Lifted Causal Inference in Relational Domains

Malte Luttermann MALTE.LUTTERMANN@DFKI.DE
German Research Center for Artificial Intelligence (DFKI), Lübeck, Germany

Mattis Hartwig MATTIS.HARTWIG@DFKI.DE
German Research Center for Artificial Intelligence (DFKI), Lübeck, Germany
singularIT GmbH, Leipzig, Germany

Tanya Braun TANYA.BRAUN@UNI-MUENSTER.DE
Data Science Group, University of Münster, Germany

Ralf Möller MOELLER@IFIS.UNI-LUEBECK.DE
Institute of Information Systems, University of Lübeck, Germany
German Research Center for Artificial Intelligence (DFKI), Lübeck, Germany

Marcel Gehrke GEHRKE@IFIS.UNI-LUEBECK.DE
Institute of Information Systems, University of Lübeck, Germany

Editors: Francesco Locatello and Vanessa Didelez

Abstract

Lifted inference exploits symmetries in probabilistic graphical models by using a representative for indistinguishable objects, thereby speeding up query answering while maintaining exact answers. Even though lifting is a well-established technique for the task of probabilistic inference in relational domains, it has not yet been applied to the task of causal inference. In this paper, we show how lifting can be applied to efficiently compute causal effects in relational domains. More specifically, we introduce parametric causal factor graphs as an extension of parametric factor graphs incorporating causal knowledge and give a formal semantics of interventions therein. We further present the lifted causal inference algorithm to compute causal effects on a lifted level, thereby drastically speeding up causal inference compared to propositional inference, e.g., in causal Bayesian networks. In our empirical evaluation, we demonstrate the effectiveness of our approach.

Keywords: Causal graphical models, lifted probabilistic inference, interventional distributions

1. Introduction

A fundamental problem in the research field of artificial intelligence for an intelligent agent is to plan and act rationally in a relational domain. To compute the best possible action in a perceived state, the agent considers the available actions and chooses the one with the maximum expected utility. When computing the expected utility of an action that acts on a specific variable, it is crucial to deploy the semantics of an intervention instead of a typical conditioning on that variable (Pearl, 2009, Chapter 4). When calculating the effect of an intervention, a specific variable is set to a fixed value and all incoming probabilistic causal influences of this variable must be ignored for the specific query. It is therefore fundamental to deploy the semantics of an intervention instead of the typical conditioning to correctly determine the effect of an action.

Over the last years, causal graphical models have become a widely used formalism to answer questions concerning the causal impact of a treatment variable on an outcome variable. These models combine probabilistic modeling with causal knowledge, which enables computing the effect of an action that intervenes on a particular variable. As our world is inherently relational (i.e., it

consists of objects and relations between those objects), it is particularly important to have models that represent the relational structure between objects in addition to capturing causal knowledge. However, commonly applied causal graphical models focus on propositional representations while at the same time relational models lack the ability to efficiently apply causal knowledge for inference. Therefore, we aim to combine the best of both worlds to allow for efficient causal inference in relational domains. In particular, this paper deals with the problem of efficiently computing causal effects in models representing objects and their causal relationships to each other.

Previous work. To perform causal effect estimation in causal graphical models, there has been a considerable amount of work and most of this work focuses on models of propositional data (Spirtes et al., 2000; Pearl, 2009). Some works extend propositional factor graphs (FGs) by adding edge directions to enable the computation of the effect of interventions (Frey, 2003; Winn, 2012). Maier et al. (2010) show that propositional models are insufficient to represent causal relationships within relational domains as required by real-world applications. To express causal dependencies within relational domains, Maier et al. (2013) introduce so-called relational causal models (RCMs) but focus on learning RCMs from observed data. Further related work covering RCMs also focuses on causal discovery and reasoning about conditional independence (e.g., Lee and Honavar, 2015, 2016, 2019). RCMs have also been extended to cover cyclic dependency structures (Ahsan et al., 2022, 2023), however, both non-cyclic and cyclic RCMs allow only for reasoning about conditional independence on a lifted level but they do not allow for lifted causal inference. Prior work dealing with the estimation of causal effects in relational domains applies propositional probabilistic inference (Arbour et al., 2016; Salimi et al., 2020) and thus does not scale for large graphs. Consequently, there is a lack of efficient algorithms to compute causal effects in relational domains. In probabilistic inference, lifting exploits symmetries in a relational model, allowing to carry out query answering more efficiently while maintaining exact answers (Niepert and Van den Broeck, 2014). First introduced by Poole (2003), parametric factor graphs (PFGs) and lifted variable elimination (LVE) allow to perform lifted probabilistic inference, i.e., to exploit symmetries in a probabilistic graphical model, resulting in significant speed-ups for probabilistic query answering in relational domains. Over time, LVE has been refined by many researchers to reach its current form (De Salvo Braz et al., 2005, 2006; Milch et al., 2008; Kisyński and Poole, 2009; Taghipour et al., 2013a; Braun and Möller, 2018). PFGs are well-studied for many years and have been developed further to efficiently perform probabilistic inference not only for single queries but also for sets of queries (Braun and Möller, 2016), to incorporate probabilistic inference over time (Gehrke et al., 2018, 2020), and, among other extensions, to allow for decision making by following the maximum expected utility principle (Gehrke et al., 2019a,b; Braun and Gehrke, 2022). Markov logic networks are another lifted representation and have been extended to incorporate maximum expected utility as well (Apsel and Brafman, 2012). Nevertheless, when a decision-making agent plans for the best action to take, previous works improperly apply conditioning (i.e., actions are treated as evidence), as also suggested by Russell and Norvig (2020, Chapter 16), instead of the notion of an intervention. Treating actions as evidence, however, is incorrect as noted by Pearl (2009, Chapter 4). To correctly handle the semantics of an action, the notion of an intervention (Pearl et al., 2016) has to be applied. Therefore, in this paper, we close the gap between PFGs and causal inference in relational domains by introducing parametric causal factor graphs as an extension of parametric factor graphs incorporating causal knowledge to allow for lifted causal inference, thereby enabling efficient decision making using the notion of an intervention.

Our contributions. PFGs are well-established models coming with LVE as a mature lifted inference algorithm, allowing for tractable probabilistic inference with respect to domain sizes in relational domains. We extend PFGs by incorporating causal knowledge, resulting in parametric causal factor graphs (PCFGs) for which we define a formal semantics of interventions. Having defined a formal semantics of interventions in PCFGs, we show how causal effects can be efficiently computed, even for multiple simultaneous interventions. We further introduce the lifted causal inference (LCI) algorithm that operates on a lifted level to allow for lifted causal inference in relational domains. Apart from the theoretical investigation of PCFGs and the LCI algorithm, we provide an empirical evaluation confirming the efficiency of LCI.

Structure of this paper. In Section 2, we introduce both FGs and PFGs as undirected probabilistic graphical models. Thereafter, in Section 3, we define PCFGs as an extension of PFGs incorporating causal knowledge and provide a formal semantics of interventions in PCFGs. In Section 4, we introduce the LCI algorithm operating on a PCFG and show how LCI computes causal effects on a lifted level to avoid grounding the PCFG as much as possible. Afterwards, in our empirical evaluation in Section 5, we investigate the speed-up of LCI compared to performing propositional causal inference both in causal Bayesian networks and in directed FGs before we conclude in Section 6.

2. Preliminaries

We begin by introducing FGs as propositional probabilistic models and afterwards continue to define PFGs which combine probabilistic models and first-order logic to allow for tractable probabilistic inference with respect to domain sizes in relational domains. An FG is an undirected graphical model to compactly encode a full joint probability distribution between random variables (randvars) (Frey et al., 1997; Kschischang et al., 2001). Similar to a Bayesian network (Pearl, 1988), an FG factorises a full joint probability distribution into a product of factors.

Definition 1 (Factor Graph) *An* FG $G = (\boldsymbol{V}, \boldsymbol{E})$ *is a bipartite graph with node set* $\boldsymbol{V} = \boldsymbol{R} \cup \boldsymbol{\Phi}$ *where* $\boldsymbol{R} = \{R_1, \ldots, R_n\}$ *is a set of variable nodes (randvars) and* $\boldsymbol{\Phi} = \{\phi_1, \ldots, \phi_m\}$ *is a set of factor nodes (functions). There is an edge between a variable node* R_i *and a factor node* ϕ_j *in* $\boldsymbol{E} \subseteq \boldsymbol{R} \times \boldsymbol{\Phi}$ *if* R_i *appears in the argument list of* ϕ_j*. A factor is a function that maps its arguments to a positive real number, called potential. The semantics of* G *is given by*

$$P_G = \frac{1}{Z} \prod_{j=1}^{m} \phi_j(\mathcal{A}_j)$$

with Z *being the normalisation constant and* $\mathcal{A}_j$ *denoting the randvars appearing in* ϕ_j*.*

Example 1 *Figure 1 shows a toy example of an FG modelling the relationships between a company's revenue, the competences of its employees, and the training of its employees. More specifically, there are randvars* $Qual.t_i$ *indicating the quality of a training program* t_i*, randvars* $Comp.e_j$ *describing the competence of an employee* e_j*, randvars* $Train.e_j.t_i$ *specifying whether employee* e_j *has been trained with training program* t_i*, and a randvar* Rev *denoting the revenue of the company. In this particular example, there is a single company with four employees* $alice$*,* bob*,* $dave$*, and* eve *and there are two training programs* t_1 *and* t_2 *each employee can be trained with. The randvars* $Qual.t_i$*,* $Comp.e_j$*, and* Rev *can take one of the values* $\{low, medium, high\}$ *and the randvars*

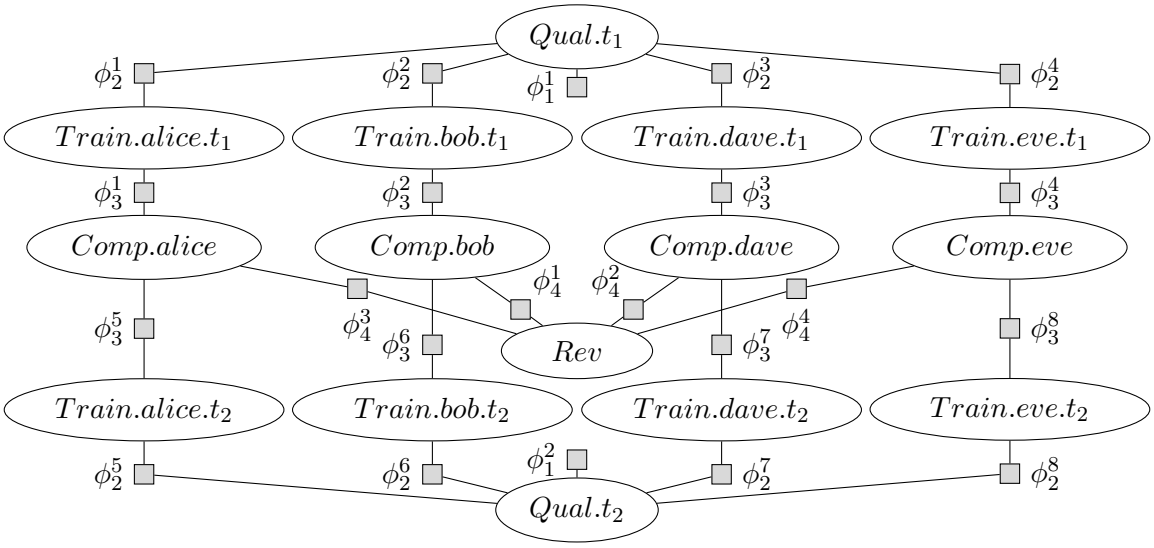

Figure 1: A toy example of an FG modelling the interplay of a company's revenue and its employees' competences, which, in turn, can be improved by training employees with a specific training program. We omit the input-output pairs of the factors for brevity.

$Train.e_j.t_i$ are Boolean. Factors ϕ_1^i encode the prior probability distribution for the quality of a training program, factors ϕ_2^i encode the relationship between the quality of a training program and an employee being trained with that program, factors ϕ_3^i encode the relationship between an employee being trained by a specific training program and the competence of that employee, and factors ϕ_4^i encode the relationship between the competence of an employee and the revenue of the company. The input-output pairs of the factors are omitted for brevity.

We continue to define PFGs, first introduced by Poole (2003), which combine probabilistic models and first-order logic. In particular, PFGs use logical variables (logvars) as parameters in randvars to represent sets of indistinguishable randvars. Each set of indistinguishable randvars is represented by a so-called parameterised randvar (PRV), defined as follows.

Definition 2 (Parameterised Random Variable) *Let $\boldsymbol{R}$ be a set of randvar names, $\boldsymbol{L}$ a set of logvar names, and $\boldsymbol{D}$ a set of constants. All sets are finite. Each logvar L has a domain $\mathcal{D}(L) \subseteq \boldsymbol{D}$. A* constraint *is a tuple $(\mathcal{X}, C_{\mathcal{X}})$ of a sequence of logvars $\mathcal{X} = (X_1, \ldots, X_n)$ and a set $C_{\mathcal{X}} \subseteq \times_{i=1}^{n} \mathcal{D}(X_i)$. The symbol $\top$ for C marks that no restrictions apply, i.e., $C_{\mathcal{X}} = \times_{i=1}^{n} \mathcal{D}(X_i)$. A* PRV *$R(L_1, \ldots, L_n)$, $n \geq 0$, is a syntactical construct of a randvar $R \in \boldsymbol{R}$ possibly combined with logvars $L_1, \ldots, L_n \in \boldsymbol{L}$ to represent a set of randvars. If $n = 0$, the PRV is parameterless and forms a propositional randvar. A PRV A (or logvar L) under constraint C is given by $A_{|C}$ ($L_{|C}$), respectively. We may omit $|\top$ in $A_{|\top}$ or $L_{|\top}$. The term $\mathcal{R}(A)$ denotes the possible values (range) of a PRV A. An* event *$A = a$ denotes the occurrence of PRV A with range value $a \in \mathcal{R}(A)$.*

Example 2 *Consider $\boldsymbol{R} = \{Qual, Train, Comp, Rev\}$ for quality, training, competence, and revenue, respectively, $\boldsymbol{L} = \{E, T\}$ with $\mathcal{D}(E) = \{alice, bob, dave, eve\}$ (employees) and $\mathcal{D}(T) = \{t_1, t_2\}$ (training programs), combined into PRVs $Qual(T)$, $Train(E, T)$, $Comp(E)$, and Rev.*

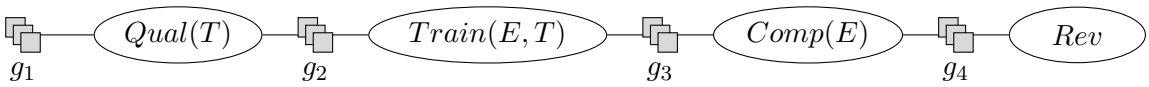

Figure 2: A visualisation of a PFG entailing equivalent semantics as the FG shown in Fig. 1. Each parfactor represents a group of factors and each PRV represents a group of randvars.

A parametric factor (parfactor) describes a function, mapping argument values to positive real numbers, of which at least one is non-zero.

Definition 3 (Parfactor) *Let Φ denote a set of factor names. We denote a* parfactor *g by $\phi(\mathcal{A})_{|C}$ with $\mathcal{A} = (A_1, \ldots, A_n)$ being a sequence of PRVs, ϕ: $\times_{i=1}^{n} \mathcal{R}(A_i) \mapsto \mathbb{R}^+$ being a function with name $\phi \in \Phi$ mapping argument values to a positive real number called* potential*, and C being a constraint on the logvars of $\mathcal{A}$. We may omit $|\top$ in $\phi(\mathcal{A})_{|\top}$. The term $lv(Y)$ refers to the logvars in some element Y, a PRV, a parfactor, or sets thereof. The term $gr(Y_{|C})$ denotes the set of all instances (groundings) of Y with respect to constraint C.*

Example 3 *Take a look at the parfactor $g_3 = \phi_3(Train(E,T), Comp(E))_{|\top}$. Assuming the same ranges of the PRVs and the same domains of the logvars as in Examples 1 and 2, g_3 specifies $2 \cdot 3 = 6$ input-output pairs $\phi_3(true, low) = \varphi_1$, $\phi_3(true, medium) = \varphi_2$, $\phi_3(true, high) = \varphi_3$, and so on with $\varphi_i \in \mathbb{R}^+$. Further, $lv(g_3) = \{E, T\}$ and $gr(g_{3_\top}) = \{\phi_3(Train(alice, t_1), Comp(alice)), \ldots, \phi_3(Train(eve, t_2), Comp(eve))\}$. Thus, in this specific example, the parfactor g_3 is able to represent a set of eight factors.*

A PFG is then built from a set of parfactors $\{g_1, \ldots, g_m\}$.

Definition 4 (Parametric Factor Graph) *A* PFG *$G = (\boldsymbol{V}, \boldsymbol{E})$ is a bipartite graph with node set $\boldsymbol{V} = \boldsymbol{A} \cup \boldsymbol{G}$ where $\boldsymbol{A} = \{A_1, \ldots, A_n\}$ is a set of PRVs and $\boldsymbol{G} = \{g_1, \ldots, g_m\}$ is a set of parfactors. A PRV A_i and a parfactor g_j are connected via an edge in G (i.e., $\{A_i, g_j\} \in \boldsymbol{E}$) if A_i appears in the argument list of g_j. The semantics of G is given by grounding and building a full joint distribution. With Z as the normalisation constant and $\mathcal{A}_k$ denoting the randvars connected to ϕ_k, G represents the full joint distribution*

$$P_G = \frac{1}{Z} \prod_{g_j \in \boldsymbol{G}} \prod_{\phi_k \in gr(g_j)} \phi_k(\mathcal{A}_k).$$

Example 4 *Figure 2 depicts a PFG G consisting of four parfactors $g_1 = \phi_1(Qual(T))$, $g_2 = \phi_2(Qual(T), Train(E,T))$, $g_3 = \phi_3(Train(E,T), Comp(E))$, and $g_4 = \phi_4(Comp(E), Rev)$. Assuming that both the ranges of the PRVs and the domains of the logvars follow Examples 1 to 3, G is a lifted representation entailing equivalent semantics as the FG shown in Fig. 1. Each parfactor $g_1, \ldots, g_4$ represents a group of factors $\phi_1^i, \ldots, \phi_4^i$, respectively, and each PRV $Qual$, $Train$, $Comp$, and Rev represents a group of randvars.*

The underlying assumption here is that there are indistinguishable objects, in this specific example employees, which can be represented by a representative. In particular, the assumption is that the

competence of every employee has the same influence on the company's revenue, i.e., all factors ϕ_4^i encode the same mappings (and the same holds for the factors ϕ_1^i, ϕ_2^i, and ϕ_3^i, meaning training programs are indistinguishable as well). In other words, it is relevant for the company how many employees are competent but it does not matter which exact employees are competent. Note that the definition of PFGs also includes FGs, as every FG is a PFG containing only parameterless randvars.

In the following, we extend PFGs to incorporate causal knowledge, represented by directed edges defining cause-effect relationships between PRVs.

3. Parametric Causal Factor Graphs

PFGs are well-established models for which lifted inference algorithms exist to allow for tractable probabilistic inference with respect to domain sizes. Even though Frey (2003) introduces directed FGs to allow for representing causal knowledge on a ground level, PFGs have not yet been extended to incorporate causal knowledge on a lifted level.

Therefore, we now introduce PCFGs as an extension of PFGs incorporating causal knowledge and give a formal semantics of interventions therein. A PCFG extends a PFG by incorporating causal knowledge in form of directed edges—that is, all edges between two PRVs (via a parfactor) are directed and describe a cause-effect relationship. For example, an edge $A_1 \to A_2$ indicates that A_1 is a cause of A_2 and, consequently, A_2 is an effect of A_1. In particular, in a PCFG, each parfactor is connected to a single child and zero or more parents, matching the definition of directed FGs in the ground case given by Frey (2003). Further, as commonly required in directed graphical models such as causal Bayesian networks, we restrict a PCFG to be acyclic.

Definition 5 (Parametric Causal Factor Graph) *A* PCFG *is a fully directed graph $G = (\boldsymbol{V}, \boldsymbol{E})$ with node set $\boldsymbol{V} = \boldsymbol{A} \cup \boldsymbol{G}$ where $\boldsymbol{A} = \{A_1, \ldots, A_n\}$ is a set of PRVs and $\boldsymbol{G} = \{g_1, \ldots, g_m\}$ is a set of* directed parfactors. *A directed parfactor $g = \phi(\mathcal{A})^{\to A_i}_{|C}$ with $\mathcal{A} = (A_1, \ldots, A_k)$ being a sequence of PRVs, $\phi\colon \times_{i=1}^{k} \mathcal{R}(A_i) \mapsto \mathbb{R}^+$ being a function, and C being a constraint on the logvars of $\mathcal{A}$, maps its argument values to a positive real number (potential). Again, we may omit $|\top$ in $\phi(\mathcal{A})^{\to A_i}_{|\top}$. $A_i \in \mathcal{A}$ denotes the* child *of $\phi(\mathcal{A})^{\to A_i}$ whereas all $A_j \in \mathcal{A}$ with $j \neq i$ are the* parents *of $\phi(\mathcal{A})^{\to A_i}$. For each directed parfactor g, there are edges $(g, A_i) \in \boldsymbol{E}$ and $(A_j, g) \in \boldsymbol{E}$ (for all $A_j \neq A_i$). A PCFG is an acyclic graph, that is, there is no sequence of edges $(g_1, A_1), (A_1, g_2), \ldots, (g_\ell, A_\ell), (A_\ell, g_1)$ in $\boldsymbol{E}$. The semantics of G is given by grounding and building a full joint distribution, identical to the semantics of a PFG, i.e., with Z as the normalisation constant, $\mathcal{A}_k$ denoting the randvars connected to ϕ_k, and $A_i^k \in \mathcal{A}_k$ specifying the child of ϕ_k, G represents*

$$P_G = \frac{1}{Z} \prod_{g_j \in \boldsymbol{G}} \prod_{\phi_k \in gr(g_j)} \phi_k(\mathcal{A}_k)^{\to A_i^k}.$$

Example 5 *Consider the PCFG G depicted in Fig. 3. G represents the same full joint probability distribution as the PFG shown in Fig. 2. In particular, both models are identical except for the fact that G contains directed edges instead of undirected edges between parfactors and PRVs. Each parfactor represents a group of directed factors and thus, grounding G results in a directed FG. Following previous examples by assuming $\mathcal{D}(T) = \{t_1, t_2\}$, for example $g_1 = \phi_1(Qual(T))^{\to Qual(T)}$ represents $gr(g_1) = \{\phi_1(Qual(t_1))^{\to Qual(t_1)}, \phi_1(Qual(t_2))^{\to Qual(t_2)}\}$.*

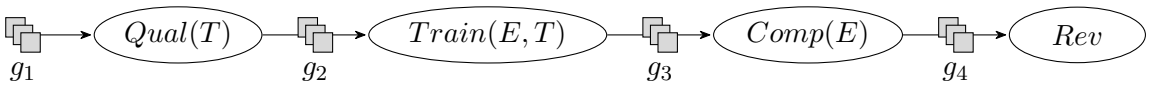

Figure 3: An illustration of a PCFG encoding the same full joint probability distribution as the PFG given in Fig. 2. The only difference between the PCFG and the PFG is that the PCFG contains directed edges instead of undirected edges between PRVs and parfactors.

In the following, we denote the parents of a PRV A by $\mathrm{Pa}_G(A) = \{\phi^{\to A_i} \mid A = A_i\}$ and the child of a parfactor ϕ by $\mathrm{Ch}_G(\phi(\mathcal{A})^{\to A_i}) = A_i$ in a PCFG G. If the context is clear, we omit the subscript G. Before we define the semantics of an intervention in a PCFG, we briefly revisit the notion of d-separation in directed acyclic graphs and afterwards apply it to PCFGs.

3.1. d-Separation in Parametric Causal Factor Graphs

The notion of d-separation (Pearl, 1986) provides a graphical criterion to test for conditional independence in directed acyclic graphs. Frey (2003) translates the notion of d-separation to directed FGs. We build on the definition of d-separation in directed FGs provided by Frey (2003) to define d-separation in PCFGs on a ground level.

Definition 6 (d-separation) *Let $G = (\boldsymbol{A} \cup \boldsymbol{G}, \boldsymbol{E})$ be a PCFG. Given three disjoint sets of randvars $\boldsymbol{X}$, $\boldsymbol{Y}$, and $\boldsymbol{Z}$ (subsets of $\bigcup_{A \in \boldsymbol{A}} gr(A)$), we say that $\boldsymbol{X}$ and $\boldsymbol{Y}$ are conditionally independent given $\boldsymbol{Z}$, written as $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y} \mid \boldsymbol{Z}$, if the nodes in $\boldsymbol{Z}$ block all paths from the nodes in $\boldsymbol{X}$ to the nodes in $\boldsymbol{Y}$ in the directed FG obtained by grounding G. A path is a connected sequence of edges $(R_1, g_1), \ldots, (R_\ell, g_\ell)$ and is not restricted to follow the arrow directions of the edges. Note that it is therefore also possible for a path to pass from a parent of a factor to another parent of the factor. A path is blocked by the nodes in $\boldsymbol{Z}$ if*

1. *the path contains a variable from $\boldsymbol{Z}$, or*
2. *the path passes from a parent of a directed factor ϕ to another parent of ϕ, and neither the child of ϕ nor any of its descendants are in $\boldsymbol{Z}$.*

The semantics of d-separation in PCFGs is defined on a ground level. However, it is possible to check for d-separation on a lifted level without having to ground the PCFG. We give a short motivational example to illustrate this idea and leave the algorithmic details for future work.

Example 6 *Consider again the PCFG depicted in Fig. 3 and assume we want to check whether $Qual(t_1) \perp\!\!\!\perp Comp(bob) \mid Train(bob, t_1)$ holds. In this case, we have assigned $T = t_1$ and $E = bob$, so we only need to examine paths involving this particular assignment of logvars. More specifically, all PRVs on the path with overlapping logvars, i.e., all PRVs having T or E as a logvar, are bound to the same assignment. Therefore, all paths from $Qual(t_1)$ to $Comp(bob)$ pass through $Train(bob, t_1)$, meaning the conditional independence statement in question holds. Note that if there were other paths involving PRVs with non-overlapping logvars, i.e., logvars not involved in the sets $\boldsymbol{X}$ and $\boldsymbol{Y}$, all randvars represented by those PRVs need to be in $\boldsymbol{Z}$ to block those paths.*

We close this subsection with a final remark on d-separation in PCFGs. In contrast to Bayesian networks, the conditional independence statements induced by a PCFG G are not guaranteed to be compatible with arbitrary potential mappings of the directed parfactors in G (i.e., it is not possible to specify the edge directions and arbitrary potential mappings independent of each other). In other words, the conditional independence statements induced by the graph structure of G must be encoded in the potential mappings of the directed parfactors in G. For example, if the graph structure of G induces a conditional independence statement $X \perp\!\!\!\perp Y \mid Z$ for the randvars X, Y, and Z, the potential mappings of the directed parfactors in G must be specified such that $P(X, Y \mid Z) = P(X \mid Z) \cdot P(Y \mid Z)$ holds[1]. To ensure that the conditional independence statements induced by the graph structure of G are compatible with the potential mappings of the directed parfactors in G, it is sufficient to specify the potential mappings for each directed parfactor $\phi(A_1, \ldots, A_k)^{\to A_i}$ such that all assignments $(a_1, \ldots, a_k)$ which differ only at position i sum up to one (which is equivalent to each row in a conditional probability table in a Bayesian network summing up to one). Specifying the potential mappings in this way ensures that each directed parfactor $\phi(A_1, \ldots, A_k)^{\to A_i}$ encodes a conditional probability distribution that corresponds to exactly one conditional probability distribution in an equivalent Bayesian network. Consequently, the conditional independence statements induced by the graph structure of G are compatible with the potential mappings of the directed parfactors in G because there exists a Bayesian network that induces the same conditional independence statements as G and encodes the same underlying full joint probability distribution as G. Note that this issue arises only when specifying the model by hand, i.e., in case the model is learned from data, the conditional independence statements induced by the data are automatically reflected both in the graph structure and the potential values.

The concept of d-separation is important for the computation of the effect of an intervention in the sense that all non-causal paths, so-called backdoor paths, need to be blocked to remove spurious effects. We next show how these backdoor paths are blocked when performing an intervention and give a formal semantics of an intervention in a PCFG.

3.2. Semantics of Interventions in Parametric Causal Factor Graphs

To correctly handle the semantics of an action, for example in the setting of a decision-making agent planning for the best action to take, we have to differentiate between *seeing* (conditioning) and *doing* (intervention). Let us take a look at the PCFG shown in Fig. 3 again. If we observe (see) an event $Train(bob, t_1) = true$, our belief about the probability distribution of $Qual(t_1)$ might change. More specifically, the probability of t_1 having a high quality might be higher when observing $Train(bob, t_1) = true$ than without the observation under the assumption that the probability of training an employee increases if the quality of a training program is high. However, if we are interested in the effect an action setting $Train(bob, t_1)$ to $true$, denoted as $do(Train(bob, t_1) = true)$, has on the remaining PRVs, we have to ensure that the belief about the probability of t_1 having a high quality remains unchanged as the action itself has no influence on the probability distribution of $Qual(t_1)$. Therefore, it is crucial to avoid the propagation of information against the edge directions whenever we are interested in the effect of an action. That is, if we are interested in the effect a specific randvar R' has on another randvar R, all so-called backdoor paths from R to R' must be

1. In a Bayesian network, the numbers in all conditional probability tables can be specified arbitrarily (except for the only restriction that each row must sum to one) because the structure of the conditional probability tables enforces that all conditional independence statements induced by the Bayesian network's structure are also encoded in the numbers, i.e., $P(X, Y \mid Z) = P(X \mid Z) \cdot P(Y \mid Z)$ holds if and only if $X \perp\!\!\!\perp Y \mid Z$.

blocked. A backdoor path is a non-causal path, i.e., a backdoor path from R to R' is a path that remains after removing all outgoing edges of R.

To account for backdoor paths and correctly handle the semantics of an action, we employ the notion of an intervention. We first define the notion of an intervention on randvars and later extend it to allow for do-expressions on PRVs. An intervention on a randvar R, denoted as $do(R = r)$ with $r \in \mathcal{R}(R)$, changes the structure of a PCFG by removing all parent edges of R and setting R to the value r. By removing the parent edges, all backdoor paths are removed. Formally, the semantics of an intervention in a PCFG is defined as below, following the definition of an intervention in Bayesian networks provided by Pearl et al. (2016).

Definition 7 (Intervention) *Let $\boldsymbol{R} = \{R_1, \ldots, R_n\}$ be the set of randvars obtained by grounding a PCFG $G = (\boldsymbol{A} \cup \boldsymbol{G}, \boldsymbol{E})$, i.e., $\boldsymbol{R} = \bigcup_{A \in \boldsymbol{A}} gr(A)$. An intervention $do(R_1 = r_1, \ldots, R_k = r_k)$ changes the underlying probability distribution such that each factor $\phi(R'_1, \ldots, R'_i, \ldots, R'_\ell)^{\to R'_i}$ with $R'_i \in \{R_1, \ldots, R_k\}$ is replaced by a factor $\phi'(R'_1, \ldots, R'_i, \ldots R'_\ell)^{\to R'_i}$ with*

$$\phi'(R'_1 = r'_1, \ldots, R'_i = r'_i, \ldots, R'_\ell = r'_\ell)^{\to R'_i} = \begin{cases} 1 & \text{if } r_i = r'_i \\ 0 & \text{if } r_i \neq r'_i. \end{cases}$$

All $\phi(R'_1, \ldots, R'_i, \ldots, R'_\ell)^{\to R'_i}$ with $R'_i \notin \{R_1, \ldots, R_k\}$ remain unchanged.

By fixing the values of all parent factors, all parent influences are (virtually) removed from the model and hence initial backdoor paths are (virtually) removed from the model as well. Having defined the semantics of an intervention in a PCFG, we are now interested in efficiently computing interventional distributions, i.e., the result of queries that contain do-expressions. Note that in a PCFG, all causal effects are identifiable per definition and thus, we do not have to rewrite a query containing do-expressions according to the do-calculus (Pearl, 1995) to obtain an equivalent query free of do-expressions. In particular, we do not estimate causal effects from observed data but instead compute them in a fully specified model as every PCFG encodes a full joint probability distribution which we can modify according to the definition of an intervention and afterwards query the modified distribution to answer any query containing do-expressions.

We next introduce the LCI algorithm, which handles interventions in PCFGs efficiently by directly applying the semantics of an intervention on a lifted level.

4. Efficient Causal Effect Computation in Parametric Causal Factor Graphs

Now that we have introduced PCFGs, we study the problem of efficiently computing the effect of interventions in PCFGs. A major advantage of using PCFGs instead of propositional models such as causal Bayesian networks is that we mostly do not have to fully ground the model to compute the effect of interventions. Consider again the PCFG illustrated in Fig. 3 and assume we want to compute the interventional distribution $P(Rev \mid do(Train(bob, t_1) = true))$ in G. Note that when intervening on a randvar, we have to treat it differently than other randvars in the same group on which we do not intervene. An intervention $do(Treat(bob, t_1) = true)$ sets the value of $Treat(bob, t_1)$ to $true$ and thus, we have to treat bob differently from $alice$, $dave$, and eve—in other words, not all employees are indistinguishable anymore. Nevertheless, and this is the crucial point, we can still treat $alice$, $dave$, and eve as indistinguishable when computing the interventional distribution.

Algorithm 1: Lifted Causal Inference

Input : A PCFG $G = (\boldsymbol{A} \cup \boldsymbol{G}, \boldsymbol{E})$, and a query $P(R_1, \ldots, R_\ell \mid do(R'_1 = r'_1, \ldots, R'_k = r'_k))$ with $\{R_1, \ldots, R_\ell\} \subseteq \bigcup_{A \in \boldsymbol{A}} gr(A)$ and $\{R'_1, \ldots, R'_k\} \subseteq \bigcup_{A \in \boldsymbol{A}} gr(A)$.

Output: The interventional distribution $P(R_1, \ldots, R_\ell \mid do(R'_1 = r'_1, \ldots, R'_k = r'_k))$.

$G' \leftarrow$ Split parfactors in G based on each $R'_i \in \{R'_1, \ldots, R'_k\}$
foreach $R'_i \in \{R'_1, \ldots, R'_k\}$ **do**
  foreach $\phi(A_1, \ldots, A_z)^{\to R'_i} \in \mathrm{Pa}_{G'}(R'_i)$ **do**
    foreach *assignment* $(a_1, \ldots, a_z) \in \mathcal{R}(A_1) \times \cdots \times \mathcal{R}(A_z)$ **do**
      Set $\phi(a_1, \ldots, a_z) = \begin{cases} 1 & \text{if } (a_1, \ldots, a_z) \text{ assigns } R'_i = r'_i \\ 0 & \text{if } (a_1, \ldots, a_z) \text{ assigns } R'_i \neq r'_i \end{cases}$
    end
  end
end
$P \leftarrow$ Call LVE to compute $P(R_1, \ldots, R_\ell)$ in G'
return P

4.1. The Lifted Causal Inference Algorithm

We now introduce the LCI algorithm to compute the interventional distribution $P(R_1, \ldots, R_\ell \mid do(R'_1 = r'_1, \ldots, R'_k = r'_k))$ in a PCFG G. The entire LCI algorithm is shown in Alg. 1.

First, LCI splits the parfactors in G based on the intervention variables $R'_i \in \{R'_1, \ldots, R'_k\}$. In particular, splitting parfactors in G results in a modified PCFG G' entailing equivalent semantics as G (De Salvo Braz et al., 2005). The procedure of splitting a parfactor works as follows. Recall that $R'_i = A(L_1 = l_1, \ldots, L_j = l_j)$, $l_1 \in \mathcal{D}(L_1), \ldots, l_j \in \mathcal{D}(L_j)$, is a particular instance of a PRV $A(L_1, \ldots, L_j)$, that is, it holds that $R'_i \in gr(A)$. The idea behind the splitting procedure is that we would like to separate $gr(A)$ into two sets $gr(A) \setminus \{R'_i\}$ and $\{R'_i\}$, as R'_i has to be treated differently than the remaining instances of A. Therefore, every parfactor g for which there is an instance $\phi \in gr(g)$ such that R'_i appears in the argument list of ϕ is split. Formally, splitting a parfactor g replaces g by two parfactors $g'_{|C'}$ and $g''_{|C''}$ and adapts the constraints of $g'_{|C'}$ and $g''_{|C''}$. The constraints C' and C'' are altered such that the inputs of $g'_{|C'}$ are restricted to all sequences that contain R'_i and the inputs of $g''_{|C''}$ are restricted to the remaining input sequences. After the splitting procedure, the semantics of the model remains unchanged as the groundings of G' are still the same as the groundings of the initial model G—they are just arranged differently across the sets of ground instances. Having completed the split of all respective parfactors, LCI next modifies the parents of R'_i, i.e., the underlying probability distribution encoded by G' is modified according to the semantics of the intervention $do(R'_i = r'_i)$ with $r'_i \in \mathcal{R}(R'_i)$. More specifically, as R'_i is fixed on r'_i, all parents $\phi \in \mathrm{Pa}_{G'}(R'_i)$ of R'_i are altered such that all input sequences assigning $R'_i = r'_i$ map to the potential value one while all other input sequences map to zero. Finally, LCI computes the result for $P(R_1, \ldots, R_\ell)$ in the modified model G', which is equivalent to the result for $P(R_1, \ldots, R_\ell \mid do(R'_1 = r'_1, \ldots, R'_k = r'_k))$ in the original model G. To perform query answering in G', LVE can be applied to G' by simply ignoring the edge directions as the semantics of a PFG and a PCFG are defined identically.

Before we continue to examine the correctness of Alg. 1, we take a look at an example.

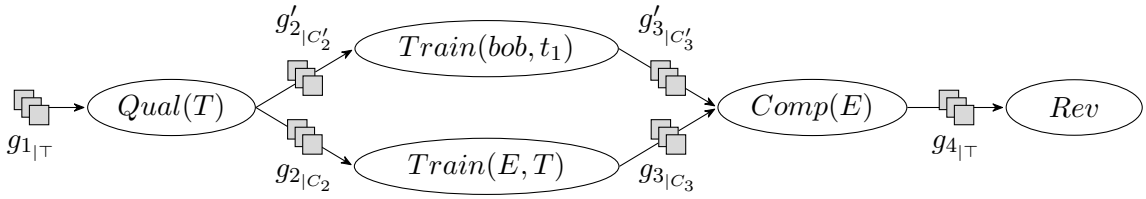

Figure 4: A visualisation of the modified PCFG obtained after altering the PCFG shown in Fig. 3 by splitting g_2 and g_3 to separate $Train(bob, t_1)$ from $Train(E, T)$. Here, the constraints C_2 as well as C_3 include all instances of $Train(E, T)$ except for $Train(bob, t_1)$ and C'_2 as well as C'_3 are restricted to the single instance $Train(bob, t_1)$ of $Train(E, T)$. Note that the graph size remains significantly smaller than for the fully grounded model.

Example 7 *Consider again the PCFG G shown in Fig. 3 and assume we would like to compute $P(Rev \mid do(Train(bob, t_1)) = true)$. As $Train(bob, t_1)$ is a particular instance of $Train(E, T)$, we have to split the parfactors g_2 and g_3 while g_1 as well as g_4 keep $\top$ as their constraint. Figure 4 shows the modified PCFG G' obtained after splitting g_2 and g_3 based on the intervention on $Train(bob, t_1)$. In G', $Train(bob, t_1)$ is now a separate node in the graph, connected to two newly introduced parfactors. In particular, g_2 has been replaced by two parfactors $g_{2_{|C_2}}$ and $g'_{2_{|C'_2}}$ with constraints $C_2 = ((E, T), \{(alice, t_1), (alice, t_2), (bob, t_2), (dave, t_1), (dave, t_2), (eve, t_1), (eve, t_2)\})$ and $C'_2 = ((E, T), \{(bob, t_1)\})$. In other words, $g_{2_{|C_2}}$ is restricted to all instances of $Train(E, T)$ except for $Train(bob, t_1)$ and $g'_{2_{|C'_2}}$ is restricted to the instance $Train(bob, t_1)$ of $Train(E, T)$. Analogously, g_3 has been replaced by two parfactors $g_{3_{|C_3}}$ and $g'_{3_{|C'_3}}$. To incorporate the semantics of $do(Train(bob, t_1)) = true$, LCI next modifies the parents of $Train(bob, t_1)$, i.e., LCI modifies $g'_{2_{|C'_2}}$ in this example. More specifically, $g'_{2_{|C'_2}}(Qual(t_1) = q, Train(bob, t_1) = true)$ is set to one and $g'_{2_{|C'_2}}(Qual(t_1) = q, Train(bob, t_1) = false)$ is set to zero for all $q \in \mathcal{R}(Qual(t_1))$. Finally, LVE can be run to compute $P(Rev)$ in G', which is equivalent to the interventional distribution $P(Rev \mid do(Train(bob, t_1)) = true)$ in the original model G.*

Due to the splitting of parfactors, it might be the case that there are PRVs in G' having more parents than they previously had in the original model G, as with $Comp(E)$ in Fig. 4. The semantics of the model, however, remains unchanged because $\bigcup_{g \in \boldsymbol{G}} gr(g) = \bigcup_{g \in \boldsymbol{G}'} gr(g)$. Given the way we specified the semantics of an intervention in a PCFG, it immediately follows that LCI correctly computes the effect of interventions. In particular, as LCI directly applies Theorem 7 by setting the parent factors of all variables we intervene on accordingly, the semantics of the modified model G' is equivalent to the semantics of interventions from Theorem 7 and thus, LCI is correct.

Corollary 8 *Algorithm 1 computes the interventional distribution according to Theorem 7.*

Moreover, directly applying Theorem 7 allows LCI to exploit the established LVE algorithm. By deploying LVE, LCI is able to perform tractable inference with respect to domain sizes for all PCFGs in the class of domain-liftable models (Taghipour et al., 2013b) if the *do*-expressions in the query do not ground a domain. The class of domain-liftable models includes all PCFGs containing

only parfactors with at most two logvars and all PCFGs containing only PRVs having at most one logvar.

Corollary 9 *Algorithm 1 performs tractable probabilistic inference with respect to domain sizes for the class of domain-liftable models if the do-expressions in the query do not ground a domain.*

A more fine-grained complexity analysis of LVE is provided in (Taghipour, 2013). Note that the loops in Lines 2 to 4 of Alg. 1 do not influence the overall run time complexity of LCI as they iterate over potential mappings that must be considered anyway during inference (both by LVE and its propositional counterpart variable elimination).

To summarise, LCI is a simple, yet effective algorithm to perform lifted causal inference, even for interventions on large groups of randvars, as we investigate next.

4.2. Handling Interventions on Groups of Random Variables

LCI is able to handle both interventions on a single (ground) randvar as well as interventions on a conjunction of multiple randvars efficiently. In particular, when intervening on multiple randvars at the same time, LCI is able to treat those randvars as a group. For example, recall the employee example and assume we want to train multiple employees simultaneously as a training program is mostly offered not only for a single employee but for a group of employees. Then, it is not necessary to split all trained employees into separate groups—it is sufficient to differentiate between trained employees and all remaining employees. Formally, the interventions $do(R'_1 = r'_1, \ldots, R'_k = r'_k)$ on an arbitrary set of randvars $\{R'_1, \ldots, R'_k\}$ can thus efficiently be handled by splitting the parfactors in G such that all R'_i that are represented by the same PRV A and set to the same value r'_i remain grouped, equal to splitting on constraints in LVE. More specifically, LCI needs just a single split per group and thus avoids manipulating the parents of each individual randvar separately. Furthermore, it is also possible to intervene on a PRV (instead of intervening on a randvar). The semantics of an intervention on a PRV A is given by $do(A = a) = do(R_1 = a, \ldots, R_k = a)$ with $gr(A) = \{R_1, \ldots, R_k\}$. Again, LCI is able to treat all randvars represented by A as a group and therefore is not required to split the group. In contrast, in a propositional model, every object has to be treated individually and therefore the parents for each randvar need to be manipulated separately.

Next, we investigate the practical performance of PCFGs and, in particular, the LCI algorithm for the computation of interventional distributions.

5. Experiments

In this section, we evaluate the run times needed to compute the result of interventional queries in Bayesian networks, directed FGs, and PCFGs. For our experiments, we use a slightly modified version of the PCFG given in Fig. 3 which can directly be translated into a Bayesian network without having to combine multiple parent factors into a single conditional probability table. More specifically, to obtain the corresponding directed FG, we simply ground the PCFG and to obtain the equivalent Bayesian network, we use the transformation from directed FG to Bayesian network given by Frey (2003). Note that the PCFG used in our experiments to demonstrate the practical efficiency of lifted causal inference is rather small with four parfactors and PRVs, respectively, and the gain we obtain from lifted inference further increases with models consisting of more PRVs.

We test the required run time for each of the three graphical models on different graph sizes by setting the domain size of the employees to $d = 8, 16, 32, \ldots, 4096$ and having a single training

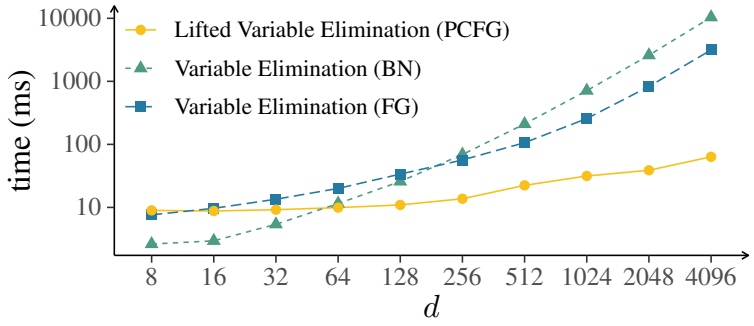

Figure 5: A comparison of the run times required to compute interventional distributions on different graphical models encoding equivalent full joint probability distributions.

program for each choice of d (i.e., $|\mathcal{D}(E)| = d$ and $|\mathcal{D}(T)| = 1$). Figure 5 shows the run times needed to compute an interventional distribution for a single intervention in the modified graph when running variable elimination on the directed FG, variable elimination on the Bayesian network, and LVE on the PCFG. The variable elimination algorithm is the propositional counterpart of LVE and operates on a propositional (ground) model, such as a Bayesian network or an FG. Consequently, variable elimination considers every object represented by a randvar (e.g., every employee) individually for computations, independent of whether there are objects behaving exactly the same. In contrast, LVE treats identically behaving objects as a group by using a representative for computations instead of considering each of those objects separately. The results emphasise that the LCI algorithm, which internally exploits LVE, overcomes scalability issues for large domain sizes as the run time of LVE, in contrast to the run times of variable elimination on the Bayesian network and the directed FG, does not exponentially increase with d (y-axis is log-scaled). To conclude, PCFGs not only provide us with expressive probabilistic graphical models for relational domains but also enable us to drastically speed up causal inference by reasoning over sets of indistinguishable objects.

6. Conclusion

We introduce PCFGs to combine lifted probabilistic inference in relational domains with causal inference, thereby allowing for lifted causal inference. To leverage the power of lifted inference for causal effect computation, we present the LCI algorithm, which operates on a lifted level and thus allows us to drastically speed up causal inference. LCI is a simple, yet effective algorithm to compute the effect of (multiple simultaneous) interventions, and builds on the well-founded LVE algorithm, thereby allowing LCI to be plugged into parameterised decision models (Gehrke et al., 2019b) to compute the maximum expected utility in accordance with Pearl (2009).

PCFGs open up interesting directions for future work. A basic problem is to learn a PCFG directly from a relational database. Following up on learning PCFGs from data, another constitutive problem for future research is to relax the assumption of having a fully directed PCFG at hand, i.e., to allow PCFGs to contain both directed and undirected edges at the same time and investigate the implications for answering causal queries.

Acknowledgments

This work is partially funded by the BMBF project AnoMed 16KISA057 and 16KISA050K, and is also partially supported by the Medical Cause and Effects Analysis (MCEA) project and partially funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2176 "Understanding Written Artefacts: Material, Interaction and Transmission in Manuscript Cultures", project no. 390893796.

References

Ragib Ahsan, David Arbour, and Elena Zheleva. Relational Causal Models with Cycles: Representation and Reasoning. In *Proceedings of the First Conference on Causal Learning and Reasoning (CLeaR-22)*, pages 1–18. PMLR, 2022.

Ragib Ahsan, David Arbour, and Elena Zheleva. Learning Relational Causal Models with Cycles through Relational Acyclification. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI-23)*, pages 12164–12171. AAAI Press, 2023.

Udi Apsel and Ronen I. Brafman. Lifted MEU by Weighted Model Counting. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI-12)*, pages 1861–1867. AAAI Press, 2012.

David Arbour, Dan Garant, and David Jensen. Inferring Network Effects from Observational Data. In *Proceedings of the Twenty-Second ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-16)*, pages 715–724. Association for Computing Machinery, 2016.

Tanya Braun and Marcel Gehrke. Explainable and Explorable Decision Support. In *Proceedings of the Twenty-Seventh International Conference on Conceptual Structures (ICCS-22)*, pages 99–114. Springer, 2022.

Tanya Braun and Ralf Möller. Lifted Junction Tree Algorithm. In *Proceedings of KI 2016: Advances in Artificial Intelligence (KI-16)*, pages 30–42. Springer, 2016.

Tanya Braun and Ralf Möller. Parameterised Queries and Lifted Query Answering. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)*, pages 4980–4986. IJCAI Organization, 2018.

Rodrigo De Salvo Braz, Eyal Amir, and Dan Roth. Lifted First-Order Probabilistic Inference. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 1319–1325. Morgan Kaufmann Publishers Inc., 2005.

Rodrigo De Salvo Braz, Eyal Amir, and Dan Roth. MPE and Partial Inversion in Lifted Probabilistic Variable Elimination. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI-06)*, pages 1123–1130. AAAI Press, 2006.

Brendan J. Frey. Extending Factor Graphs so as to Unify Directed and Undirected Graphical Models. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI-03)*, pages 257–264. Morgan Kaufmann Publishers Inc., 2003.

Brendan J. Frey, Frank R. Kschischang, Hans-Andrea Loeliger, and Niclas Wiberg. Factor Graphs and Algorithms. In *Proceedings of the Thirty-Fifth Annual Allerton Conference on Communication, Control, and Computing*, pages 666–680. Allerton House, 1997.

Marcel Gehrke, Tanya Braun, and Ralf Möller. Lifted Dynamic Junction Tree Algorithm. In *Proceedings of the Twenty-Third International Conference on Conceptual Structures (ICCS-2018)*, pages 55–69. Springer, 2018.

Marcel Gehrke, Tanya Braun, and Ralf Möller. Lifted Temporal Maximum Expected Utility. In *Proceedings of the Thirty-Second Canadian Conference on Artificial Intelligence (CANAI-19)*, pages 380–386. Springer, 2019a.

Marcel Gehrke, Tanya Braun, Ralf Möller, Alexander Waschkau, Christoph Strumann, and Jost Steinhäuser. Lifted Maximum Expected Utility. In *Proceedings of the First International Workshop on Artificial Intelligence in Health (AIH-18)*, pages 131–141. Springer, 2019b.

Marcel Gehrke, Ralf Möller, and Tanya Braun. Taming Reasoning in Temporal Probabilistic Relational Models. In *Proceedings of the Twenty-Fourth European Conference on Artificial Intelligence (ECAI-20)*, pages 2592–2599. IOS Press, 2020.

Jacek Kisyński and David Poole. Constraint Processing in Lifted Probabilistic Inference. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI-09)*, pages 293–302. AUAI Press, 2009.

Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor Graphs and the Sum-Product Algorithm. *IEEE Transactions on Information Theory*, 47:498–519, 2001.

Sanghack Lee and Vasant Honavar. Lifted Representation of Relational Causal Models Revisited: Implications for Reasoning and Structure Learning. In *Proceedings of the UAI 2015 Conference on Advances in Causal Inference*, pages 56–65. CEUR, 2015.

Sanghack Lee and Vasant Honavar. On Learning Causal Models from Relational Data. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*, pages 3263–3270. AAAI Press, 2016.

Sanghack Lee and Vasant Honavar. Towards Robust Relational Causal Discovery. In *Proceedings of The Thirty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI-19)*, pages 345–355. PMLR, 2019.

Marc Maier, Brian Taylor, Huseyin Oktay, and David Jensen. Learning Causal Models of Relational Domains. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10)*, pages 531–538. AAAI Press, 2010.

Marc Maier, Katerina Marazopoulou, David Arbour, and David Jensen. A Sound and Complete Algorithm for Learning Causal Models from Relational Data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence (UAI-13)*, pages 371–380. AUAI Press, 2013.

Brian Milch, Luke S. Zettlemoyer, Kristian Kersting, Michael Haimes, and Leslie Pack Kaelbling. Lifted Probabilistic Inference with Counting Formulas. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI-08)*, pages 1062–1068. AAAI Press, 2008.

Mathias Niepert and Guy Van den Broeck. Tractability through Exchangeability: A New Perspective on Efficient Probabilistic Inference. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI-14)*, pages 2467–2475. AAAI Press, 2014.

Judea Pearl. Fusion, Propagation, and Structuring in Belief Networks. *Artificial Intelligence*, 29: 241–288, 1986.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

Judea Pearl. Causal Diagrams for Empirical Research. *Biometrika*, 82:669–688, 1995.

Judea Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2nd edition, 2009.

Judea Pearl, Madelyn Glymour, and Nicholas P. Jewell. *Causal Inference in Statistics: A Primer*. Wiley, 1st edition, 2016.

David Poole. First-Order Probabilistic Inference. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03)*, pages 985–991. Morgan Kaufmann Publishers Inc., 2003.

Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, 4th edition, 2020.

Babak Salimi, Harsh Parikh, Moe Kayali, Lise Getoor, Sudeepa Roy, and Dan Suciu. Causal Relational Learning. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pages 241–256. Association for Computing Machinery, 2020.

Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. MIT Press, 2nd edition, 2000.

Nima Taghipour. *Lifted Probabilistic Inference by Variable Elimination*. PhD thesis, KU Leuven, 2013.

Nima Taghipour, Daan Fierens, Jesse Davis, and Hendrik Blockeel. Lifted Variable Elimination: Decoupling the Operators from the Constraint Language. *Journal of Artificial Intelligence Research*, 47:393–439, 2013a.

Nima Taghipour, Daan Fierens, Guy Van den Broeck, Jesse Davis, and Hendrik Blockeel. Completeness Results for Lifted Variable Elimination. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics (AISTATS-13)*, pages 572–580. PMLR, 2013b.

John Winn. Causality with Gates. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics (AISTATS-12)*, pages 1314–1322. PMLR, 2012.