Relational Bayesian Networks: a Survey

Manfred Jaeger
Max-Planck-Institut für Informatik
Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany

Abstract

We give an overview of the relational Bayesian network modeling language. First the semantic concept of a random relational structure model is introduced, and then it is shown how such models can be represented with relational Bayesian networks. We consider a number of inference problems for relational Bayesian networks that range from elementary probabilistic queries to the computation of limit probabilities and learning problems. For some of these inference problems fully developed solution algorithms are available, for others we describe solution strategies by reduction to well-established logical inference and numerical optimization problems.

1 Introduction

Numerous proposals have been made for probabilistic models that integrate elements of first-order logical representation and inference with the techniques for tractable probabilistic inference provided by graphical models. Many of these proposals are based on the language of logic programming (Poole 1993, Sato 1995, Ngo & Haddawy 1997, Muggleton 1996, Cussens 1999, Kersting & de Raedt 2001), others on the language of relational databases (Friedman, Getoor, Koller & Pfeffer 1999, Koller 1999).

Formal semantics for these frameworks can in most cases be given by probability distributions on Herbrand bases. This can be a single distribution on one (typically infinite) Herbrand base, or a set of distributions on a class of (typically finite) Herbrand bases. The first type of semantics is usually favored by the logic programming based approaches, whereas the latter underlies the database oriented framework (Friedman et al. 1999, Koller 1999), as well as the relational Bayesian network modeling language (Jaeger 1997, Jaeger 2001).

A more accurate and refined description of the second type of semantics is provided by the definition of a *probabilistic relational model* as given in (Jaeger 2001). In order to prevent a possible confusion with Friedman et al.'s (1999) probabilistic relational models, we here restate this definition introducing a different name. In this definition and in the remainder of this paper we use $\mathrm{Mod}_D(S)$ to denote the set of all relational structures $\mathcal{D}$ that interpret the relations from the vocabulary (or signature) S over the finite domain $D = \{d_1, \ldots, d_n\}$. Also, $\mathrm{Mod}_{\mathit{fin}}(S)$ denotes the class of all finite relational structures for S. In logic programming terminology, $\mathrm{Mod}_D(S)$ is the set of Herbrand interpretations for S over the Herbrand universe $\{d_1, \ldots, d_n\}$.

Definition 1.1 Let S, R be two sets of relation symbols. The elements of S are called the *predefined relations*; the elements of R are called the *probabilistic relations.* A *random relational structure model for S and R* is a partial mapping P that assigns to S-structures $\mathcal{D}$ with finite domain D a probability distribution $P(\mathcal{D})$ over $\mathrm{Mod}_D(R)$. In the sequel we write $P_{\mathcal{D}}$ for $P(\mathcal{D})$, and also call such a single distribution an *instance* of the random relational structure model.

Example 1.2 A *Markov chain* over states $s_1, \ldots, s_l$ defines for every $n \in \mathbb{N}$ a probability distribution on state sequences of length n. This is a random relational structure model for a single binary predefined relation $<$, and unary probabilistic relations $\mathtt{s}_1, \ldots, \mathtt{s}_l$: the distribution $P_{\mathcal{D}}$ is defined whenever $<$ is interpreted in $\mathcal{D}$ as a linear order. If $|\, D \,| = n$, then state sequences of length n can be identified with "colorings" of D by the unary relations $\mathtt{s}_1, \ldots, \mathtt{s}_l$, i.e. interpretations of the $\mathtt{s}_i$ over D in which for every $d \in D$ exactly one relation $\mathtt{s}_i$ is true. For every $\mathcal{E} \in \mathrm{Mod}_D(R)$ then $P_{\mathcal{D}}(\mathcal{E}) = p$ if $\mathcal{E}$ encodes a state sequence of probability p ($p = 0$ if $\mathcal{E}$ is not a coloring).

Example 1.3 A *random graph* is constructed by inserting edges randomly between nodes $d_1, \ldots, d_n$. More precisely, a random graph model is given by defining for each $n \in \mathbb{N}$ a probability distribution on all graphs with n nodes. The most prominent such model is the Erdös-Renyi model, in which for every n is defined an edge probability $p(n)$, and a graph $\mathcal{E}$ with n nodes and k edges has probability $p(n)^k(1 - p(n))^{n^2 - k}$. Such a random graph model is a random relational structure model with $S = \emptyset$ and a single binary probabilistic (edge-) relation $\mathtt{e}$.

Markov chains and random graphs are "pure" mathematical examples for random relational structure models. More "real-world" examples will be given in section 2. It should be noted however, that also dynamic Bayesian networks (Dagum, Galper & Horvitz 1992), hidden Markov models, and (in a slightly less obvious way) stochastic context-free grammars can be formalized as random relational structure models.

In this paper we give a survey of the language of *relational Bayesian networks* (Jaeger 1997) for the representation of random relational structure models. We discuss a number of relevant inference problems that one can formulate for random relational structure models, and their solutions based on relational Bayesian network representations. We review results from (Jaeger 1997, Jaeger 1998*a*, Jaeger 2001), and indicate solution approaches to some new inference and learning problems. In this paper we emphasize the logical nature of relational Bayesian networks, and highlight some of the connections that exist between the investigation of random relational structure models with relational Bayesian network representations, and topics in finite model theory (Ebbinghaus

& Flum 1999). A more practice-oriented account that focuses on algorithmic aspects is given in (Jaeger 2001).

2 Representation

The core instrument for the representation of random relational structure models with relational Bayesian networks is the *probability formula*. There are a number of ways to look at probability formulas. One can see them as a functional programming language for the computation of cpt-entries in Bayesian networks representing particular model instances. Here we shall emphasize their analogy to formulas in predicate logic. A predicate logic formula $\phi(\boldsymbol{v})$ containing symbols from the signatures S and R, and quantifiers from a set Γ (containing the basic first-order quantifiers $\exists, \forall$, but possibly also a number of generalized quantifiers) can be evaluated over a S, R-structure $\mathcal{F}$ for a tuple $\boldsymbol{d} \subseteq D$ to compute a truth value $\phi(\boldsymbol{d})[\mathcal{F}] \in \{\mathit{true}, \mathit{false}\}$ (or, in more standard notation, to decide whether $\mathcal{F} \models \phi(\boldsymbol{d})$ or $\mathcal{F} \not\models \phi(\boldsymbol{d})$). A probability formula $F(\boldsymbol{v})$ for the vocabularies S, R is evaluated in a similar fashion for a tuple $\boldsymbol{d}$ over $\mathcal{F}$, but yields a probability value: $F(\boldsymbol{d})[\mathcal{F}] \in [0,1]$.

The probability-formula equivalent to a quantifier is the combination function:

Definition 2.1 A *combination function* is any function that maps finite multisets with elements from [0,1] into [0,1].

We use braces $\}\!\!\}, \{\!\!\{$ to denote multisets: if $q_i \in [0,1]$ for all i from some index set I, then $\{\!\!\{q_i \mid i \in I\}\!\!\}$ denotes the multiset that contains $|\{i \in I \mid q_i = r\}|$ copies of $r \in [0,1]$. The two most important combination functions for practical modeling problems are

$$\begin{array}{lll} \mathit{noisy\text{-}or}: & \mathit{n\text{-}o}\{\!\!\{q_i \mid i \in I\}\!\!\} & := 1 - \prod_{i \in I}(1 - q_i) \\ \mathit{mean}: & \mathit{mean}\{\!\!\{q_i \mid i \in I\}\!\!\} & := \frac{1}{|I|}\sum_{i \in I} q_i. \end{array}$$

The syntax of probability formulas can now be defined. In this definition we call an *S-constraint* any boolean combination of atomic formulas $s(\boldsymbol{v})$ for symbols $s \in S$, and variables $\boldsymbol{v}$ (no constant symbols are allowed in S-constraints).

Definition 2.2 Let S, R be sets of relation symbols, Γ a set of combination functions. The class of (S, R, Γ)-*probability formulas* is inductively defined as follows.

(i) (Constants) Each $q \in [0,1]$ is a probability formula.

(ii) (Indicator functions) For each $\mathtt{r} \in R$, and every $|\mathtt{r}|$-tuple $\boldsymbol{v}$ of variables, $\mathtt{r}(\boldsymbol{v})$ is a probability formula.

(iii) (Convex combinations) When F_1, F_2, F_3 are probability formulas, then so is $F_1F_2 + (1 - F_1)F_3$.

(iv) (Combination functions) When $F_1, \ldots, F_k$ are probability formulas, $\mathit{comb} \in \Gamma$, $\boldsymbol{v}, \boldsymbol{w}$ are tuples of variables, and $c(\boldsymbol{v}, \boldsymbol{w})$ is an S-constraint, then

$$\mathit{comb}\{\!|F_1, \ldots, F_k \mid \boldsymbol{w}; c(\boldsymbol{v}, \boldsymbol{w})|\!\}$$

is a probability formula.

There is a close correspondence between the construction rules for probability formulas and the construction rules for predicate logic formulas: Constants are the probabilistic extensions of logical constants *true* and *false.* Indicator functions are relational atoms. Convex combinations play the role of Boolean connectives. Finally, as already mentioned, a combination function corresponds to a quantifier (that binds the variables $\boldsymbol{w}$).

Given a S, R-probability formula $F(\boldsymbol{v})$, a S, R -structure $\mathcal{F}$, and $\boldsymbol{d} \subseteq D$, it is straightforward to define the value $F(\boldsymbol{d})[\mathcal{F}] \in [0, 1]$ by induction on the structure of F: for indicator functions $F = \mathtt{r}(\boldsymbol{v})$ one defines $\mathtt{r}(\boldsymbol{d})[\mathcal{F}] = 1$ iff $\mathcal{F} \models \mathtt{r}(\boldsymbol{d})$, and $\mathtt{r}(\boldsymbol{d})[\mathcal{F}] = 0$, else. For combination functions $F = \mathit{comb}\{\!| \ldots |\!\}$, one applies *comb* to the multisets of values $F_i(\boldsymbol{d}, \boldsymbol{d}')[\mathcal{F}]$ for $i = 1, \ldots, k$, and all $\boldsymbol{d}'$ with $\mathcal{F} \models c(\boldsymbol{d}, \boldsymbol{d}')$.

However, we do not wish to evaluate probability formulas over existing S, R-structures, we want to use them to define the probability of different R-structures, given a particular S-structure. This, however, only requires a slight change of perspective: the recursive evaluation of $F(\boldsymbol{d})[\mathcal{F}]$ leads to evaluations of ground atoms $\mathtt{r}(\boldsymbol{d}')$ for certain $\mathtt{r} \in R$ and $\boldsymbol{d}' \subseteq D$. Which of these ground atoms need to be evaluated depends on the interpretation in $\mathcal{F}$ of the relations from S, because the relations in S determine what recursive evaluations of $F_i(\boldsymbol{d}, \boldsymbol{d}')[\mathcal{F}]$ are made in the processing of combination functions. Now writing $\mathcal{D}$ for the restriction of $\mathcal{F}$ to the symbols in S, we can thus write $\mathit{Pa}(F(\boldsymbol{d})[\mathcal{D}])$ for the set of ground R- atoms that will be evaluated in the computation of $F(\boldsymbol{d})[\mathcal{F}]$. If $I(\mathit{Pa}(F(\boldsymbol{d})[\mathcal{D}]))$, now, is the interpretation in $\mathcal{F}$ of the ground atoms in $\mathit{Pa}(F(\boldsymbol{d})[\mathcal{D}])$, then we can also write $F(\boldsymbol{d})[\mathcal{D}, I(\mathit{Pa}(F(\boldsymbol{d})[\mathcal{D}]))]$ for $F(\boldsymbol{d})[\mathcal{F}]$, and use this value as the conditional probability for some ground R-atom $\mathtt{r}(\boldsymbol{d}) \notin \mathit{Pa}(F(\boldsymbol{d})[\mathcal{D}])$.

Our strategy for defining random relational structure models with probability formulas now is simply to assign to every n-ary $\mathtt{r} \in R$ one S, R-probability formula $F_{\mathtt{r}}(v_1, \ldots, v_n)$. We call the resulting set

$$\Phi = \{F_{\mathtt{r}}(v_1, \ldots, v_{|\mathtt{r}|}) \mid \mathtt{r} \in R\} \tag{1}$$

a *Relational Bayesian Network.* If the dependency relation

$$\mathtt{r}(\boldsymbol{d}) \succeq_{\Phi, \mathcal{D}} \mathtt{r}'(\boldsymbol{d}') \quad :\Leftrightarrow \quad \mathtt{r}'(\boldsymbol{d}') \in \mathit{Pa}(F_{\mathtt{r}}(\boldsymbol{d})[\mathcal{D}])$$

on ground R-atoms is acyclic, then Φ defines a probability distribution $P^{\Phi}_{\mathcal{D}}$ on $\mathrm{Mod}_D(R)$ by letting for $\mathcal{E} \in \mathrm{Mod}_D(R)$

$$\begin{aligned} P^{\Phi}_{\mathcal{D}}(\mathcal{E}) := \prod_{\mathtt{r} \in R} & \prod_{\boldsymbol{d}: \mathcal{E} \models \mathtt{r}(\boldsymbol{d})} F_{\mathtt{r}}(\boldsymbol{d})[\mathcal{D}, I(\mathit{Pa}(F(\boldsymbol{d})[\mathcal{D}]))] \\ & \prod_{\boldsymbol{d}: \mathcal{E} \not\models \mathtt{r}(\boldsymbol{d})} (1 - F_{\mathtt{r}}(\boldsymbol{d})[\mathcal{D}, I(\mathit{Pa}(F(\boldsymbol{d}), [\mathcal{D}]))]) \end{aligned} \tag{2}$$

A relational Bayesian network Φ thus represents the random relational structure model $\mathcal{D} \mapsto P^{\Phi}_{\mathcal{D}}$ ($\mathcal{D} \in \mathrm{Mod}_{\mathit{fin}}(S) : \succeq_{\Phi,\mathcal{D}}$ is acyclic).

$$
\begin{aligned}
F_{\mathit{father\text{-}in\text{-}pedigree}}(v) &= \mathit{noisy\text{-}or}\{\!\!\{1 \mid u; \mathit{father}(u,v)\}\!\!\} \\
F_{\mathit{mother\text{-}in\text{-}pedigree}}(v) &= \mathit{noisy\text{-}or}\{\!\!\{1 \mid u; \mathit{mother}(u,v)\}\!\!\} \\
F_{A-from-father}(v) &= \mathit{mean}\{\!\!\{\mathtt{FA}(u), \mathtt{MA}(u) \mid u; \mathit{father}(u,v)\}\!\!\} \\
F_{A-from-mother}(v) &= \mathit{mean}\{\!\!\{\mathtt{FA}(u), \mathtt{MA}(u) \mid u; \mathit{mother}(u,v)\}\!\!\} \\
F_{\mathtt{FA}}(v) &= F_{\mathit{father\text{-}in\text{-}pedigree}}(v) \cdot F_{A-from-father}(v) + (1 - F_{\mathit{father\text{-}in\text{-}pedigree}}(v)) \cdot 1/3 \\
F_{\mathtt{MA}}(v) &= F_{\mathit{mother\text{-}in\text{-}pedigree}}(v) \cdot F_{A-from-mother}(v) + (1 - F_{\mathit{mother\text{-}in\text{-}pedigree}}(v)) \cdot 1/3
\end{aligned}
$$

Table 1: Genetic Example

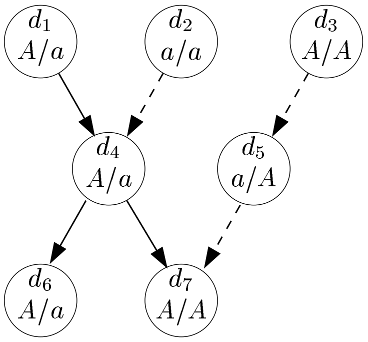

Figure 1: Pedigree

Example 2.3 Figure 1 shows a (partial) pedigree for seven individuals $d_1, \ldots, d_7$. The pedigree is specified using a binary *father* relation (indicated by solid arrows), and a binary *mother* relation (indicated by broken arrows). In this pedigree, both father and mother are known for individuals d_4 and d_7. For all other individuals only one or no parents are known.

Also represented in the pedigree is information on a gene that has two alleles A and a. The notation x/y here represents an ordered genotype and stands for the fact that x was inherited from the father, and y from the mother. This genetic information can be represented by two unary relations (or attributes) FA and MA that hold for those individuals that have inherited allele A from their father, respectively mother. Thus, for instance, $\mathtt{FA}(d_5)$ is false and $\mathtt{MA}(d_5)$ is true.

Table 1 now shows a relational Bayesian network that encodes a probabilistic model for the relations $R = \{\mathtt{FA}, \mathtt{MA}\}$ given the relations $S = \{\mathit{father}, \mathit{mother}\}$.

According to the formula $F_{\mathtt{FA}}$ the probability that d inherits A from his/her father is determined as follows: first it is determined by the subformula $F_{\mathit{father\text{-}in\text{-}pedigree}}$ whether the father of d is in the pedigree (using the convention that *noisy-or* evaluates to 0 when applied to an empty multiset, one sees that $F_{\mathit{father\text{-}in\text{-}pedigree}}(d)$ evaluates to 0 if $\mathit{father}(u,d)$ does not hold for any u, and to 1 otherwise). If d's father is in the pedigree, then the probability of $\mathtt{FA}(d)$ is determined using the formula $F_{A-from-father}(v)$, which evaluates to $\mathit{mean}\{\!|1,1|\!\} = 1$ if both $\mathtt{FA}$ and $\mathtt{MA}$ are true for d's father, to $\mathit{mean}\{\!|1,0|\!\} = 1/2$ if only one of $\mathtt{FA}$ and $\mathtt{MA}$ is true, and to $\mathit{mean}\{\!|0,0|\!\} = 0$ if neither is true. If d's father is not in the pedigree, then $\mathtt{FA}(d)$ is assigned a base rate probability 1/3. In exactly the same way the probability for $\mathtt{MA}(d)$ is determined.

Note that even though table 1 shows six probability formulas, it really represents a relational Bayesian networks composed of the two formulas $F_{\mathtt{FA}}$ and $F_{\mathtt{MA}}$. The other four formulas are only subformulas of these two formulas which are displayed separately for better readability, but that do not represent separate probabilistic relations *father-in-pedigree*, etc.

The formula $F_{\mathit{father\text{-}in\text{-}pedigree}}$ in the preceding example is an *indicator function* for the first-order formulas $\exists u \mathit{father}(u,v)$, i.e. for any S-structure $\mathcal{D}$ and $d \in D$:

$$F_{\mathit{father\text{-}in\text{-}pedigree}}(d)[\mathcal{D}] = 1 \quad \Leftrightarrow \quad \mathcal{D} \models \exists u \mathit{father}(u,d)$$

As shown in (Jaeger 1997), one can construct for any first-order formula $\phi(\boldsymbol{v})$ over the vocabulary $S \cup R$ a $S, R, \{\mathit{noisy\text{-}or}\}$-probability formula $F_\phi(\boldsymbol{v})$, such that for every $S \cup R$-structure $\mathcal{F}$ and $\boldsymbol{d} \subseteq D$: $F_\phi(\boldsymbol{d})[\mathcal{F}] \in \{0,1\}$, and

$$F_\phi(\boldsymbol{d})[\mathcal{F}] = 1 \quad \Leftrightarrow \quad \mathcal{F} \models \phi(\boldsymbol{d})$$

It is this fact that makes the full expressive power of first-order logic available for probabilistic modeling in relational Bayesian networks. In the general mapping $\phi \mapsto F_\phi$ (existential) quantifiers are translated into *noisy-or* combination functions. This is not very surprising, as (existential) quantification is basically a (deterministic) *or*, and *noisy-or* applied to multisets with 0,1-elements just reduces to *or*. In other words, we have found a close correspondence between first-order logical formulas, and $\{\mathit{noisy\text{-}or}\}$-probability formulas. This raises the question whether there are other natural correspondences between logics that use generalized quantifiers (e.g. second-order or Lindström quantifiers), or extend first-order logic in some other way (e.g. fixpoint logics), and probability formulas using other combination functions in addition to *noisy-or*. Unfortunately, it seems that other natural combination functions do not lead to correspondences to other logics: while it is possible to design special-purpose combination functions so that translations $\phi \mapsto F_\phi$ can also be obtained for ϕ in extended logics, it is not the case that natural combination functions like *mean* or *max* give rise to such translations.

We close this section with a second example that has a somewhat different flavor than example 2.3, and illustrates some different modeling techniques.

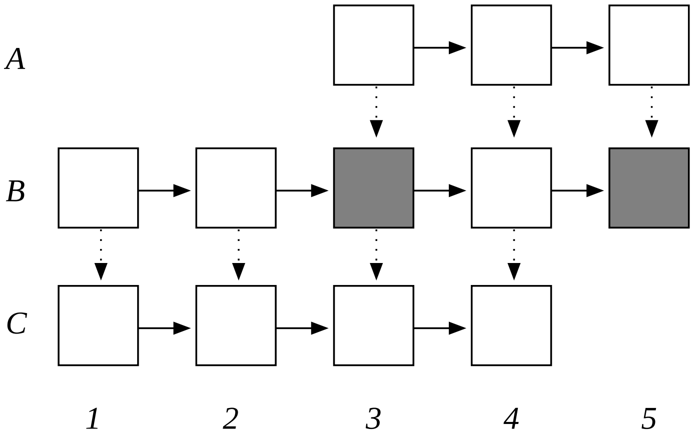

Figure 2: Robot environment

Example 2.4 Figure 2 shows a grid map that a robot may use to navigate in an office environment. The map distinguishes 12 possible locations, whose relative positions in a coordinate system are defined with binary relations *left-neighbor* (solid arrow), and *down-neighbor* (dotted arrows). In locations $B3$ and $B5$ are placed reading tables. The map, thus, can be seen as a relational structure for the vocabulary $S = \{\textit{left-neighbor}, \textit{down-neighbor}, \textit{table}\}$. To each reading table belongs one chair, which may be placed in any location directly adjoining the table, i.e. in one of $A3, B2, B4, C3$ for the chair belonging to the table at $B3$, and in $A5$ or $B4$ for the chair belonging to the table at $B5$. We now want to construct a probabilistic model for what locations are free, and what locations are blocked (by either a table or a chair) in this environment. More precisely, we want to represent a random relational structure model that takes a map in form of an S-structure, and returns a probability distribution over the interpretations of $R = \{\texttt{blocked}\}$. However, directly representing this as a S, R- random relational structure model with a S, R-relational Bayesian network will be impossible. Main reason for that is that there is a mutual dependency between e.g. $\texttt{blocked}(C3)$ and $\texttt{blocked}(A3)$, and we have no way to turn this symmetric dependency into an acyclic dependency relation $\succeq_{\Phi,\mathcal{D}}$ by a pure S, R - relational Bayesian network Φ.

We can avoid this problem by assuming that there is an additional order relation $<$ given in S, which defines an arbitrary total order on D. From a practical modeling point of view, this assumption is completely unproblematic, as we can always impose some order on D, and we can use this order to make dependency relations acyclic in such a way that the resulting distribution on R does not depend on the particular order chosen.

In addition to the auxiliary relation $<$ added to S, the relational Bayesian network in table 2 also adds an auxiliary binary relation `blocked-by-table` to R, where $\texttt{blocked} - \texttt{by} - \texttt{table}(u, v)$ represents the fact that u is the position of a table, and v is the location adjacent to u that is blocked by the chair belonging to u. Given the interpretation of `blocked-by-table` the interpretation of `blocked` is deterministically defined by the formula $F_{\texttt{blocked}}$, which is of the form F_ϕ, where ϕ is a first-order formula that says that v is a table, or blocked by a chair belonging to some table. The distribution of `blocked-by-table` is determined by the formula $F_{\texttt{blocked-by-table}}$. For better readability, we again have introduced some abbreviations: *is-table-neighbor* and *pred-selected* are abbreviations for first-order formulas; $F_{\textit{selected-from-remaining}}$ and $F_{\textit{selected-neighbor}}$ are abbreviations for probability formulas. To compute the probability that $\texttt{blocked}-\texttt{by}-\texttt{table}(u,v)$ holds, one first evaluates the subformula $F_{is-table-neighbor}(u,v)$, which returns 1 if u is a table location, and v adjacent to u, and 0 else. In the second case, the probability for $\texttt{blocked}-\texttt{by}-\texttt{table}(u,v)$ is 0. In the first case, this probability is computed with the subformula $F_{\textit{selected-neighbor}}(u,v)$. To evaluate $F_{\textit{selected-neighbor}}(u,v)$, one first computes $F_{\textit{pred-selected}}(u,v)$, which returns 1 iff there is another location w adjacent to u that precedes v in the order on D and for which $\texttt{blocked}-\texttt{by}-\texttt{table}(u,w)$ is true. If this is not the case, then $F_{\textit{selected-neighbor}}(u,v)$ evaluates to $1/k$, where k is the number of locations adjacent to u that do not precede v in the order on D. Intuitively, $F_{\textit{selected-neighbor}}(u,v)$ randomly chooses one neighbor of u by going through u's neighbors in ascending order, and selecting neighbor v with probability $1/k$ if no neighbor has already been selected. By this process, exactly one neighbor will be selected, each with equal probability.

$$
\begin{aligned}
&\textit{is-table-neighbor}(u,v) = \textit{table}(u) \wedge (\textit{left-neighbor}(u,v) \vee \textit{left-neighbor}(v,u) \vee \\
&\qquad\qquad \textit{down-neighbor}(u,v) \vee \textit{down-neighbor}(v,u))\\
&\textit{pred-selected}(u,v) = \exists w (w < v \wedge \texttt{blocked}-\texttt{by}-\texttt{table}(u,w))\\
&F_{\textit{selected-from-remaining}}(u,v) = \\
&\qquad \textit{mean}\{\!|\, v = w \mid w; (v < w \vee v = w) \wedge \textit{is-table-neighbor}(u,w) \,|\!\}\\
&F_{\textit{selected-neighbor}}(u,v) = F_{\textit{pred-selected}}(u,v)\cdot 0 + \\
&\qquad (1 - F_{\textit{pred-selected}}(u,v)) F_{\textit{selected-from-remaining}}(u,v)\\
&F_{\texttt{blocked-by-table}}(u,v) = F_{\textit{is-table-neighbor}}(u,v)\cdot F_{\textit{selected-neighbor}}(u,v) + \\
&\qquad (1 - F_{\textit{is-table-neighbor}}(u,v))\cdot 0\\
&F_{\texttt{blocked}}(v) = F_{table(v)\vee\exists u \texttt{blocked-by-table}(u,v)}(v)
\end{aligned}
$$

Table 2: Robot navigation example

3 Inference Problems

We now look at a number of inference problems for relational Bayesian networks. All of these are, in fact, inference problems for random relational structure models, i.e. they arise for whatever representation language one uses for these models. As our solution methods are based on relational Bayesian network representations, we here nevertheless formulate them directly in terms of relational Bayesian networks.

3.1 Elementary Inference

By elementary inference problems we mean inference problems that refer to one model instance $P_{\mathcal{D}}$ at a time, and therefore can be solved by elementary data structures and algorithms for handling such distributions, notably standard Bayesian networks and their inference algorithms. The most important inference problem of this kind is the *single-instance probabilistic inference problem*:

Input: A S, R- relational Bayesian network Φ
A S-structure $\mathcal{D}$
A query $P(\mathtt{r}_0(\boldsymbol{d}_0) = \alpha_0 \mid \mathtt{r}_1(\boldsymbol{d}_1) = \alpha_1, \ldots, \mathtt{r}_l(\boldsymbol{d}_l) = \alpha_l) = ?$
with $\mathtt{r}_i \in R$, $\boldsymbol{d}_i \subseteq D$, $\alpha_i \in \{true, false\}$.

Output: The probability value $P^{\Phi}_{\mathcal{D}}(\mathtt{r}_0(\boldsymbol{d}_0) = \alpha_0 \mid \mathtt{r}_1(\boldsymbol{d}_1) = \alpha_1, \ldots, \mathtt{r}_l(\boldsymbol{d}_l) = \alpha_l)$ if $\succeq_{\Phi,\mathcal{D}}$ is acyclic, and a message "$P^{\Phi}_{\mathcal{D}}$ undefined" otherwise.

This inference problem can be solved using the traditional approach of *knowledge based model construction*: one tries to construct a standard Bayesian network with one node for each ground atom $\mathtt{r}(\boldsymbol{d})$ constructible from the relations $\mathtt{r} \in R$ and elements $\boldsymbol{d} \subseteq D$. This construction will fail (because cycles are introduced among the nodes of the network) iff $\succeq_{\Phi,\mathcal{D}}$ is cyclic. Otherwise one obtains a Bayesian network representation of $P^{\Phi}_{\mathcal{D}}$. The query probability can then be computed using standard inference algorithms for Bayesian networks.

A straightforward implementation of such a construction would simply first determine for each ground atom $\mathtt{r}(\boldsymbol{d})$ the set $Pa(F_{\mathtt{r}}(\boldsymbol{d})[\mathcal{D}])$, and then create a conditional probability table for $\mathtt{r}(\boldsymbol{d})$ given $Pa(F_{\mathtt{r}}(\boldsymbol{d})[\mathcal{D}])$ by computing $F_{\mathtt{r}}(\boldsymbol{d})[\mathcal{D}, I(Pa(F_{\mathtt{r}}(\boldsymbol{d})[\mathcal{D}]))]$ for each instantiation I of $Pa(F_{\mathtt{r}}(\boldsymbol{d})[\mathcal{D}])$. This, however, will lead to Bayesian networks whose size grows exponentially in the size of the structure $\mathcal{D}$, because the size of $Pa(F_{\mathtt{r}}(\boldsymbol{d})[\mathcal{D}])$ can grow polynomially in the size of $\mathcal{D}$. Fortunately, one usually can do better by using a more sophisticated construction algorithm, in which auxiliary nodes are introduced that intuitively correspond to intermediate results in the recursive evaluation of the probability formulas. This optimized construction can be applied to relational Bayesian networks that only use multilinear combination functions:

Definition 3.1 A combination function is called *multilinear* if for all $n \geq 1$, and for all $i_1, \ldots, i_n \in \{0, 1\}$ there exists $\alpha_{i_1,\ldots,i_n} \in \mathbb{R}$, such that for all $p_1, \ldots, p_n \in [0, 1]$

$$comb\{\!|p_1, \ldots, p_n|\!\} = \sum_{(i_1,\ldots,i_n)\in\{0,1\}^n} \alpha_{i_1,\ldots,i_n} p_1^{i_1} \cdots p_n^{i_n}.$$

Theorem 3.2 Let Φ be a S, R, Γ-relational Bayesian network with Γ only containing multilinear combination functions. For every $\mathcal{D} \in \mathrm{Mod}_{fin}(S)$ for which

$P^{\Phi}_{\mathcal{D}}$ is defined there exists a standard Bayesian network $N^{\Phi}_{\mathcal{D}}$ representing $P^{\Phi}_{\mathcal{D}}$ whose size is polynomial in the size of $\mathcal{D}$.

This theorem is not constructive, and does not give rise directly to a construction algorithm for the Bayesian network. Indeed, such a general construction algorithm does not exist: consider the combination function *halts* defined by $\mathit{halts}\{p_i \mid i \in I\} = 1$ iff $|I|$ is the Gödel number of a Turing machine that halts, and $\mathit{halts}\{p_i \mid i \in I\} = 0$ otherwise. This is a multilinear combination function, but for relational Bayesian networks Φ that use this function the mapping $\mathcal{D} \mapsto N^{\Phi}_{\mathcal{D}}$ will not be computable. Constructive versions of theorem3.2 therefore have to be obtained for suitable subsets of multilinear combination functions. In (Jaeger 2001) an effective construction method is developed for the combination functions *noisy-or* and *mean*.

Figure 3 shows the network constructed for the relational Bayesian network of table 2 and the input structure of figure 2 augmented by an order $<$ on the locations. The shaded nodes labeled with single locations XK in this figure are the nodes for the ground atoms `blocked`(XK); the unfilled nodes labeled with pairs of locations (XK, YL) are the nodes for the ground atoms `blocked-by-table`(XK, YL), and the small unlabeled nodes are auxiliary nodes added in the construction (here they all are deterministic *or* nodes). The network shown was generated for an order $<$ with $A3 < B4 < C3 < B2$ and $B4 < A5$. Other orders would generate slightly different but structurally very similar networks. The network shown in figure 3 is somewhat simplified from that originally produced by the algorithm: the original network also contained nodes for all the other ground atoms `blocked`(XK) and `blocked-by-table`(XK, YL) not shown in the figure. These, however, all are isolated nodes with probability zero of being true. The original network also contained further auxiliary nodes that are not shown here (and that do not significantly change the basic structure of the network).

It is not immediately obvious that solving the elementary inference problem via the construction of a standard Bayesian network is a good approach. One might expect that based on the high-level representation language of relational Bayesian networks one can also develop high-level inference techniques, which directly operate on probability formulas, and do not first compile the low-level Bayesian network model. It turns out, however, that with such more sophisticated algorithms we cannot hope to improve the worst-case time complexity of inference via standard Bayesian network construction.

Theorem 3.3 If ETIME$\neq$ NETIME then there exists a $\emptyset, R, \{\mathit{noisy\text{-}or}\}$-relational Bayesian network Φ, such that elementary inference for Φ is not polynomial in the size of $\mathcal{D}$.

3.2 Non-elementary Inference

By a non-elementary inference problem we mean any inference problem that refers to a global property of a random relational structure model, not only one of its instances.

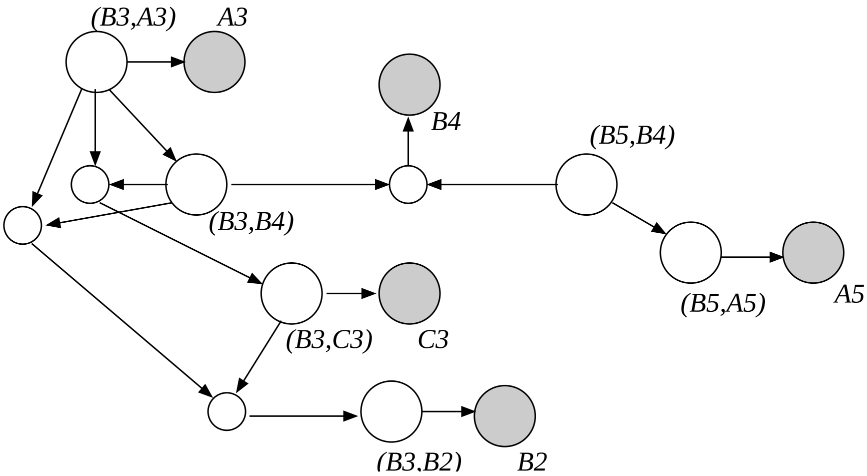

Figure 3: Standard Bayesian network constructed for example 2.4

3.2.1 Global semantics

One basic question one may have about a relational Bayesian network Φ is: is $P^{\Phi}_{\mathcal{D}}$ defined for all intended input structures $\mathcal{D}$? To illustrate this question, consider example 2.3. The relational Bayesian network in table 1 is meant to be applied to input structures $\mathcal{D}$ that encode pedigrees, i.e. S-structures in which the *father* and *mother* relations are acyclic, every element in the domain has at most one *father*- and one *mother*-predecessor, and perhaps some further restrictions are satisfied. It is easily verified in this case that Φ as given in table 1 does define an acyclic relation $\succeq_{\Phi,\mathcal{D}}$ for all such $\mathcal{D}$, and in fact for a much larger class of structures $\mathcal{D}$. In general, we are faced with the following *global semantics inference problem*:

Input: A S, R-relational Bayesian network Φ.
A class $\mathbf{D} \subseteq \mathrm{Mod}_{\mathit{fin}}(S)$ of S-structures

Output: "Yes" if $P^{\Phi}_{\mathcal{D}}$ is defined for all $\mathcal{D} \in \mathbf{D}$,
"no" otherwise.

This description of our inference problem is not quite complete, as we also need to say how to encode the class $\mathbf{D}$ of input structures. The canonical way to do this is to represent $\mathbf{D}$ by some logical sentence $\phi(\mathbf{D})$ so that $\mathbf{D} = \{\mathcal{D} \in \mathrm{Mod}_{\mathit{fin}}(S) \mid \mathcal{D} \models \phi(\mathbf{D})\}$. It will not be sufficient to use first-order sentences ϕ for this purpose, as first-order logic is not expressive enough to encode acyclicity conditions that will usually be part of the definition of $\mathbf{D}$ (as e.g. the acyclicity of the *father* and *mother* relations in example 2.3, and the acyclicity of the *left-neighbor* and *down-neighbor* relations in example 2.4. One of the weakest logics that will allow us to encode the required acyclicity conditions is *transitive* *closure logic.* This is an extension of first-order logic that allows to represent statements of the form "(a,b) is in the transitive closure of the relation defined by the formula $\phi(u,v)$" (see e.g. (Ebbinghaus & Flum 1999)). For example 2.3 we can then say, for instance, "there does not exist u such that (u,u) is in the transitive closure of $\mathit{father}(v,w)$". Given Φ, the acyclicity of $\succeq_{\Phi,\mathcal{D}}$, too, can be expressed by a sentence in transitive closure logic, i.e. there is a sentence $\psi(\Phi)$, such that for all $\mathcal{D} \in \mathrm{Mod}_{\mathit{fin}}(S)$:

$$\succeq_{\Phi,\mathcal{D}} \text{ is acyclic} \quad \Leftrightarrow \quad \mathcal{D} \models \psi(\Phi).$$

The solution of our inference problem now can be seen to be equivalent to checking whether the sentence

$$\phi(\mathbf{D}) \to \psi(\Phi) \tag{3}$$

is valid in all finite S-structures. It thus becomes clear that this problem cannot be decidable in general, because even for pure first-order sentences ϕ it is not decidable whether ϕ is satisfiable by a finite model (Trahtenbrot's Theorem). From this the undecidability of our problem follows, because for the Φ consisting of the single probability formula $F_{\mathtt{r}}(v) = \mathtt{r}(v)$ we get that (3) is valid iff $\phi(\mathbf{D})$ is not satisfiable.

We can thus only hope to solve the global semantics inference problem for certain restricted problem classes. One subclass for which one may conjecture the inference problem to be solvable is given by the case where S only contains unary relation symbols. Results on the decidability of first-order and monadic second-order logic for vocabularies of unary relation symbols indicate that transitive closure logic, too, is decidable in this case.

3.2.2 Limit Probabilities

Suppose we know that $P^{\Phi}_{\mathcal{D}}$ is defined for every $\mathcal{D} \in \mathbf{D}$, and let $\mathcal{D}_1, \mathcal{D}_2, \ldots$, with $D_1 \subseteq D_2 \subseteq \ldots$ be an infinite sequence of structures from $\mathbf{D}$. One should think of the $\mathcal{D}_i$ as a sequence of "similar" structures of increasing size. If $\boldsymbol{d} \subseteq D_1$, then we can evaluate the query $P(\mathtt{r}(\boldsymbol{d})) =?$ for every input structure $\mathcal{D}_i$, and consider the limiting behavior of

$$P^{\Phi}_{\mathcal{D}_i}(\mathtt{r}(\boldsymbol{d}))$$

as $i \to \infty$. This limiting behavior can be of interest for a number of reasons. Existence of the limit can be interpreted as a robustness property of the model Φ; the concrete value of the limit (if it exists) can be used as an approximation of the true query probability $P^{\Phi}_{\mathcal{D}}(\mathtt{r}(\boldsymbol{d}))$ if we are unable to specify the input structure $\mathcal{D}$ exactly, and only know that it is some element in the sequence $\mathcal{D}_i$.

To illustrate these issues, reconsider example 2.3. Given some pedigree $\mathcal{D}_1$ and $d \in D$ we can compute the probability $P^{\Phi}_{\mathcal{D}_1}(\mathtt{FA}(d))$. Next we may add some additional ancestors or descendants of d to the pedigree, obtaining an extended pedigree $\mathcal{D}_2$. In this pedigree we can again compute $P^{\Phi}_{\mathcal{D}_2}(\mathtt{FA}(d))$, which we would regard as a better approximation to the true probability of $\mathtt{FA}(d)$ than the first value. Continuing in this manner, we obtain the sequence $P^{\Phi}_{\mathcal{D}_i}(\mathtt{FA}(d))$ of probability values. We would now expect from our model Φ that this sequence converges to a limiting value (otherwise it would be impossible to justify the values computed for any specific input $\mathcal{D}$ as an approximation to the "true" probabilities). Moreover, we would like to compute this limit. In this particular example it is easy to see that here $P^{\Phi}_{\mathcal{D}}(\mathtt{FA}(d)) = 1/3$ for all input pedigrees $\mathcal{D}$, so that the desired convergence/robustness properties trivially hold.

The general *limit-probability inference problem* now can be formulated as follows:

Input: A S, R relational Bayesian network Φ
A sequence $\mathcal{D}_1, \mathcal{D}_2, \ldots,$ of S-structures with $D_1 \subseteq D_2 \subseteq \ldots$
A query $P(\mathtt{r_1}(\boldsymbol{d}_1) = \alpha_1, \ldots, \mathtt{r_k}(\boldsymbol{d}_1) = \alpha_k) =?$ with $\boldsymbol{d}_j \subseteq D_1$

Output: "undefined" if $P^{\Phi}_{\mathcal{D}_i}$ is undefined for some i
"no limit" if all $P^{\Phi}_{\mathcal{D}_i}$ are defined, but $P^{\Phi}_{\mathcal{D}_i}(\mathtt{r_1}(\boldsymbol{d}_1) = \alpha_1, \ldots, \mathtt{r_k}(\boldsymbol{d}_1) = \alpha_k)$ does not converge
$lim_i P^{\Phi}_{\mathcal{D}_i}(\mathtt{r_1}(\boldsymbol{d}_1) = \alpha_1, \ldots, \mathtt{r_k}(\boldsymbol{d}_1) = \alpha_k)$ otherwise.

Some comments are required: first, we here have integrated the question of whether all $P^{\Phi}_{\mathcal{D}_i}$ are defined into the formulation of the limit probability inference problem, even though one will usually only attempt to solve the limit probability inference problem for sequences $\mathcal{D}_i$ for which all $P^{\Phi}_{\mathcal{D}_i}$ are known to exist. Second, we only allow queries for unconditional probabilities. The reason for this is that limits of conditional probabilities $P_i(A \mid B)$ very often do not exist, even when the unconditional limits $P_i(A \wedge B)$ and $P_i(B)$ exist (Fagin 1976, Grove, Halpern & Koller 1992). However, this is only possible when $lim_i P_i(B) = 0$. Our approach to deal with limits of conditional probabilities, therefore, is to consider them only when the limit probability of the conditioning event is nonzero, in which case it is given by the fraction $lim_i P_i(A \wedge B)/lim_i P_i(B)$ of unconditional limits. We shall here not go into the question of how, in general, to encode a sequence $\mathcal{D}_i$ of input structures, because we will presently restrict attention to the special case $S = \emptyset$, in which there is only one canonical input sequence, with $\mathcal{D}_i$ being the structure containing i elements. The following example shows that for nonempty S one very easily constructs examples with non-converging probabilities.

Example 3.4 Let $S = \{s\}$ with binary s, R contain the three unary relations `blue`, `green`, and `even`, and let Φ be as given in table 3. This is a completely logical relational Bayesian network, i.e. all formulas are of the form F_{ϕ} for first-order formulas ϕ. For better readability they therefore here are directly written as ϕ, rather than in the form F_{ϕ}. Now let $\mathcal{D}_i$ be the S-structure with i elements in which s is interpreted as a successor relation. Then, according to $F_{\mathtt{blue}}$, $d \in D_i$ will be `blue` if it is the first element in the order defined by s, or if it has a `green` predecessor. Similarly, elements are `green` if they have a blue predecessor. Thus, the two formulas $F_{\texttt{blue}}, F_{\texttt{even}}$ describe a deterministic alternating coloring of D_i, starting with `blue`. For any $d \in D_i$ the formula $F_{\texttt{even}}(d)$ evaluates to 1 iff the last element in the s-order of D_i is `green`, i.e. if i is even. Thus, for any d: $P^{\Phi}_{\mathcal{D}_i}(\texttt{even}(d))$ alternates between 1 for even and 0 for odd i.

$$
\begin{aligned}
F_{\texttt{blue}}(v) &= \neg\exists w(s(w,v)) \vee \exists w(s(w,v) \wedge \texttt{green}(v)) \\
F_{\texttt{green}}(v) &= \exists w(s(w,v) \wedge \texttt{blue}(v)) \\
F_{\texttt{even}}(v) &= \forall w(\neg\exists u(s(w,u)) \rightarrow \texttt{green}(w))
\end{aligned}
$$

Table 3: A model with non-converging probabilities

We now simplify our problem-setting in two ways: first we assume $S = \emptyset$. Up to renaming of the elements, there then exists only one possible input structure $\mathcal{D}_i = \{d_1, \ldots, d_i\}$ of size i, and we can write P^{Φ}_i for $P^{\Phi}_{\mathcal{D}_i}$. Second, we impose a restriction on Φ, which ensures that all P^{Φ}_i are defined: call Φ *R-acyclic* if there exists an order on R such that the probability formula $F_{\texttt{r}}$ only contains indicator functions for relation symbols preceding `r` in that order. If Φ is R-acyclic, then clearly $P^{\Phi}_{\mathcal{D}}$ is defined for every $\mathcal{D}$ (this also holds when S is not empty). The restriction to R-acyclic Φ is not very material when the restriction to empty S has already been made, because the ability to condition the probability of one `r`-atom on other `r`-atoms only becomes a powerful modeling tool when there are suitable S-relations with which we can define these dependencies.

The restrictions $S = \emptyset$ and Φ being R-acyclic were actually part of the original definition of a relational Bayesian network given in (Jaeger 1997); the general framework there being labeled "recursive relational Bayesian network". For this restricted class of relational Bayesian networks, we can now obtain a partial solution to the limit probability inference problem. In the following theorem we refer to *exponentially convergent* combination functions. The exact definition of this property is rather technical and can be found in (Jaeger 1998*a*). Here we only mention that *noisy-or* is exponentially convergent, but *mean* is not.

Theorem 3.5 Let Φ be a $\emptyset, R, \Gamma$- relational Bayesian network that is R-acyclic, and where Γ only contains exponentially convergent combination functions. Then $lim_{i\rightarrow\infty} P^{\Phi}_i(\texttt{r}_1(\boldsymbol{d}_1) = \alpha_1, \ldots, \texttt{r}_k(\boldsymbol{d}_k) = \alpha_k)$ exists and is computable.

While from a practical point of view not wholly satisfactory due to the restriction to empty S, this theorem is already quite interesting from a theoretical point of view, as it substantially strengthens some previous convergence laws in finite model theory, especially Fagin's (1976) original 0-1 law, and a result by Oberschelp (1982) on the convergence of certain conditional probabilities. A further substantial strengthening of this result would be obtained if it could be extended to include the *mean* combination function.

3.2.3 Maximum Likelihood Input Structure

So far we have always assumed that the input S-structure $\mathcal{D}$ is given, and we want to make inferences about probabilities in randomly created R-structures. However, one can also consider a converse problem: given a model Φ, and an observed R-structure $\mathcal{E} \in \mathrm{Mod}_D(R)$, what is the most likely underlying S-structure $\mathcal{D}$, i.e. for what $\mathcal{D}$ is $P^{\Phi}_{\mathcal{D}}(\mathcal{E})$ maximal? In example 2.3, for instance, we may be given the genetic model of table 1 and genetic information on a number of individuals, and want to reconstruct the most likely pedigree for these individuals. This is almost the problem of the reconstruction of phylogenetic trees, only that here we have the simpler scenario that all nodes in the target tree are given, whereas in the case of phylogenetic trees only the leaves are taken to be observed, and suitable interior nodes have to be hypothesized (this latter scenario can also be realized in our framework by encoding a tree structure over a set of leaves directly by a suitable relation on the leaves). Instead of observing only one R-structure $\mathcal{E}$, we may also have observed a sample $\mathcal{E}_1, \ldots, \mathcal{E}_N \subseteq \mathrm{Mod}_D(R)$ of (independent) realizations of $P^{\Phi}_{\mathcal{D}}$. In example 2.4, for instance, we could have observed the `blocked-by-table` relation at several points in time, which would allow us to partly reconstruct the underlying map.

This leads us to the following *maximum likelihood input structure inference problem*:

Input: A S, R-relational Bayesian network Φ
A sample $\mathcal{E}_1, \ldots, \mathcal{E}_N \subseteq \mathrm{Mod}_D(R)$

Output: An S-structure $\mathcal{D} \in \mathrm{Mod}_D(S)$ that maximizes $\prod_{i=1}^{N} P^{\Phi}_{\mathcal{D}}(\mathcal{E}_i)$.

Formally, this is a maximum-likelihood statistical inference problem for the unknown parameter $\mathcal{D}$. The problem is trivially solvable in time exponential in the size of D by enumerating all S-structures $\mathcal{D}$ and computing $\prod_{i=1}^{N} P^{\Phi}_{\mathcal{D}}(\mathcal{E}_i)$ (if $P^{\Phi}_{\mathcal{D}}$ is defined). While clearly infeasible, it is instructive to look at this approach from a particular perspective: any S-structure $\mathcal{D}$ over D is defined by the values of the ground atoms $s(\boldsymbol{d})$ ($s \in S, \boldsymbol{d} \in D^{|s|}$) seen as 0,1-valued indicator variables. Now let $s_1(\boldsymbol{d}_1), \ldots, s_K(\boldsymbol{d}_K)$ be an enumeration of all ground S-atoms over D, and let $\overline{s_i(\boldsymbol{d}_i)}$ be either $s_i(\boldsymbol{d}_i)$ or $(1 - s_i(\boldsymbol{d}_i))$. Then the product $\prod_{j=1}^{K} \overline{s_i(\boldsymbol{d}_i)} =: 1_{\mathcal{D}}$ is the indicator of exactly one S-structure $\mathcal{D}$, i.e. it is a function of the indicator variables $s_i(\boldsymbol{d}_i)$ that evaluates to 1 for the truth values of $s_i(\boldsymbol{d}_i)$ in $\mathcal{D}$, and to 0 else. We can now express the likelihood function for S-structures $\mathcal{D}$ given the data $\mathcal{E} \in \mathrm{Mod}_D(R)$ as

$$L(\mathcal{D} \mid \mathcal{E}) = \sum_{\mathcal{D}' \in \mathrm{Mod}_D(S)} P^{\Phi}_{\mathcal{D}'}(\mathcal{E}) 1_{\mathcal{D}'}(\mathcal{D}). \tag{4}$$

In this way, the likelihood function is represented as a polynomial in the indicator functions $s_i(\boldsymbol{d}_i)$, and maximizing the likelihood becomes the problem of maximizing a polynomial in 0,1-valued variables. On the basis of the polynomial (4) this is nothing but a complicated way to describe the naive approach of computing $P^{\Phi}_{\mathcal{D}}(\mathcal{E})$ for every $\mathcal{D}$.

These considerations, however, motivate a somewhat different approach: we can try to represent $L(\mathcal{D} \mid \mathcal{E})$ as a polynomial different from (4), such that, first, the polynomial is smaller, and second, the structure of the polynomial permits a more directed search in the optimization. The strategy we can employ, is to transform for a given structure $\mathcal{E}$ the definition of $P^{\Phi}_{\mathcal{D}}(\mathcal{E})$ by (2) directly into a polynomial in the $s_i(\boldsymbol{d}_i)$.

We illustrate this technique with the relational Bayesian network Φ shown in table 4. This network satisfies two important restrictions: it only uses *noisy-or* as a combination function, and it is R-acyclic. The latter restriction ensures that $P^{\Phi}_{\mathcal{D}}$ is defined for all $\mathcal{D}$, so that the optimization problem is unconstrained over $\mathrm{Mod}_D(S)$. The first restriction is vital for our transformation of $P^{\Phi}_{\mathcal{D}}(\mathcal{E})$ into a polynomial, which in its present form only works for combination functions that are *insensitive to zeros*, i.e. that do not change their value when zeros are added or removed from its multiset-argument. Noisy-or is insensitive to zeros, but mean is not.

$F_{\texttt{red}}(v) = 0.8$
$F_{\texttt{blue}}(v) = \textit{noisy-or}\{\!\{0.6 \cdot \texttt{red}(w) \mid w; s(v,w)\}\!\}$

Table 4: Maximum likelihood input structure example

In this example R contains the two unary relations `red` and `blue`, and S the one binary s. Suppose we have observed a R-structure $\mathcal{E}$ over the domain D. Let $d \in D$, and suppose the `blue`(d) is true in $\mathcal{D}$. Then (2) contains the factor

$$F_{\texttt{blue}}(d) = \textit{noisy-or}\{\!\{0.6 \cdot \texttt{red}(w) \mid w; s(d,w)\}\!\} \tag{5}$$

As *noisy-or* is insensitive to zeros, we can rewrite this as

$$F_{\texttt{blue}}(d) = \textit{noisy-or}\{\!\{0.6 \cdot \texttt{red}(w) \cdot s(d,w) \mid w; \}\!\}, \tag{6}$$

i.e. instead of evaluating the subformula $0.6 \cdot \texttt{red}(w)$ only for those w for which $s(d,w)$ holds, we evaluate $0.6 \cdot \texttt{red}(w) \cdot s(d,w)$ for all w. This changes the resulting multiset by adding one zero for each w for which $s(d,w)$ does not hold. The given $\mathcal{E}$ instantiates all the indicators `red`(w), so that by substituting their truth values and expanding the *noisy-or*, (6) becomes

$$F_{\texttt{blue}}(d) = 1 - \prod_{d':\mathcal{D}\models\texttt{red}(d')} (1 - 0.6 \cdot s(d,d')) \tag{7}$$

All the other factors $F_{\texttt{blue}}(d')$, respectively $1 - F_{\texttt{blue}}(d')$ in (2) are obtained in the same way. Taking the product of all these factors (plus the $F_{\texttt{red}}(d)$ and $1 - F_{\texttt{red}}(d)$-factors, but these only add a constant) gives us a polynomial representation of $L(\mathcal{D} \mid \mathcal{E})$. This representation now only has polynomial size in $|D|$. Moreover, it is easy to optimize this polynomial: each indicator $s(d,d')$ appears at most once in the product, and the factor that contains $s(d,d')$ is maximized by setting $s(d,d')$ to 1 if it appears in a factor of the form $F_{\texttt{blue}}(d)$, and to 0 if it appears in a factor of the form $1 - F_{\texttt{blue}}(d)$. One thus immediately obtains that any structure $\mathcal{D}$ optimizes the likelihood in which $s(d, d')$ is true when $\texttt{blue}(d)$ and $\texttt{red}(d')$ are true in $\mathcal{E}$, and $s(d, d')$ is false when $\texttt{blue}(d)$ is false and $\texttt{red}(d')$ is true in $\mathcal{E}$.

In general (but under the restriction to combination functions insensitive to zeros) one will always obtain a polynomial representation of $L(\mathcal{D} \mid \mathcal{E})$ that has polynomial size. Of course, it will not always be possible to optimize this polynomial efficiently (using similar arguments as for theorem 3.3 one can show that the maximum likelihood input structure problem is not polynomial), but if the likelihood function $L(\mathcal{D} \mid \mathcal{E})$ has certain regularity properties that facilitate its optimization, they can be expected to be reflected in the structure of polynomial obtained from $P^{\Phi}_{\mathcal{D}}(\mathcal{E})$.

4 Learning

The learning problem for relational Bayesian networks in its most general form is the following:

Input: A set Γ of admissible combination functions
A sample $\mathcal{F}_1, \ldots, \mathcal{F}_N$ of S, R-structures

Output: A S, R, Γ-relational Bayesian network Φ that maximizes
a score function $\sigma(\Phi, \mathcal{E}_1, \ldots, \mathcal{E}_N)$.

The data elements $\mathcal{F}_i$ can also be written as pairs $(\mathcal{D}_i, \mathcal{E}_i)$ of S-structures $\mathcal{D}_i$ and R-structures $\mathcal{E}_i$ over a common domain D_i. Then we can define the likelihood of Φ given the data as

$$L(\Phi \mid \mathcal{F}_1, \ldots, \mathcal{F}_N) := \prod_{i=1}^{N} P^{\Phi}_{\mathcal{D}_i}(\mathcal{E}_i). \tag{8}$$

The score function σ will usually be composed of the likelihood function and some penalty term for the model complexity of Φ, or a Bayesian prior probability of Φ.

The optimization has to be constrained to relational Bayesian networks with combination functions from a set Γ that posses a parametric representation, so that one can effectively search over the elements of Γ. In fact, we will typically take Γ to be a small finite set, e.g. $\Gamma = \{\textit{noisy-or}, \textit{mean}\}$. This not only reduces the complexity of the search space, it also ensures that the learned Φ encodes the probabilistic model by the structure of its probability formulas, and not by some very specialized combination functions that are custom-built for the specific data set.

Implicit in the problem formulation here given is that $P^{\Phi}_{\mathcal{D}_i}$ must be defined for every $\mathcal{D}_i$. One could also generalize the problem setting by adding as an additional input a class $\mathbf{D} \subseteq \mathrm{Mod}_{\mathit{fin}}(S)$ of S-structures, and demand that for the learned Φ all $P^{\Phi}_{\mathcal{D}}$ with $\mathcal{D} \in \mathbf{D}$ are defined. In light of the discussion of section 3.2.1, however, we see that this will be very difficult for general classes $\mathbf{D}$, as the search space of admissible Φ will not always be decidable.

The nature of the learning problem described here differs from the usual statistical setting where the data consists of a random sample from a target distribution P that is to be learned. The random relational structure model Φ is not a single distribution P, but a family of distributions $\{P_{\mathcal{D}} \mid \mathcal{D} \in \mathbf{D}\}$, and the data consists of samples from these different distributions. For that reason the function (8), strictly speaking, is not a likelihood function in the usual sense. However, we can imagine the input structures $\mathcal{D}_i$ also be drawn from some distribution P on $\mathrm{Mod}_{\mathit{fin}}(S)$, in which case (8) becomes a proper likelihood up to the factor $\prod P(\mathcal{D}_i)$ that does not depend on Φ, and therefore can be neglected.

Clearly, certain models can only be learned if the data contains a sufficiently "rich" selection of different S-structures $\mathcal{D}$. On the other hand, for other models it makes no difference whether the data consists of a large sample $\mathcal{F}_1, \ldots, \mathcal{F}_N$ with many different underlying S-structures $\mathcal{D}_i$, or a small sample of a few large structures $\mathcal{F}_i$. In the extreme case, the data consists of a single structure $\mathcal{F}$. This is a particularly interesting case in practice, as it corresponds to learning a model from a single relational database (Friedman et al. 1999). To illustrate these issues, consider the two probability formulas

$$\begin{aligned} F^1_{\mathtt{edge}}(v,w) &= 1/1000 & (9)\\ F^2_{\mathtt{edge}}(v,w) &= \mathit{mean}\{\!|v = u \mid u;|\!\} & (10)\end{aligned}$$

each of which defines a random relational structure model for $S = \emptyset$ and $R = \{\mathtt{edge}\}$. $F^2_{\mathtt{edge}}$ encodes a sparse random graph model, where any two nodes in a graph with n nodes are connected with probability $1/n$. Now suppose the data consists of a single R-structure $\mathcal{E}$ with 1000 nodes that contains a total of 1000 edges. Then $F^1_{\mathtt{edge}}$ and $F^2_{\mathtt{edge}}$ obtain the same likelihood score given the data. However, a penalty for model complexity included in the score function will lead to a preference for model $F^1_{\mathtt{edge}}$ over $F^2_{\mathtt{edge}}$. If, on the other hand, the data consists of a number of graphs $\mathcal{D}_i$ of different sizes n_i, and each containing approximately n_i edges, then $F^2_{\mathtt{edge}}$ obtains a much higher likelihood score than $F^1_{\mathtt{edge}}$ and can be learned.

Given a score function σ and assuming a finite set Γ, our learning problem has a structure that is familiar from the learning problem for Bayesian networks and other graphical models, and more specifically the learning problems considered e.g. in (Friedman et al. 1999, Muggleton 2000, Sato & Kameya 2001): it is an optimization over a discrete search space of model structures (here the probability formula structures), and for each structure an optimization over a continuous space of model parameters (here the values of the constants). The following definition makes the concept of a probability formula structure precise.

Definition 4.1 Let S, R, Γ be as in definition 2.2. Let $\theta_1, \theta_2, \ldots$ be a set of *parameter variables*. The set of S, R, Γ-probability formula structures is defined inductively by the syntax rule

(i) (Parameters) Each θ_i is a probability formula structure,

and the rules (ii)-(iv) from definition 2.2.

Note that we here view the choice of a particular combination function in construction rule (iv) as part of the discrete structure. We will denote probability formula structures with F^*, and relational Bayesian network structures with Φ^*. Note that we may use the same parameter variable more than once in a base case of the construction of a probability formula structure. This enables us to include equality constraints between parameters in the discrete model structure. As an example, consider the relational Bayesian network in table 1. Each of the two formulas $F_{\texttt{FA}}$ and $F_{\texttt{MA}}$ contains two constants: the constant 1 inside the *noisy-or* that defines the subformulas $F_{\textit{father-in-pedigree}}$, respectively $F_{\textit{mother-in-pedigree}}$, and the constants 1/3. Legal relational Bayesian network structures can now be obtained both by substituting for these constants four different parameter variables $\theta_1, \ldots, \theta_4$, or by substituting the same variable θ_1 for the two occurrences of 1, and θ_2 for the two occurrences of 1/3. In the latter case we encode in the structure the prior knowledge that the models for $F_{\texttt{FA}}$ and $F_{\texttt{MA}}$ are the same.

In many learning problems, for a given structure and assuming complete data, the optimization over the continuous model parameters is easy and reduces to some frequency counts in the empirical distribution. This, unfortunately, is not the case for relational Bayesian networks. However, under the restriction to multilinear combination functions, it still is a fairly well-behaved optimization problem.

Theorem 4.2 Let Γ be a set of multilinear combination functions, $\mathcal{F}_1, \ldots, \mathcal{F}_N$ be a set of complete data items. Let Φ^* be a relational Bayesian network structure with parameter variables $\theta_1, \ldots, \theta_K$. The likelihood function $L(\theta_1, \ldots, \theta_K \mid \mathcal{F}_1, \ldots, \mathcal{F}_N)$ for the parameter values given the structure Φ^* then is a polynomial in the θ_j.

The proof is straightforward, and the result actually holds for the wider class of polynomial combination functions.

An incomplete data item is a structure $(\mathcal{D}, \hat{\mathcal{E}})$ where $\mathcal{D}$ is a fully specified S-structure, and $\hat{\mathcal{E}}$ is a partially specified R-structure over the same domain, i.e. $\hat{\mathcal{E}}$ defines the truth values of some ground atoms $\mathbf{r}(\boldsymbol{d})$, whereas the truth values of other ground atoms may be missing. The basic structure of the parameter learning problem for relational Bayesian networks is the same for incomplete as for complete data: under the restriction to multilinear combination functions it still is the problem of optimizing a polynomial. In practice, however, the incomplete data case can be substantially harder than the complete data case, as the polynomial will have exponential size in the number of missing truth values if constructed naively. Whether this exponential size can typically be avoided in practice by using more sophisticated constructions is a topic of ongoing work, as is the question whether a suitable variant of the EM-algorithm can be developed for our parameter learning problem.

5 Infinite Domains

So far we have presented relational Bayesian networks strictly as a representation language for random relational structure models in the sense of definition 1.1, i.e. restricted to finite domains. However, the language can also be used to define distributions on the classes of R-structures over infinite domains D, especially Herbrand universes arising from function and constant symbols. In (Jaeger 1998*b*) this has been investigated for the case where D is an unstructured countably infinite set (i.e $S = \emptyset$), and Φ is $R-$*acyclic*. For this case it has been shown that Φ defines a unique probability distribution over $\mathrm{Mod}_D(R)$, and that probabilistic queries of the form $P(\mathtt{r}_0(\boldsymbol{d}_0) = \alpha_0 \mid \mathtt{r}_1(\boldsymbol{d}_1) = \alpha_1, \ldots, \mathtt{r}_l(\boldsymbol{d}_l) = \alpha_l) =?$ can be solved.

When $S \neq \emptyset$ (and, in particular, now allowing that S also may contain function symbols that are interpreted in the canonical way over a Herbrand universe), then one can have the case that the dependency relation $\succeq_{\Phi,\mathcal{D}}$ is acyclic, but has infinite descending chains: if $S = \{f\}$ with a single unary function symbol f, for instance, and $D = \{a, f(a), f(f(a)), \ldots\}$, then Φ consisting of the single probability formula

$$F_{\mathtt{r}}(v) = \textit{noisy-or}\{\!\{0.2r(w) \mid w; w = f(v)\}\!\} \tag{11}$$

induces the infinite chain $\mathtt{r}(a) \succeq_{\Phi,\mathcal{D}} \mathtt{r}(f(a)) \succeq_{\Phi,\mathcal{D}} \ldots$. For infinite $\mathcal{D}$, a unique distribution is only guaranteed to be defined by Φ if $\succeq_{\Phi,\mathcal{D}}$ is both acyclic and well-founded. However, even in that case, elementary inference problems may be undecidable (Jaeger 1998*b*).

When $\succeq_{\Phi,\mathcal{D}}$ is not acyclic and well-founded (or simply not known to be acyclic and well-founded), then one can still interpret a probability formula as a constraint on probability distributions on $\mathrm{Mod}_D(R)$ that is satisfied by no, exactly one, or several distributions. One easily sees, for example, that the conditional probabilities defined by (11) can only be satisfied by a distribution $P_{\mathcal{D}}$ with $P_{\mathcal{D}}(\mathtt{r}(d)) = 0$ for all $d \in \{a, f(a), f(f(a)), \ldots\}$.

Independent from the properties of $\succeq_{\Phi,\mathcal{D}}$, one can use similar techniques as used by Pfeffer and Koller (2000) to make approximate inferences about probabilities entailed by the relational Bayesian network, i.e. to compute for a query $P(\mathtt{r}(\boldsymbol{d})) =?$ a sequence of intervals $I_1 \supseteq I_2 \supseteq I_3 \supseteq \ldots$ such that for all j: $P_{\mathcal{D}}(\mathtt{r}(\boldsymbol{d})) \in I_j$ for all distributions $P_{\mathcal{D}}$ that satisfy the constraints imposed by Φ.

6 Conclusion

We have given an overview of probabilistic modeling and inference with relational Bayesian networks. The language of relational Bayesian networks is defined by a rigorous syntax with only four construction rules that resemble the syntax elements of predicate logic. This elementary syntax and its close connection to logical formulas on the one hand, and (for multilinear combination functions) polynomials on the other hand, enables us to reduce many non-trivial inference problems to more standard inference problems in logic or optimization. While many of these inference problems are inherently intractable in general, their formulation as standard optimization or satisfiability problems allows us to directly utilize the vast amount of knowledge on solution heuristics and tractable sub-classes available for such problems.

References

Cussens, J. (1999), Loglinear models for first-order probabilistic reasoning, *in* 'Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–99)', Morgan Kaufmann Publishers, San Francisco, CA, pp. 126–133.

Dagum, P., Galper, A. & Horvitz, E. (1992), Dynamic network models for forecasting, *in* 'Proceedings of the Eighth Annual Conference on Uncertainty in Artificial Intelligence (UAI–92)', Morgan Kaufmann Publishers, San Francisco, CA, pp. 41–48.

Ebbinghaus, H.-D. & Flum, J. (1999), *Finite Model Theory*, Perspectives in Mathematical Logic, second edition edn, Springer Verlag.

Fagin, R. (1976), 'Probabilities on finite models', *Journal of Symbolic Logic* **41**(1), 50–58.

Friedman, N., Getoor, L., Koller, D. & Pfeffer, A. (1999), Learning probabilistic relational models, *in* 'Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)'.

Grove, A., Halpern, J. & Koller, D. (1992), Asymptotic conditional probabilities for first-order logic, *in* 'Proc. 24th ACM Symp. on Theory of Computing'.

Jaeger, M. (1997), Relational bayesian networks, *in* D. Geiger & P. P. Shenoy, eds, 'Proceedings of the 13th Conference of Uncertainty in Artificial Intelligence (UAI-13)', Morgan Kaufmann, Providence, USA, pp. 266–273.

Jaeger, M. (1998*a*), Convergence results for relational Bayesian networks, *in* V. Pratt, ed., 'Proceedings of the 13th Annual IEEE Symposium on Logic in Computer Science (LICS-98)', pp. 44–55.

Jaeger, M. (1998*b*), Reasoning about infinite random structures with relational bayesian networks, *in* A. G. Cohn, L. Schubert & S. C. Shapiro, eds, 'Proceedings of the 6th International Conference on Principles of Knowledge Representation and Reasoning (KR-98)', Morgan Kaufmann, Trento, Italy, pp. 570–581.

Jaeger, M. (2001), 'Complex probabilistic modeling with recursive relational Bayesian networks', *Annals of Mathematics and Artificial Intelligence* **32**, 179–220.

Kersting, K. & de Raedt, L. (2001), Towards combining inductive logic programming and bayesian networks, *in* 'Proceedings of the Eleventh International Conference on Inductive Logic Programming (ILP-2001)', Springer Lecture Notes in AI 2157.

Koller, D. (1999), Probabilistic relational models, *in* 'Proceedings of ILP-99', LNAI 1634, pp. 3–13.

Muggleton, S. (1996), Stochastic logic programs, *in* L. de Raedt, ed., 'Advances in Inductive Logic Programming', IOS Press, pp. 254–264.

Muggleton, S. (2000), 'Learning stochastic logic programs', *Electronic Transactions on Artificial Intelligence* **4, Section B**, 141–153.

Ngo, L. & Haddawy, P. (1997), 'Answering queries from context-sensitive probabilistic knowledge bases', *Theoretical Computer Science* **171**, 147–177.

Oberschelp, W. (1982), Asymptotic 0-1 laws in combinatorics, *in* D. Jungnickel, ed., 'Combinatorial Theory', Vol. 969 of *Lecture Notes in Mathematics*, Springer Verlag.

Pfeffer, A. & Koller, D. (2000), Semantics and inference for recursive probability models, *in* 'Proceedings of AAAI-2000'.

Poole, D. (1993), 'Probabilistic horn abduction and Bayesian networks', *Artificial Intelligence* **64**, 81–129.

Sato, T. (1995), A statistical learning method for logic programs with distribution semantics, *in* 'Proceedings of the 12th International Conference on Logic Programming (ICLP'95)', pp. 715–729.

Sato, T. & Kameya, Y. (2001), 'Parameter learning of logic programs for symbolic-statistical modeling', *Journal of Artificial Intelligence Research* **15**, 391–454.