-empty conditioning set, up to 57% were wrong. These results indicate that fully specifying abstract ground graphs is critical for accurately deriving most conditional independence facts from relational models.

Second, we provided an empirical characterization of the factors that influence the size of abstract ground graphs, and, thus, the computational complexity of relational d-separation. This analysis showed that (1) as the number of entities, relationships, attributes, and MANY cardinalities increases, the abstract ground graph grows exponentially with respect to both nodes and edges; and (2) as the number of dependencies in the model increases, the number of edges increases linearly, but the number of nodes remains invariant.

Third, because abstract ground graphs can become large, one might expect that separating sets[3] would also grow to impractical sizes. Fortunately, relational d-separation produces minimal separating sets that are empirically observed to be small. We discovered that, in summation, roughly 83% were marginal independencies, 13% had separating sets of size 1, and less than 0.1% had separating sets with more than 5 variables. These results indicate that separating set size is strongly influenced by model density, primarily because the number of potential d-connecting paths increases as the number of dependencies increases.

Finally, we examined how the expectations of the relational d-separation theory match the results of statistical tests on actual data. We parameterized relational models with linear effects and found that 98% of all expected conditional independencies had an average effect size less than 0.01. The remaining cases that did exhibit a positive effect were discovered to be due to an interaction between aggregation and relational structure, which suggests the need for more accurate tests of conditional independence for relational data.

6 SUMMARY AND DIRECTIONS

In this paper, we extend the theory of d-separation to graphical models of relational data. We present the *abstract ground graph*, a new representation that is (B, h)-reachably sound and complete in its abstraction of dependencies across all possible ground graphs of a given relational model. We formally define relational d-separation and offer a sound, complete, and computationally efficient approach to deriving conditional independence facts from relational models by exploiting their abstract ground graphs. The proofs for all theorems presented in this paper, as well as an empirical analysis of relational d-separation, can be found in Maier et al. (2013).

The results of this paper imply potential flaws in the design and analysis of some real-world studies. If researchers of social or economic systems choose inappropriate data and model representations, then their analyses may omit important classes of dependencies. Our theory implies that choosing a propositional representation from an inherently relational domain may lead to serious errors. Abstract ground graphs define the set of variables that should be included in propositionalizations. The absence of any relational variable may violate causal sufficiency, which could result in the inference of a causal dependency where conditional independence was not detected. Our work indicates that researchers should carefully consider how to represent their domains in order to accurately reason about conditional independence.

Abstract ground graphs also present an opportunity to derive new edge orientation rules for algorithms that learn the structure of relational models, such as RPC (Maier et al., 2010). Deriving and formalizing the implications of relational d-separation is a main direction of future research. This work has also focused solely on relational models of attributes; future work should consider models of relationship and entity existence to fully characterize generative models of relational structure. Finally, the theory could also be extended to incorporate deterministic dependencies, as D-separation extends d-separation for Bayesian networks.

Acknowledgments

The authors wish to thank Cindy Loiselle for her editing expertise. This effort is supported by the Intelligence Advanced Research Project Agency (IARPA) via Department of Interior National Business Center Contract number D11PC20152, Air Force Research Lab under agreement number FA8750-09-2-0187, the National Science Foundation under grant number 0964094, and Science Applications International Corporation (SAIC) and DARPA under contract number P010089628. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements either expressed or implied, of IARPA, DoI/NBC, AFRL, NSF, SAIC, DARPA or the U.S. Government. The Greek State Scholarships Foundation partially supported Katerina Marazopoulou.

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

References

W. L. Buntine. Operations for learning with graphical models. *Journal of Artificial Intelligence Research*, 2:159–225, 1994.

N. Friedman. Inferring cellular networks using probabilistic graphical models. *Science*, 303(5659):799–805, 2004.

D. Geiger and J. Pearl. On the logic of causal models. In *Proceedings of the Fourth Annual Conference on Uncertainty in Artificial Intelligence*, pages 136–147, 1988.

D. Geiger, T. Verma, and J. Pearl. Identifying independence in Bayesian networks. *Networks*, 20(5): 507–534, 1990.

A. Gelman and J. Hill. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press New York, 2007.

L. Getoor. *Learning Statistical Models from Relational Data*. Ph.D. thesis, Stanford University, 2001.

L. Getoor and B. Taskar, editors. *Introduction to Statistical Relational Learning*. MIT Press, Cambridge, MA, 2007.

L. Getoor, N. Friedman, D. Koller, A. Pfeffer, and B. Taskar. Probabilistic relational models. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*, pages 129–174. MIT Press, Cambridge, MA, 2007.

W. R. Gilks, A. Thomas, and D. J. Spiegelhalter. A language and program for complex Bayesian modeling. *The Statistician*, 43:169–177, 1994.

D. Heckerman, C. Meek, and D. Koller. Probabilistic entity-relationship models, PRMs, and plate models. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*, pages 201–238. MIT Press, Cambridge, MA, 2007.

M. G. Hudgens and M. E. Halloran. Toward causal inference with interference. *Journal of the American Statistical Association*, 103(482):832–842, 2008.

M. Maier, B. Taylor, H. Oktay, and D. Jensen. Learning causal models of relational domains. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

M. Maier, K. Marazopoulou, and D. Jensen. *Reasoning about Independence in Probabilistic Models of Relational Data*. arXiv preprint arXiv:1302.4381, 2013.

R. E. Neapolitan. *Learning Bayesian Networks*. Pearson Prentice Hall Upper Saddle River, NJ, 2004.

R. Scheines. An introduction to causal inference. In V. R. McKim and S. P. Turner, editors, *Causality in Crisis? Statistical Methods and the Search for Causal Knowledge in the Social Sciences*, pages 185–199. University of Notre Dame Press, 1997.

E. Segal, B. Taskar, A. Gasch, N. Friedman, and D. Koller. Rich probabilistic models for gene expression. *Bioinformatics*, 17(suppl 1):S243–S252, 2001.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction and Search*. MIT Press, Cambridge, MA, 2nd edition, 2000.

B. Taskar, E. Segal, and D. Koller. Probabilistic classification and clustering in relational data. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 870–878, 2001.

E. J. T. Tchetgen and T. J. VanderWeele. On causal inference in the presence of interference. *Statistical Methods in Medical Research*, 21(1):55–75, 2012.

I. Tsamardinos, L. E. Brown, and C. F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65(1):31–78, October 2006.

T. Verma and J. Pearl. Causal networks: Semantics and expressiveness. In *Proceedings of the Fourth Annual Conference on Uncertainty in Artificial Intelligence*, pages 352–359, 1988.