

Modelling Relational Statistics With Bayes Nets

Oliver Schulte, Hassan Khosravi, Arthur E. Kirkpatrick, Tianxiang Gao, and
Yuke Zhu

School of Computing Science, Simon Fraser University,
Burnaby-Vancouver, Canada

Abstract. Class-level dependencies model general relational statistics over attributes of objects and links connecting them. Class-level relationships are important in themselves, and they support applications like policy making, strategic planning, and query optimization. An example of a class-level query is “what is the percentage of friendship pairs where both friends are women?”. To represent class-level statistics, we utilize Parametrized Bayes nets (PBNs), a first-order logic extension of Bayes nets. The standard grounding semantics for PBNs is appropriate for answering queries about specific ground facts but not appropriate for answering queries about classes of individuals. We propose a *grounding-free* interpretation of PBNs that supports class-level queries, based on Halpern’s classic random selection semantics for probabilistic first-order logic [1]. Learning the parameters for this semantics can be done using the recent pseudo-likelihood measure [2] as the objective function. The parameter settings that maximize this objective function are the empirical frequencies in the relational data. A naive computation of the empirical frequencies of the relations is intractable due to the complexity imposed by negated relations. We render the computation tractable by using the inverse Möbius transform. Evaluation on four benchmark datasets shows that maximum pseudo-likelihood provides accurate estimates at different sample sizes.

1 Introduction

Many applications store data in relational format, with different tables for entities and their links. Relational data introduces the machine learning problem of *class-level frequency estimation*: building a model that can answer generic statistical queries about classes of individuals in the database [3]. For example, a class-level query for a social network database may be “what is the percentage of friendship pairs where both are women”? A movie database example would be “what is the percentage of male users who have given a high rating to an action movie?” A model of database statistics can be used for several applications:

Statistical first-order Patterns. AI research into combining first-order logic and probability investigated in depth the representation of statistical patterns in relational structures [1, 4]. Often such patterns can be expressed as *generic statements*, like “intelligent students tend to take difficult courses”.

Policy making and strategic planning. A university administrator may wish to know which program characteristics attract high-ranking students in general, rather than predict the rank of a specific student in a specific program. Maier *et al.* [5] describe several applications of causal-relational knowledge for decision making. These causal relations reflect generic correlations in the database.

Query optimization. A statistical model predicts a probability for given table join conditions that can be used to infer the size or cardinality of a database query result [3]. Cardinality is used to minimize the size of intermediate join tables [15].



Semantics. We focus on building a Bayes net model for relational statistics, using the Parametrized Bayes nets (PBNs) of Poole [7]. The nodes in a PBN are constructed with functors and first-order variables (e.g., *gender(X)* may be a node). The original PBN semantics is a grounding semantics where the first-order Bayes net is instantiated with all possible groundings to obtain a directed graph, whose nodes are functors with constants (e.g., *gender(sam)*). The ground graph can be used to answer queries *about individuals*, such as “if user *sam* has 3 friends, female *rozita*, males *ali* and *victor*, what is the probability that *sam* is a woman”? However, as pointed out by Getoor [8], the ground graph is not appropriate for answering class-level queries because these are about generic rates and percentages, not about any particular individuals.

We propose a new interpretation for Parametrized Bayes nets that supports class-level queries. The semantics is based on Halpern’s classic random selection semantics for probabilistic first-order logic [1, 4]. While we focus on PBNs, the random selection semantics can be applied to any statistical-relational model whose syntax is based on first-order logic. Halpern’s semantics views statements with first-order variables as expressing statistical information about classes of individuals. For instance, the claim “the percentage of friendship pairs where both are women is 60%” could be expressed by the formula

$$P(\text{Gender}(X) = \text{female}, \text{Gender}(Y) = \text{female} | \text{Friend}(X, Y)) = 60\%.$$


Learning. A standard Bayes net parameter learning method is maximum likelihood estimation, but this method is difficult to apply for Bayes nets that represent relational data because the cyclic data dependencies in relations violate the requirements of a traditional likelihood measure. We circumvent the limitations of classical likelihood measures by using a relational pseudo-likelihood measure for Bayes nets [2] that is well defined even in the presence of cyclic dependencies. In addition to this robustness, the relational pseudo-likelihood matches the random selection semantics because it is also based on the concept of random instantiations. An estimator that chooses the parameters that maximize this pseudo-likelihood function (MPLE), has a closed-form solution: the MPLE parameters are the empirical frequencies, as with classical i.i.d. maximum likelihood estimation. Since MPLE depends only on the generic event frequencies in the

data, it can be viewed as an instance of *lifted learning*. Computing the empirical frequencies for negated relationships is difficult, however, because enumerating the complement of a relationship table is computationally infeasible. We show that the inverse Möbius transform [9] makes MPLE tractable, even in the case of negated relationships.

Results. We evaluate MPLE on four benchmark real-world datasets. On complete-population samples MPLE achieves near perfect accuracy in parameter estimates, and excellent performance on Bayes net queries. The accuracy of MPLE parameter values is high even on medium-size samples.

Contributions. Our main contributions for frequency modelling in relational data are the following:

1. A new class-level semantics for graphical first-order models, deriving from the random selection semantics for probabilistic first-order logic.
2. Making the computation of frequency estimates tractable by computing database statistics using the inverse Möbius transform. This transform is a general procedure for computing relational counts that involve negated links. It has application in Probabilistic Relational Models [10, Sec.5.8.4.2], multi-relational data mining, and inductive logic programming models with clauses that contain negated relationships.
3. We contribute to unifying instance-level/ground-level and class-level relational probabilities (defined in the next section) in two ways. (1) We show that the same first-order model can be used for both types of inference. (2) We show that the same objective function is suitable for learning models for both types of queries.

 *Paper Organization.* We begin with background in probability logic and relational Bayes net models. Section 3.2 presents the random selection semantics for Bayes nets. Section 5 describes the inverse Möbius transform for relational data. Simulation results are presented in Section 7, showing the runtime cost of estimating parameters, and evaluations of their quality by (a) comparison with the true population parameter values, and (b) inference on random queries.

2 Background: Probability Logic

Our work is a synthesis of concepts and techniques from logic, database theory, probability theory, and Bayes nets. We take a minimalist approach, reviewing just enough concepts to discuss the issues that are the focus of this paper, with an emphasis on semantics. Our results can be extended to richer frameworks.

2.1 Syntax

We adopt Chiang and Poole’s version of predicate logic for statistical-relational modelling [11], and extend it with concepts from Halpern’s probability logic [1].

Constants are written in lower case, for instance *bob* and *cs100*, and represent individuals. A **type** is associated with each individual, for instance *bob* is a *person*. The notation $\mathcal{P}(\tau)$ represents the **population** associated with type τ , which is a set of individuals. We assume that the populations are specified as part of the type description. In contrast, Bayesian Logic (BLOG) extends predicate logic to an open-world setting with uncertainty about the existence or identity of individuals [12]. A logical variable, or **population variable**, is written in upper-case (e.g., X or *Person*). A population variable is also typed, e.g. *Person* is associated with the type $\tau = \text{person}$. A **functor** represents a mapping $f : \Omega_f \rightarrow V_f$ where f is the name of the functor, the symbol $\Omega_f \equiv \mathcal{P}(\tau_1), \dots, \mathcal{P}(\tau_a)$ denotes the *domain of the relation*, and τ_1, \dots, τ_a lists the input or *argument types* of the functor. The output type or *range* of the functor is denoted by V_f . In this paper we consider only functors with a finite range of values, none of which appears in any of the populations. If $V_f = \{T, F\}$, the functor f is a (Boolean) **predicate**. A predicate with more than one argument is called a **relationship**; other functors are called **attributes**.

An **atom** is an expression of the form $f(\sigma_1, \dots, \sigma_a)$, where each σ_i is either a constant or a population variable [11]. The types of $\sigma_1, \dots, \sigma_a$ must match the argument type of f . If all the $\sigma_1, \dots, \sigma_a$ are constants, then $f(\sigma_1, \dots, \sigma_a)$ is a **ground atom**. A **literal** specifies the value of an atom, generically $f(\sigma_1, \dots, \sigma_a) = v$, where $v \in V_f$. A literal is the basic unit of information in this language. Literals can be combined to generate *formulas*. In this paper we consider only formulas that are **conjunctions** of literals. We use the Prolog-style comma notation $f(\sigma_1, \dots, \sigma_a) = v, \dots, f'(\sigma'_1, \dots, \sigma'_{a'}) = v'$ to denote the conjunction of two literals or more.

Let $\{X_1, \dots, X_k\}$ be a set of population variables with types τ_1, \dots, τ_k . A **grounding** γ is a set $\gamma = \{X_1 \setminus x_1, \dots, X_k \setminus x_k\}$ where each x_i is a constant of the same type as X_i . A grounding for an atom (formula) is a grounding for the variables that appear in the grounding (formula). We denote the result of applying a grounding γ to a formula ϕ by $\phi\gamma$.

In classic AI research into the connections between probability and logic, Halpern and Bacchus extended predicate logic to incorporate statements about probabilities [1, 4]. Formalizing probability within logic has several advantages, including: (1) A formal semantics for probability assertions. (2) Principles for probabilistic reasoning can be defined as logical axioms. A **probability formula** is a conjunction of the form

$$P(\phi_1) = p_1, \dots, P(\phi_n) = p_n$$

where each p_i is a term that denotes a real number in the interval $[0,1]$ and each ϕ_i is a conjunctive formula in predicate logic as defined above. We refer to an extension of predicate logic with probability formulas as a *probability logic*.

Examples. Consider a social network model with a single type of individual, $\tau = \text{people}$, associated with two population variables X and Y . The expres-

sion $\mathcal{P}(\text{people})$ denotes a set of individual people, for instance $\{\text{anna}, \text{bob}\}$. The functor

$$\text{gender} : \mathcal{P}(\text{people}) \rightarrow \{M, W\}$$

takes as input a person and returns either M for “man” or W for “woman”. The functor

$$\text{Friend} : \mathcal{P}(\text{people}) \times \mathcal{P}(\text{people}) \rightarrow \{T, F\}$$

is a predicate that takes as input a pair of people and returns a truth value indicating whether the two people are friends or not.

Nonground atoms in this language include $\text{gender}(X)$, $\text{gender}(Y)$, $\text{Friend}(X, Y)$. Ground atoms include $\text{gender}(\text{anna})$, $\text{Friend}(\text{anna}, \text{bob})$. A conjunction of nonground literals is $\text{gender}(X) = W$, $\text{gender}(Y) = M$, $\text{Friend}(X, Y) = T$. Applying the grounding $\{X \setminus \text{anna}, Y \setminus \text{bob}\}$ produces the conjunction of ground literals $\text{gender}(\text{anna}) = W$, $\text{gender}(\text{bob}) = M$, $\text{Friend}(\text{anna}, \text{bob}) = T$. An example of a ground probability formula is $P(\text{Flies}(\text{tweety}) = T) = 90\%$, and an example of a nonground probability formula is $P(\text{Flies}(B) = T) = 90\%$.

2.2 Semantics for Ground Formulas.

Logic semantics defines the rules for determining the truth of a sentence with respect to a particular relational structure [13, Ch.7.4.2]. A **relational structure** \mathcal{S}_f for functor f is a set of tuples $\{\langle x_1, \dots, x_a, v \rangle\}$, where the tuple $\langle x_1, \dots, x_a \rangle$ is in the input domain of f , and the value v is in the range of f . Each tuple in the input domain of f appears exactly once in \mathcal{S}_f . A relational structure \mathcal{S} for a functor language contains one relational structure for each functor. If the structure \mathcal{S} assigns the same value to a ground atom as a literal l , the literal is true in \mathcal{S} , written as $\mathcal{S} \models l$. Thus

$$\mathcal{S} \models f(x_1, \dots, x_a) = v \text{ iff } \langle x_1, \dots, x_a, v \rangle \in \mathcal{S}_f.$$

A conjunction is true if all its conjuncts are true. Databases systems employ a tabular representation of a relational structure, as illustrated in Figure 1.

Examples. All ground literals listed above are true in the structure of Figure 1. False literals include $\text{Coffee_Drinker}(\text{bob}) = T$ and $\text{Friend}(\text{anna}, \text{anna}) = T$.

With respect to a fixed relational structure, a ground literal has a definite truth value, with no uncertainty. Therefore a ground probability formula such as $P(\text{gender}(\text{bob}) = M) = 50\%$ expresses uncertainty about which relational structure is correct. This uncertainty can be represented by a distribution μ that assigns a probability to each relational structure [1]. The probability of a ground literal is then the probability of the set of structures in which the literal is true, and similarly for a formula:

$$\mu \models P(\phi) = p \text{ iff } p = \sum_{\mathcal{S} : \mathcal{S} \models \phi} \mu(\mathcal{S})$$

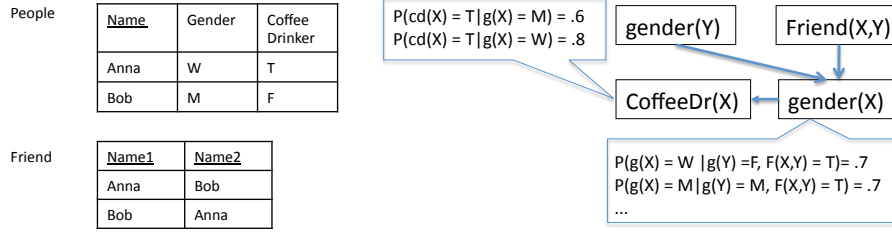


Fig. 1. Left: A relational structure represented in tabular form. By convention, a pair not listed in the *Friend* table is not a pair of friends. Right: An illustrative Bayes Net for this database, whose nodes are atoms. *Friend*(*X*, *Y*) is a relationship node, the other three nodes are attribute nodes. The conditional probabilities are not related to the database.

We refer to the probabilities defined by ground probability formulas as **instance-level probabilities** since they concern instances of types.



Semantics for Nonground Formulas.

Formulas Without Probabilities: Database Queries. We follow database theory and view a nonground formula as a *logical query* [14]. Intuitively, the formula specifies conditions on the population variables that appear in it, and the result set lists all individuals that satisfy these conditions in a given relational structure. Formally, we denote the formula **result set** for a structure as $\mathbb{R}_S(\phi)$, defined by

$$\mathbb{R}_S(\phi) \equiv \{\langle x_1, \dots, x_k \rangle : S \models \phi\{X_1 \setminus x_1, \dots, X_k \setminus x_k\}\}$$

If we extend our predicate logic with disjunctions, and restrict negations in a natural way, we obtain a logical query language known as the *safe domain relational calculus* [14]. A fundamental result of database theory states that the result sets definable in the safe domain relational calculus are exactly the same as those definable in standard relational algebra, using table joins.

The **cardinality** of a query formula in a structure is simply the size of its resultset: $\#_S(\phi) \equiv |\mathbb{R}_S(\phi)|$. The *cardinality estimation problem* is to estimate the size of a result set, without actually retrieving the tuples in the set. Cardinality estimation is key for optimizing query processing, and has therefore been studied extensively by database researchers [15]. Getoor, Taskar, and Koller observed that the query cardinality estimation problem is equivalent to the *query probability estimation problem*, defined by dividing the cardinality of a query's resultset by its maximum size:

$$P_S(\phi) \equiv \frac{\#_S(\phi)}{\mathcal{P}(\tau_1) \times \dots \times \mathcal{P}(\tau_k)} \quad (1)$$

check DRC definition

where τ_1, \dots, τ_k are the types of the k population variables that appear in the query formula. For example, in the structure of Figure 1, we have

$$P_S(\text{Friend}(X, Y) = T, \text{Gender}(X) = M, \text{Gender}(Y) = F) = 1/4$$

since the gender conditions are met in 1 out of 4 possible instantiations of the X, Y population variables. The advantage of the probability formulation is that it casts cardinality estimation as a statistical density estimation problem. Getoor [8] developed a Bayes net-type model for query probability estimation.

Formulas With Probabilities: Random Selections. Halpern introduced the **random selection semantics** for formulas containing population variables [1]. A relational structure is augmented with a distribution P_τ over each type population. We may then view a population variable as *a random selection* from its population. These random selections are independent, so for a fixed set of population variables, the population distributions induce a distribution over the groundings of the variables. We assume that all individuals are treated equally, and hence we assume a uniform distribution over each population.¹ This leads to the following definitions.

1. $P_S(X = x) \equiv P_\tau(x) = \frac{1}{|\mathcal{P}(\tau)|}$.
2. $P_S(X_1 = x_1, \dots, X_k = x_k) \equiv P_{\tau_1}(x_1) \times \dots \times P_{\tau_k}(x_k) = \frac{1}{|\mathcal{P}(\tau_1)| \times \dots \times |\mathcal{P}(\tau_k)|}$

where τ_i is the type of variable X_i and constant x_i . The random selection semantics states that the probability of a nonground formula is the probability of the set of groundings that satisfy the formula.

$$\mathcal{S} \models P(\phi) = p \text{ iff } p = \sum_{\langle x_1, \dots, x_k \rangle \in \mathbb{R}_S(\phi)} P_S(x_1, \dots, x_k) \quad (2)$$

With uniform population distributions, the random selection semantics assigns to each formula its query probability $P_S(\phi)$ (Equation 1).

Interpretation. As the random selection semantics is a key concept in this paper, we discuss several interpretations.

Atoms as Random Variables. Population variables are random variables selecting population members. Nonground atoms represent functions applied to the result of the selection. *Since a function of random variables is also a random variable*, we can view nonground atoms as random variables. Figure 2 illustrates this conception.

Random Individuals. With the random selection semantics, probability formulas can be interpreted as describing the properties of *random individuals*. For example, the formula $P(\text{Flies}(B)) = 90\%$ can be interpreted as “the probability that a randomly selected bird flies is 90%”. In many domains, an intuitive

¹ Halpern allows nonuniform distributions.

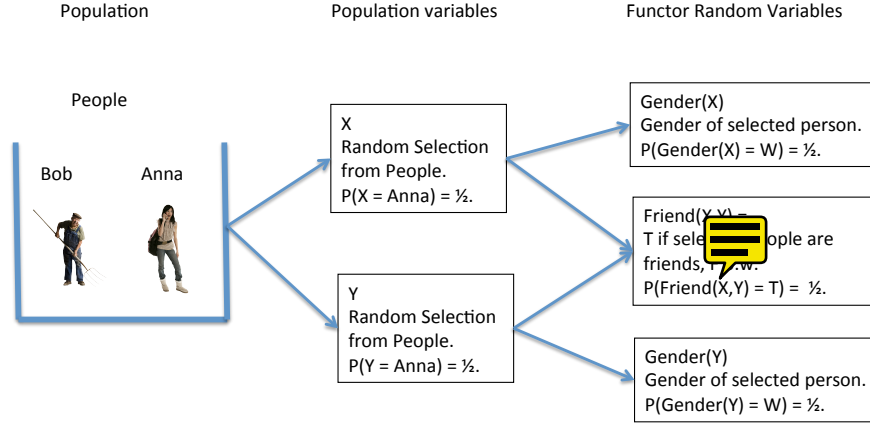


Fig. 2. Viewing population variables as random selections from the associated population, functor nodes are functions of random variables, hence themselves random variables.

interpretation is in terms of typical or normal individuals; for instance, we may read the formula as “the probability that a typical bird flies is 90%”.

Class-level Probabilities. Random selection formulas can also be read as describing the distribution of properties in *classes of individuals*. Thus $P(\text{Flies}(B)) = 90\%$ can be read as “the percentage of birds that fly is 90%”. We therefore refer to probabilities defined by random selection as **class-level probabilities**.

2.4 Mixed Probability Formulas.

Probability formulas that contain both ground and nonground literals express connections between instance-level and class-level probabilities. Halpern investigated general connections between the two probability types [1]. An example would be the axiom schema

$$P(\text{Flies}(B)) = p \leftrightarrow P(\text{Flies}(b) = p)$$

where \leftrightarrow is shorthand for two implications in each direction, and B and b are from the same class (Birds). Intuitively, this equivalence says that if the only thing we know about an individual b is that b is a bird, then the probability that that individual flies is the proportion of birds that fly. Schulte argues that this equivalence principle is valid for statistical-relational inference, and leads to useful constraints on model structures and parameters [16]. The equivalence principle can be operationalized with a statistical-relational system as follows. (1) Learn a model from relational data. (2) Introduce a new constant that denotes an individual that has not been observed in the data (e.g., *new_person*). (3) Apply inference to query the marginal probability that the new individual has

a property (e.g., $P(\text{gender}(\text{new_person}) = M?)$). Because the new person is not linked to any other individuals, the answer to this query should be the frequency with which the property holds in the individual’s population (e.g., the frequency of men). This construction provides a *method for querying an instance-level inference system to obtain class-level probabilities* [1]. In our experiments below, we use it to compare Markov Logic Network predictions with class-level frequencies.

We believe that a unified approach to both instance-level and class-level probabilities is an exciting direction for statistical-relational learning. In the next section we show how a graphical model can support both types of queries at the same time. While our treatment focuses on Bayes nets, it also applies to other logic-based graphical models, such as Markov random fields represented in Markov Logic [17].

check Cussen, probabilistic logic programming

3 Bayes Nets for Relational Data

A **Bayes net structure** is a directed acyclic graph (DAG) G , whose nodes comprise a set of random variables. A **family** is a child node together with its parents. A Bayes net (BN) is a directed acyclic graph with conditional probability parameters. The standard product formula specifies a joint probability to each assignment of values to the nodes in the BN, namely the product of the conditional probabilities $P(\text{child}|\text{parent_values})$ [13, Ch.14.2]. A **Parametrized Bayes Net** (PBN) is a Bayes net whose nodes are atoms. We also refer to a PBN as a **Functor Bayes Net**, or simply Bayes net, and to its atoms as **functor nodes**. This is for brevity and to avoid confusion with the statistical sense of “parametrized”, meaning that values have been assigned to the parameters of the Bayes net. In this paper we consider Bayes nets with nonground atoms only. Figure 1 shows an illustrative Bayes net.

3.1 Instance-level Semantics for Bayes Nets.

The most common semantics for Functor Bayes nets views them as a *template* for a graphical model whose nodes are ground atoms, to support instance-level inferences [18]. A **ground** Bayes net \bar{B} is a directed graph derived from B by instantiating the population variables in the functor nodes in B with all constants of the appropriate type. Via the product formula, the inference graph specifies a joint probability to each assignment of values to the ground atoms. Since a relational structure specifies a value for each ground atom, the set of joint value assignments stands in 1-1 correspondence with the set of relational structures. Therefore *the ground graph defines a probability distribution μ over relational structures*.

A well-known problem for Bayes net template semantics arises in the presence of recursive dependencies or autocorrelations, where the value of an attribute depends on the values of the same attribute for related entities [19]. In this case the ground graph often contains cycles. For example, the ground inference graph for Figure 1 contains an edge $\text{gender}(\text{anna}) \rightarrow \text{gender}(\text{bob})$ as well as an edge $\text{gender}(\text{anna}) \leftarrow \text{gender}(\text{bob})$.

3.2 Class-Level Semantics for Bayes Nets

The random selection semantics can be applied to Bayes Nets, to view them as representing class-level probabilities, without reference to a ground model. Via the standard product formula, a Bayes net B entails a probability value P_B for each joint assignment of values to functor nodes. These are probability formulas with nonground literals. For example, the Bayes net of Figure 1 entails a set of joint probabilities of the form

$$P_B(G(Y) = v_1, F(X, Y) = v_2, G(X) = v_3, CDr(X) = v_4),$$

where we have used the obvious abbreviations for functors, and each v_i is a constant from the appropriate domain. To illustrate, with uniform distributions on $G(Y)$ and $F(X, Y)$, the Bayes net of Figure 1 entails the probability

$$P_B(G(Y) = M, F(X, Y) = T, G(X) = W, CDr(X) = T) = 0.5 \times 0.5 \times 0.3 \times 0.8 = 0.06. \quad (3)$$

The random selection provides two related interpretations of what these joint probabilities mean. (1) As shown in Figure 2, if we view each population variable as a random selection from its domain, then each functor node is a function of one or more random variables, and hence itself a random variable. The Bayes net model therefore represents a joint distribution over random variables as usual. The difference to the nonrelational setting is only that a random variable in the model is a complex object with internal syntactic structure, rather than a simple “flat” object. The random selection semantics provides a meaning for these structured random variables. For example, using the obvious abbreviations for the BN of Figure 1, the joint probability (3) can be read as “if we randomly select two users X and Y , there is a 6% chance that they are friends, one is a man, the other is a woman, and the woman drinks coffee”.

(2) Each joint probability specified by the Bayes net is a conjunction in probability logic. The Bayes net therefore specifies a set of nonground probability formulas. In the terminology of Bacchus, Grove, Koller, and Halpern, such a set is a *statistical knowledge base* [20]. In this view, a Functor Bayes net is a compact graphical representation of a statistical knowledge base, much as a small set of axioms can provide a compact representation of a logical knowledge base.

4 Parameter Learning For Class-Level Probabilities

The data for relational learning constitute a relational substructure drawn from a complete structure (e.g., a web crawl identifies part of a network). When the substructure represents observed data, we refer to it as a **database** and use the notation \mathcal{D} instead of \mathcal{S} . Like other work in statistical-relational learning, we assume that the database is complete [17, 21]. This means that for each individual (node) observed, the database lists its attributes, and for each pair of observed individuals (nodes), the database specifies which links hold between them.

A fundamental question for statistical model selection techniques is how to measure the fit of a Bayes net model to a database. That is, we must specify a likelihood function $P_B(\mathcal{D})$ for a given database. A natural candidate for the likelihood function is the distribution μ_B defined by the template semantics described in Section 3.1. However, this likelihood function is intractable, and in the presence of cyclic dependencies, it is not well defined. Instead, the random selection semantics can be used to define a tractable *pseudo-likelihood* [2]: The pseudo log-likelihood is the expected log-probability assigned by the Bayes net to the links and attributes for a *random grounding* of the population variables. This is defined by the following procedure.

1. Randomly select a grounding for *all* 1st-order variables that occur in the Bayes Net. The result is a ground graph with as many nodes as the original Bayes net.
2. Look up the value assigned to each ground node in the database. Compute the log-likelihood of this joint assignment using the usual product formula; this defines a log-likelihood for the random instantiation.
3. The expected value of this log-likelihood is the *pseudo log-likelihood* of the database given the Bayes net.

X	Y	F(X,Y)	G(X)	CD(X)	G(Y)	BN probability	BN log-p
Anna	Bob	T	F	T	M	$0.5^2 \cdot 0.3 \cdot 0.8 = 0.06$	-2.81
Bob	Anna	T	M	F	F	$0.5^2 \cdot 0.3 \cdot 0.6 = 0.24$	-3.10
Anna	Anna	F	F	T	F	$0.5^2 \cdot 0.5 \cdot 0.8 = 0.26$	-2.30
Bob	Bob	F	M	F	M	$0.5^2 \cdot 0.5 \cdot 0.6 = 0.11$	-2.59

Table 1. An example computation of the pseudo-likelihood for the database and the Bayes net of Figure 1. (Unspecified BN parameters are chosen as uniform solely for computational convenience.) The pseudo log-likelihood is -2.7, the average of the log-probabilities (rightmost column).

Table 1 shows an example computation of the pseudo likelihood. The pseudo likelihood matches the random selection semantics that we have proposed for class-level Bayes nets. It has several attractive theoretical properties [2]. (1) It is closely related to, but different from, other relational pseudo-likelihood measures proposed for graphical models (e.g., Markov random fields). (2) It is invariant under equivalence transformations of the logical predicates (database normalization). (3) For a fixed database \mathcal{D} and Bayes net structure, *the parameter values that maximize the pseudo-likelihood are the conditional empirical frequencies* defined by the database distribution $P_{\mathcal{D}}$ [2, Prop.3.1]. This result is exactly analogous to maximum likelihood estimation for i.i.d. data. In the remainder of the paper we evaluate database frequencies as parameter estimates. We begin with a procedure for computing them.

5 Computing Relational Statistics

We consider the problem of computing the database joint probability $P_{\mathcal{D}}(child_value, parent_values)$ of a nonground formula that describes a family state. Joint probabilities are easily converted to conditional probability parameters in the Bayes net. The computation depends on whether the family formula contains negated relationships. An example of a formula with positive relationships only, would be $P_{\mathcal{D}}(gender(X) = M, Friend(X, Y) = T)$, the frequency of pairs (x, y) who are friends with x male. The required count $\#_{\mathcal{D}}(gender(X) = M, Friend(X, Y) = T)$ can be computed by table joins with SQL, or with optimized virtual join methods, such as tuple ID propagation [22].

5.1 Statistics With One Negated Relationship

A Functor Bayes net represents *uncertainty about relationships* with relationship nodes such as $Friend(X, Y)$. Learning with link uncertainty requires computing sufficient statistics that involve the *absence* of relationships. A naive approach would explicitly construct new data tables that enumerate tuples of objects that are *not* related. However, the number of unrelated tuples is too large to make this scalable (think about the number of user pairs who are *not* friends on Facebook). Getoor et al. observed that we can use a “1-minus trick” when the query formula involves only a single relationship that is negated [10]. For example, we can compute the frequency of pairs (x, y) who are *not* friends with x male, by using the equation

$$P_{\mathcal{D}}(g(X) = M, F(X, Y) = F) = P_{\mathcal{D}}(g(X) = M) - P_{\mathcal{D}}(g(X) = M, F(X, Y) = T)$$

where we have used obvious abbreviations for the functors. Notice that the right-hand side contains no negative relationship literal. This equation is valid for any estimate of the right-hand side quantities.

5.2 Statistics With Multiple Negated Relationships: The Fast Möbius Transform

The **inverse Möbius transform** (IMT) is an optimal algorithm for converting probability values for formulas with positive relationships only, to a complete set of joint probabilities with any number of positive and negative relationships [9]. The IMT was originally described using category theory with lattice structures. Our version is adapted for **joint probability tables** (JP-tables). A JP-table is just like a CP-table whose rows correspond to joint probabilities rather than conditional probabilities.

Consider a set R_1, \dots, R_m of Boolean relationship atoms, and a set f_1, \dots, f_k of attribute atoms. The goal is to compute joint probability estimates $P(R_1 = b_1, \dots, R_m = b_m, f_1 = v_1, \dots, f_k = v_k)$ for all possible Boolean values $b_i \in \{T, F\}$, and all possible values v_i in the range of f_i .

The algorithm begins by estimating all marginal probabilities that involve only positive values; these are called the **Möbius parameters** of P [23, p.239]. For simplicity, consider a fixed conjunction of attribute literals ϕ . Then we compute 2^m estimates such as $P(R_1 = T, \phi)$; $P(R_1 = T, R_2 = T, \phi)$; $P(R_1 = \text{true}, R_3 = T, R_m = T, \phi)$; etc. To represent a Möbius parameter, we allow relationship nodes to take on the value $*$ for “unspecified”. For instance, suppose that the atoms are $Int(S)$, $Reg(S, C)$, $RA(S, P)$. Then the Möbius parameter $P(Int(S) = 1)$ is stored in the JP-table row where $Int(S) = 1$, $Registered(S, C) = *$, $RA(S, P) = *$. The *Möbius extension theorem* shows that the Möbius parameters entail a unique set of values for all joint probabilities [23, p.239]. To compute the entailed values, the IMT uses a local update operation corresponding to the probabilistic identity

$$P(\phi, \mathbf{R}, R = F) := P(\phi, \mathbf{R}) - P(\phi, \mathbf{R}, R = T). \quad (4)$$

where \mathbf{R} is a conjunction of relationship literals that assign Boolean values to relationship atoms. Eq. 4 generalizes the 1-minus trick, and shows how a probability that involves $k + 1$ negated relationships can be determined from two probabilities that each involve only k negated relationships, for $k \geq 0$. The IMT initializes the JP-table with the Möbius parameters that do not contain negated relationships, that is, all relationship nodes have the value T or $*$. It then goes through the relationship nodes R_1, \dots, R_m in order, replaces at stage i all occurrences of $R_i = *$ with $R_i = F$, and applies the local update equation for the probability value for the modified row. At termination, all $*$ values have been replaced by F and the JP-table specifies all joint frequencies. Algorithm 1 gives pseudocode and Figure 3 illustrates the IMT in a schematic example with two relationship nodes.

5.3 Complexity Analysis.

Kennes and Smets [9] provide a thorough theoretical analysis of the IMT. We summarize the main points. (1) The primary property of the IMT is that *it accesses data only about existing links*, never about nonexisting links. A secondary but attractive property of the IMT is that the number of additions performed is $m2^{m-1}$, where m is the number of relationship nodes. A lower bound argument shows that this is optimal [9, Cor.1]. (2) Without a bound on m , computing sufficient statistics in a relational structure is #P-complete [24, Prop.12.4]. In practice, the number m is small, 4 or less.

6 Related Work and Discussion

Statistical Relational Models. To our knowledge, Statistical Relational Models (SRMs) due to Getoor, Taskar and Koller [8], are the only prior statistical models with a class-level probability semantics. A direct empirical comparison is difficult as code has not been released. Comparisons with benchmark cardinality

Algorithm 1 The inverse Möbius transform for parameter estimation in a Parametrized Bayes Net.

Input: database \mathcal{D} ; a set of functor nodes divided into attribute nodes f_1, \dots, f_j and relationship nodes R_1, \dots, R_m .

Output: Joint Probability specifying the data frequencies for each joint assignment to the input functor nodes.

- 1: **for all** attribute value assignments $f_1 := v_1, \dots, f_j := v_j$ **do**
 - 2: initialize the JP-table with the Möbius parameters: set all relationship nodes to either T or $*$; find joint frequencies with data queries.
 - 3: **for** $i = 1$ to m **do**
 - 4: Change all occurrences of $R_i = *$ to $R_i = F$.
 - 5: Update the joint frequencies using (4).
 - 6: **end for**
 - 7: **end for**
-

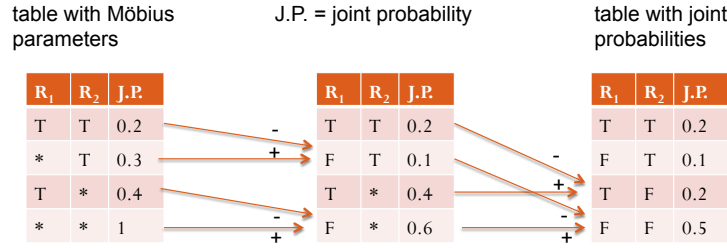


Fig. 3. The inverse Möbius transform with $m = 2$ relationship nodes. For simplicity we omit attribute conditions.

estimation methods from database research report accurate results with SRMs [3].

The syntax of SRMs differ from FBNs and other statistical-relational models in several respects. (1) The SRM syntax is not that of first-order logic, but is derived from a tuple semantics [8, Def.6.3], which is different from the random selection semantics we propose for FBNs. (2) SRMs are less expressive: They cannot express general combinations of positive and negative relationships [8, Def.6.11]. Basically, this restriction stems from the fact that the SRM semantics is based on randomly selecting tuples from *existing tables in the database*. Complements of relationship tables are usually not stored as existing tables (e.g., there is no table that lists the set of user pairs who are *not* friends). The expressive power of SRMs and FBNs becomes essentially equivalent, if the SRM semantics is extended so that random tuples can be drawn from complement tables as well as the original tables. Adding complement tables to SRMs leads to the challenge that SRM learning has to include learning with negated relations, which we address in this paper.

The Complete-Data and Closed-World Assumptions. In logic, the closed-world assumption is that “atomic sentences not known to be true are in fact false” [13,

Ch.8.2.8]. While the closed-world assumption is used in logical reasoning, rather than learning, it is relevant for learning if we view it as *an assumption about how the data are generated*. Specifically in our experiments, we assume that if a link between two individuals is not listed in a database, it does not exist (cf. Figure 1). To illustrate, we may assume that if a student registers in a course, this event is recorded in the university database, so the absence of a record implies that the student has not registered. If this assumption is not warranted for a particular data generating mechanism, the complete-data assumption fails, because for some pairs of individuals, the data does not indicate whether a link between does not actually exist or simply has not been recorded. One way to deal with missing link data is to use imputation methods [25, 26]. Another approach is downsampling existing links by adding random negative links [21, 27]. The important point for this paper is that our methods can be used no matter how link frequencies are estimated, whether from corrected or uncorrected observed link counts. Specifically, given estimates for the Möbius parameters, the Möbius transform finds the probabilities for negative link events that those parameters determine. This computation assumes nothing but the laws of the probability calculus.

Parfactors and Lifted Inference. Lifted inference aims to speed up instance-level inferences by computing *parfactors*, that is, counts of equivalent events, rather than repeating equivalent computations [7]. Because of the similarity with computing event counts in a database, Schulte and Khosravi refer to learning with the pseudo-likelihood as lifted learning [28]. Lifted inference represents event counts as parfactors. The counting algorithms in this paper can likely be applied to computing parfactors that involve negated relations. The motivation for parfactors, however, is as a means to the end of computing instance-level predictions (e.g., predict the gender of Sam). In contrast, our motivation is to learn class frequencies as an end in itself. Because parfactors are used with instance-level inferences, they usually concern classes defined with reference to specific individuals, whereas the statistics we model in this paper are at the class-level only.

7 Evaluation

All experiments were done on a QUAD CPU Q6700 with a 2.66GHz CPU and 8GB of RAM. We evaluated the algorithm on real-world datasets. The datasets and our code are available on the Web [29].

7.1 Datasets

We used four benchmark real-world databases, with the modifications by [30], which contains details and references.

Mondial Database. A geography database. Mondial features a self-relationship, *Borders*, that indicates which countries border each other.

Hepatitis Database. A modified version of the PKDD’02 Discovery Challenge database.

Financial A dataset from the PKDD 1999 cup.

MovieLens. A dataset from the UC Irvine machine learning repository.

To obtain a Bayes net structure for each dataset, we applied the learn-and-join algorithm [30] to each database. This is the state-of-the-art structure learning algorithm for PBNs; for an objective function, it uses the pseudo-likelihood described in Section 4. We also conducted experiments with synthetic graphs and datasets. The results are similar to those on real-life datasets. We omit details because of space constraints. Our implementation makes use of version 4.3.9-0 of CMU’s Tetrad package [31].

7.2 Learning Times

Table 2 shows the runtimes for computing parameter values. The Complement method uses SQL queries that explicitly construct tables for the complement of relationships (tables that contain tuples of unrelated entities), while the IMT method uses the inverse Möbius transform to compute the conditional probabilities. The IMT is faster by orders of magnitude, ranging from a factor of 15–237.

7.3 Inference

The basic inference task for Bayes nets is answering probabilistic queries. If the given Bayes net structure matches the true distribution (i.e., is an I-map), then correct parameter values lead to correct predictions. Thus the performance on queries has been used to evaluate parameter learning [32]. We randomly generate queries for each dataset according to the following procedure. First, choose a target node V 100 times, and go through each possible value a of V such that $P(V = a)$ is the probability to be predicted. For each value a , choose the number k of conditioning variables, ranging from 1 to 3. Select k variables V_1, \dots, V_k and corresponding values v_1, \dots, v_k . The query to be answered is then $P(V = a | V_1 = v_1, \dots, V_k = v_k)$.

As in [3], we evaluate queries after learning parameter values on the entire database (complete population). Thus the Bayes Net is viewed as a statistical summary of the data rather than as generalizing from a sample. Inference is carried out using the Approximate Updater in CMU’s Tetrad program. Figure 4 shows the query performance for each database. A point (x, y) on a curve indicates that there is a query such that the true probability value in the database is x and the probability value estimated by the model is y . The Bayes net inference is close to the ideal identity line, with an average error of less than 1%.

Comparison With Markov Logic Networks. Halpern’s method described in Section 2.4 allows us to apply an instance-level inference model for frequency estimates. To benchmark our results, we compare PBN inferences with Markov Logic Network (MLN) frequency estimates. In graphical terms, MLNs can be

Fig. 4. *Left:* Query Performance: Estimated vs. true probability. The average error and standard deviation are shown as well. Number of queries/average inference time per query: Mondial, 506/0.08sec; MovieLens, 546/0.05sec; Hepatitis, 489/0.1sec; Financial, 140/0.02sec. *Right:* The estimates of conditional probability parameters, averaged over 10 random subdatabases and all BN parameters. Error (absolute difference) in conditional probability estimates. The median observation is the center line and the box comprises 75% of the observed values. The whisker indicates the maximum acceptable value (1.5 IQR upper).

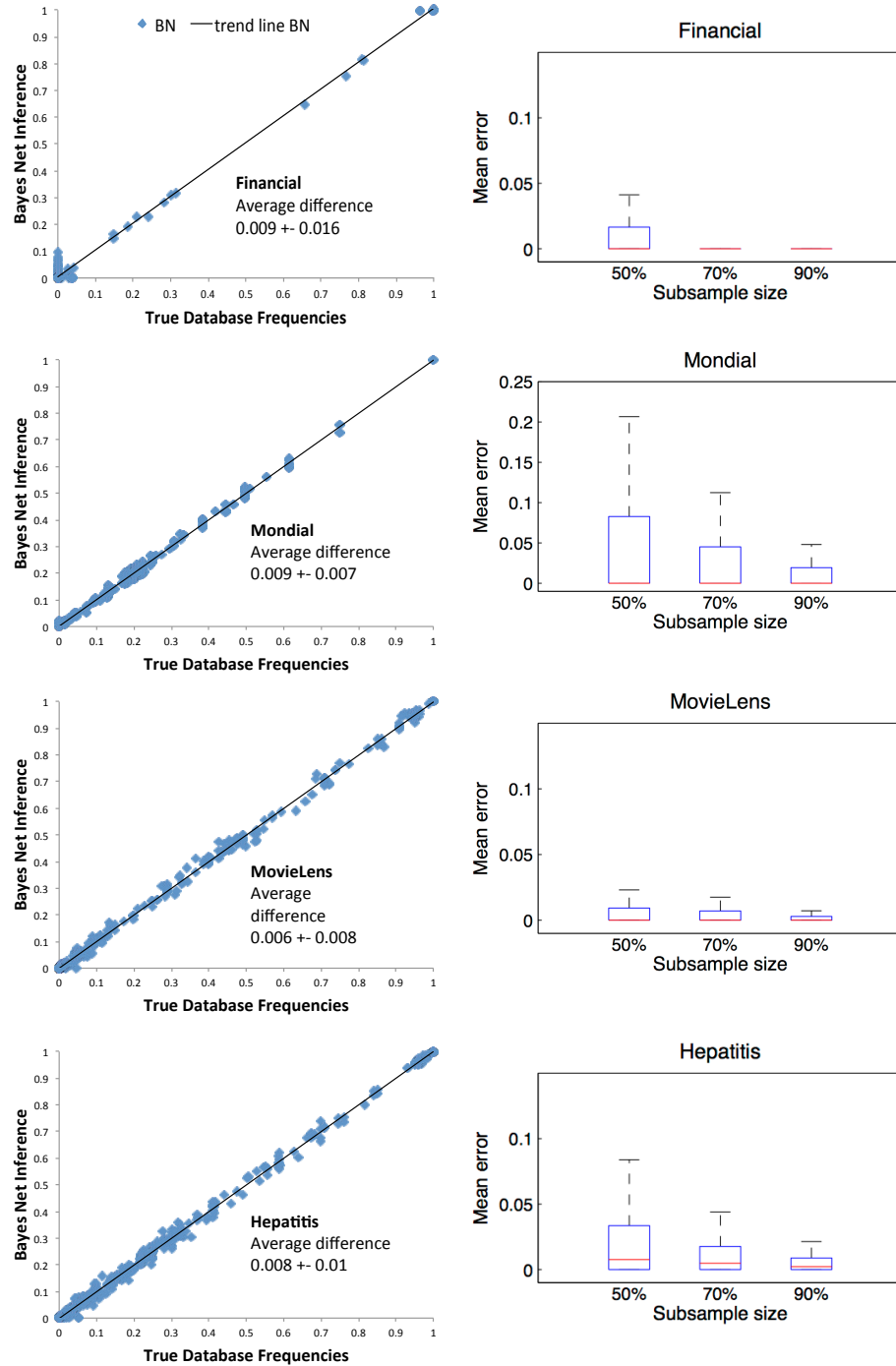


Table 2. Learning time results (sec) for the inverse Möbius transform vs. constructing complement tables. For each database, we show the number of tuples, and the number of parameters (conditional probabilities) in the fixed Bayes net structure.

Database	Parameters	#tuples	Complement	IMT	Ratio
Mondial	1618	814	157	7	22
Hepatitis	1987	12,447	18,246	77	237
Financial	10926	17,912	228,114	14,821	15
MovieLens	326	82,623	2,070	50	41

viewed as defining an undirected relational model. They are a good comparison point because (1) they are currently one of the most active areas of SRL research; the Alchemy system provides open-source, state-of-the-art learning and inference software [33]. (2) They do not require the specification of further components (e.g., a combining rule or aggregation function). (3) An undirected model can accommodate recursive relationships (cyclic dependencies). We compare the following learning algorithms [34]. We use the MC-SAT algorithm for MLN inference as implemented in Alchemy.

BN: Bayes net parametrized with maximum pseudo likelihood estimates.

LHL: The LHL algorithm is a structure learning algorithm that produces a parametrized MLN.

LSM: The state-of-the-art Alchemy structure learning algorithm that produces a parametrized MLN. In experiments by Kok and Domingos, LSM outperformed previous MLN learners.

The Bayes net models provide much more accurate frequency estimates than the MLN models, with an average improvement of 10% or more. This shows that even with complete-population data, constructing an accurate model of relational frequencies is not a trivial task.

Dataset	Average error		
	BN	MLN(LSM)	MLN(LHL)
Mondial	0.9%	8.6%	10.5%
Hepatitis	0.8%	11.2%	13.2%
Financial	0.9%	9.1%	NT
Movielens	0.6%	14.2%	NT

Table 3. The frequency query performance of Bayes nets vs. Markov Logic Networks. We show the average absolute error over all random queries between the predicted frequency and the true database frequencies. NT denotes non-termination within the system resources.

7.4 Conditional Probabilities

In the previous inference results, the correct query answers are defined by the entire population (benchmark database), and the model is also trained on the entire database. To study parameter estimation at different sample sizes, we continue to define the correct value in terms of the entire population, but train the model on $N\%$ of the data for varying N . Conceptually, we treated each benchmark database as specifying an entire population, and consider the problem of estimating the complete-population frequencies from partial-population data. The $N\%$ parameter is uniform across tables and databases. We employed standard subgraph subsampling [35, 30], which selects entities from each entity table uniformly at random and restricts the relationship tuples in each subdatabase to those that involve only the selected entities. Subgraph sampling provides complete link data, and matches the random selection semantics which is based on random draws from a population. It is applicable when the observations include positive and negative link information (e.g., not listing two countries as neighbors implies that they are not neighbors). The subgraph method satisfies an ergodic law of large numbers: as the subsample size increases, the subsample relational frequencies approach the population relational frequencies.

Figure 4 plots the difference between the true conditional probabilities and the MPLE estimates. With increasing sample size, **MPLE** estimates approach the true value in all cases. Even for the smaller sample sizes, the median error is close to 0, confirming that most estimates are very close to correct. As the box plots show, the 3rd error quartile of estimates is bound within 10% on Mondial, the worst case, and within less than 5% on the other datasets.

8 Conclusion



Introduced a new interpretation of Bayes nets as models of class-level statistics in a relational structure. The class-level interpretation is based on the classic random selection semantics for probability logic. For parameter learning we utilized the empirical joint frequencies, which can be feasibly computed using the inverse Möbius transform, even for frequencies concerning negated links. In evaluation on four benchmark databases, the maximum pseudo-likelihood estimates approach the true conditional probabilities as observations increase. The fit is good even for medium data sizes.

Overall, our simulations show that the Bayes net models, with maximum pseudo-likelihood parameter estimates, provide excellent estimates of class-level probabilities. Previous research indicates that the same Bayes net models also lead to state-of-the-art instance-level inferences [28]. Indeed, we would argue that we obtain good performance on instance-level prediction *because* the learned Bayes net structures are good models of class-level probabilities. Intuitively, if a model is wrong about the general domain dependencies, it is likely to be wrong about individual dependencies as well. For example, suppose that a model posits that grades and the difficulty of a course are generally negatively correlated,

when in fact they are positively correlated. This model will not give accurate predictions when we query the model to predict the difficulty of a specific course, given the grades that students have obtained in that course. Therefore aligning class-level and instance-level probabilities is a desirable goal for statistical-relational modelling.

References

- [1] Halpern, J.Y.: An analysis of first-order logics of probability. *Artificial Intelligence* **46**(3) (1990) 311–350
- [2] Schulte, O.: A tractable pseudo-likelihood function for Bayes nets applied to relational data. In: *SIAM SDM*. (2011) 462–473
- [3] Getoor, L., Taskar, B., Koller, D.: Selectivity estimation using probabilistic models. *ACM SIGMOD Record* **30**(2) (2001) 461–472
- [4] Bacchus, F.: Representing and reasoning with probabilistic knowledge: a logical approach to probabilities. MIT Press, Cambridge, MA, USA (1990)
- [5] Maier, M., Taylor, B., Oke, A., Jensen, D.: Learning causal models of relational domains. In: *AAAI*. (2010)
- [6] McMahan, B.J., Pan, G., Porter, P., Vardi, M.Y.: Projection pushing revisited. In: *EDBT*. (2004) 441–458
- [7] Poole, D.: First-order probabilistic inference. In: *IJCAI*. (2003) 985–991
- [8] Getoor, L.: Learning Statistical Models From Relational Data. PhD thesis, Department of Computer Science, Stanford University (2001)
- [9] Kennes, R., Smets, P.: Computational aspects of the Möbius transformation. In: *UAI*. (1990) 401–416
- [10] Getoor, L., Friedman, N., Koller, D., Pfeffer, A., Taskar, B.: Probabilistic relational models. [36] chapter 5 129–173
- [11] Chiang, M., Poole, D.: Reference classes and relational learning. *Int. J. Approx. Reasoning* **53**(3) (2012) 326–346
- [12] Milch, B., Marthi, B., Russell, S., Sontag, D., Ong, D., Kolobov, A.: Blog: Probabilistic models with unknown objects. In: *Statistical Relational Learning*. MIT Press (2007) 373–395
- [13] Russell, S., Norvig, P.: *Artificial Intelligence: A Modern Approach*. Prentice Hall (2010)
- [14] Ullman, J.D.: *Principles of database systems*. 2. Computer Science Press (1982)
- [15] Babcock, B., Chaudhuri, S.: Towards a robust query optimizer: a principled and practical approach. In: *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*. SIGMOD '05, New York, NY, USA, ACM (2005) 119–130
- [16] Schulte, O.: Challenge paper: Marginal probabilities for instances and classes. *ICML-SRL Workshop on Statistical Relational Learning*. (June 2012)
- [17] Domingos, P., Lowd, D.: *Markov Logic: An Interface Layer for Artificial Intelligence*. Morgan and Claypool Publishers (2009)
- [18] Getoor, L., Taskar, B.: Introduction. [36] 1–8
- [19] Neville, J., Jensen, D.: Relational dependency networks. *Journal of Machine Learning Research* **8** (2007) 653–692
- [20] Bacchus, F., Grove, A.J., Koller, D., Halpern, J.Y.: From statistics to beliefs. In: *AAAI*. (1992) 602–608

- [21] Khot, T., Natarajan, S., Kersting, K., Shavlik, J.W.: Learning markov logic networks via functional gradient boosting. In: ICDM. (2011) 320–329
- [22] Yin, X., Han, J., Yang, J., Yu, P.S.: Crossmine: Efficient classification across multiple database relations. In: Constraint-Based Mining and Inductive Databases. (2004) 172–195
- [23] Lauritzen, S.L.: Graphical Models (Oxford Statistical Science Series). Oxford University Press, USA (July 1996)
- [24] Domingos, P., Richardson, M.: Markov logic: A unifying framework for statistical relational learning. [36]
- [25] Hoff, P.D.: Multiplicative latent factor models for description and prediction of social networks. Computational and Mathematical Organization Theory (2007)
- [26] Namata, G.M., Kok, S., Getoor, L.: Collective graph identification. In: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. KDD '11, New York, NY, USA, ACM (2011) 87–95
- [27] Yang, S.H., Long, B., Smola, A., Sadagopan, N., Zheng, Z., Zha, H.: Like like alike: joint friendship and interest propagation in social networks. In: Proceedings of the 20th international conference on World wide web. WWW '11, New York, NY, USA, ACM (2011) 537–546
- [28] Schulte, O., Khosravi, H.: Learning graphical models for relational data via lattice search. Machine Learning **88:3** (2012) 331–368
- [29] Khosravi, H., Man, T., Hu, J., Gao, E., Schulte, O.: Learn and join algorithm code. URL = <http://www.cs.sfu.ca/~oschulte/jbn/>.
- [30] Khosravi, H., Schulte, O., Man, T., Xu, X., Bina, B.: Structure learning for Markov logic networks with many descriptive attributes. In: AAAI. (2010) 487–493
- [31] The Tetrad Group: The Tetrad project (2008) <http://www.phil.cmu.edu/projects/tetrad/>.
- [32] Allen, T.V., Singh, A., Greiner, R., Hooper, P.: Quantifying the uncertainty of a belief net response: Bayesian error-bars for belief net inference. Artif. Intell. **172**(4-5) (2008) 483–513
- [33] Kok, S., Summer, M., Richardson, M., Singla, P., Poon, H., Lowd, D., Wang, J., Domingos, P.: The Alchemy system for statistical relational AI. Technical report, University of Washington. (2009) Version 30.
- [34] Kok, S., Domingos, P.: Learning Markov logic networks using structural motifs. In: ICML. (2010) 551–558
- [35] Frank, O.: Estimation of graph totals. Scandinavian Journal of Statistics **4:2** (1977) 81–89
- [36] Getoor, L., Taskar, B.: Introduction to statistical relational learning. MIT Press (2007)