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, British Columbia, Canada

Abstract. Class-level models capture relational statistics over object attributes and their connecting links, answering questions such as "what is the percentage of friendship pairs where both friends are women?" Class-level relationships are important in themselves, and they support applications like policy making, strategic planning, and query optimization. We represent class statistics using Parametrized Bayes Nets (PBNs), a first-order logic extension of Bayes nets. Queries about classes require a new semantics for PBNs, as the standard grounding semantics is only appropriate for answering queries about specific ground facts. We propose a novel random selection semantics for PBNs, which does not make reference to a ground model, and supports class-level queries. The parameters for this semantics can be learned using the recent pseudo-likelihood measure [1] as the objective function. This objective function is maximized by taking the empirical frequencies in the relational data as the parameter settings. We render the computation of these empirical frequencies tractable in the presence of negated relations by the inverse Möbius transform. Evaluation of our method on four benchmark datasets shows that maximum pseudo-likelihood provides fast and accurate estimates at different sample sizes.

1 Introduction

Many applications store data in relational form. Relational data introduces the machine learning problem of *class-level frequency estimation*: building a model that can answer statistical queries about classes of individuals in the database [2]. Examples of such queries include:

- What fraction of birds fly?—referring to the class of birds.
- What fraction of the grades awarded to highly intelligent students are As?—referring to the class of (student, course-grade) pairs.
- In a social network, what is the percentage of friendship pairs where both are women?—referring to the class of friendship links.

As the examples illustrate, class-level probabilities concern the frequency, proportion, or rate of generic events and conditions, rather than the attributes and links of individual entities. Estimates of class-level frequency have 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 [3, 4]. Often such patterns can be expressed as *generic statements* about the average member of a class, like "intelligent students tend to take difficult courses".

Policy making and strategic planning. An administrator may explore the program characteristics that 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.

Query optimization. A statistical model predicts a probability for given table join conditions that can be used to estimate the cardinality of a database query result [2]. Cardinality estimation is used to minimize the size of intermediate join tables [6].

In this paper, we apply Bayes nets to the problem of class-level frequency estimation. We define the semantics, describe a learning algorithm, and evaluate the method's effectiveness.

Semantics. We build 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 or template 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 *instance-level 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 Getoor pointed out [8], the ground graph is not appropriate for answering class-level queries, which are about generic rates and percentages, not about any particular individuals.

We support class-level queries via a class-level semantics for Parametrized Bayes nets, based on Halpern's classic random selection semantics for probabilistic first-order logic [3, 4]. Halpern's semantics views statements with first-order variables as expressing statistical information about classes of individuals. For instance, the claim "60% is the percentage of friendship pairs where both friends are women" could be expressed by the formula

$$P(gender(X) = female, gender(Y) = female | Friend(X, Y) = T) = 60\%.$$

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 these limitations by using a relational pseudo-likelihood measure for Bayes nets [1] 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. The estimator that maximizes the parameters of this pseudo-likelihood function (MPLE) has a closed-form solution: the parameters are the empirical frequencies, as with classical i.i.d. maximum likelihood estimation. However, computing these empirical frequencies is nontrivial for negated relationships because enumerating the complement of a relationship table is computationally infeasible. We show that the MPLE can be made tractable for negated relationship using the Möbius transform [9].

Evaluation. 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.

Paper Organization. We begin with relational Bayes net models and the random selection semantics for Bayes nets (Sections 2 and 3). Section 4 describes parameter learning and Section 5 presents the inverse Möbius transform for computing relational statistics. Simulation results are presented in Section 6, showing the run time cost of estimating parameters, together with evaluations of their accuracy. We provide deeper background on the connections between random selection semantics and probability logic in Section 7 and summarize related work in Section 8.

2 Bayes Nets for Relational Data

Poole introduced the Parametrized Bayes net (PBN) formalism that combines Bayes nets with logical syntax for expressing relational concepts [7]. We adopt the PBN formalism, following Poole's presentation.

2.1 Relational Concepts

We introduce here a minimum of logical terminology. In Section 7 we present a more complete probabilistic logic. A **population** is a set of individuals. Individuals are denoted by lower case expressions (e.g., *bob*). A **population variable** is capitalized. An instantiation of a population variable X selects a member x of the associated population. An instantiation corresponds to the logical concept of grounding. A **functor** represents a mapping $f : \mathcal{P}_1, \ldots, \mathcal{P}_a \to V_f$ where f is the name of the functor, and V_f is the output type or **range** of the functor. In this paper we consider only functors with a finite range, disjoint from all populations. If $V_f = \{T, F\}$, the functor f is a (Boolean) **predicate**; other functors are called **attributes**. A predicate with more than one argument is called a **relationship**. We use lowercase for attribute functors and uppercase for predicates.

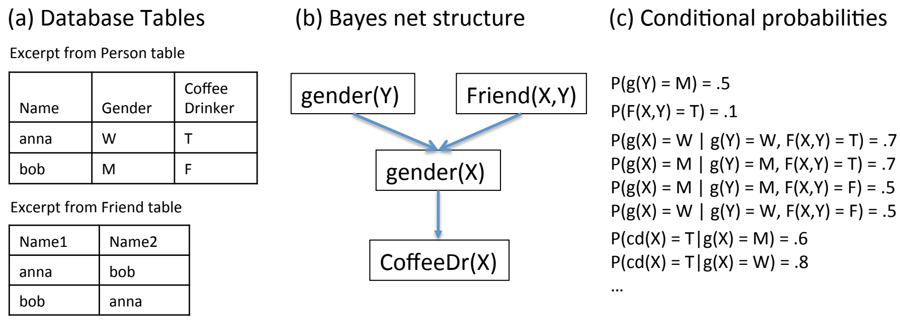


Fig. 1. (a) A relational structure represented as tables. By convention, a pair not listed in the *Friend* table are not friends. The *Friend* relation is symmetric: $Friend(X, Y) = Friend(Y, X)$. Only part of the rows are shown. (b) A Bayes net structure for this database. $Friend(X, Y)$ is a relationship node, while the other three are attribute nodes. (c) The conditional probabilities for the Bayes net structure based upon the entries in the entire database.

A **relational structure** specifies the value of a functor for any tuple of appropriate individuals [11]. A relational structure can be visualized as a complex heterogeneous network [12, Ch.8.2.1]: individuals are nodes, attributes of individuals are node labels, relationships correspond to (hyper)edges, and attributes of relationships are edge labels. In relational databases, a relational structure is represented as a set of tables that list individuals and their attributes; see Figure 1(a). The set of individual pairs or tuples that are linked by a relationship is usually much smaller than the set that are not so linked, so database tables list only the existing relationships.

2.2 Bayes Nets

A **Bayes Net (BN)** is a directed acyclic graph (DAG) whose nodes comprise a set of random variables and conditional probability parameters. For each assignment of values to the nodes, the joint probability is specified by the product of the conditional probabilities, $P(child|parent\_values)$.

A **Parametrized random variable** is of the form $f(X_1, \ldots, X_a)$, where the populations associated with the variables are of the appropriate type for the functor. A **Parametrized Bayes Net** (PBN) is a Bayes net whose nodes are Parametrized random variables [7]. In the remainder of this paper we follow [13] and use the terms **functor random variable** and **functor Bayes Net** instead (FBN), for the following reasons. (1) The name "Parametrized" denotes a semantics that views the Bayes net as a template for a ground graph structure. Our random selection semantics does not refer to a ground model. (2) The usual statistical meaning of "parametrized" is that values have been assigned to the model parameters. We do not assume that such values have been assigned; learning them is a key topic of this paper.

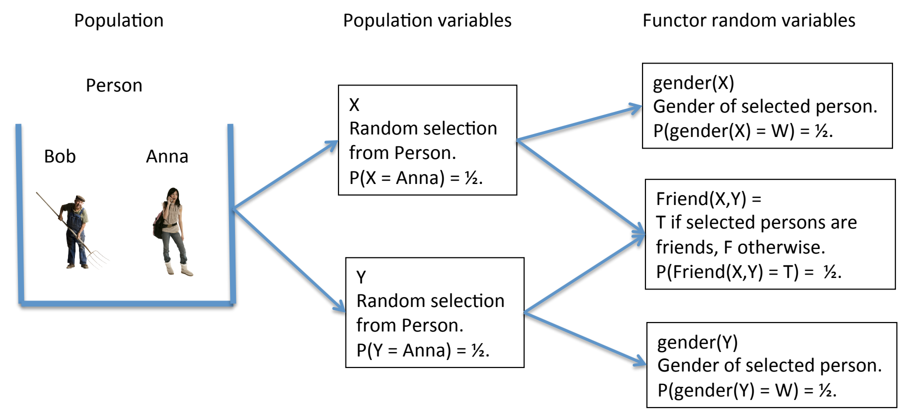


Fig. 2. Viewing population variables as random selections from the associated population, the nodes represent functions of random variables, hence are themselves random variables. Probabilities are computed assuming the relations in Fig. 1 (a) are complete (the database only describes Bob and Anna).

In our view, a Functor Bayes net is a type of Bayes net whose nodes are complex objects with an internal syntax derived from the relational structure. Where the distinction between BNs in general and FBNs is not important, we simply refer to FBNs as Bayes nets and to functor random variables as functor nodes or nodes. Figure 1(b) shows an illustrative FBN. The next section presents a novel semantics of Functor Bayes nets as representing relational statistics.

3 Random Selection Semantics for Bayes Nets.

We provide a semantics that views a functor Bayes net as a representation of a joint probability distribution over its nodes [12, Sec.14.2]. The semantics is for nodes that refer to population variables, rather than for ground nodes that refer to individuals as in Poole's original paper [7]. Such a nonground node can be given a probabilistic interpretation in terms of *random selections*.

Consider first a node with a single population variable, e.g., $gender(X)$. We view X as a random variable that selects a member x from its population according to a uniform population distribution $P(X = x)$. The functor *gender* then represents *a function of a random variable*, and hence is itself a random variable. Figure 2 illustrates this interpretation.

In the case of a functor with several population variables $f(X_1, \ldots, X_a)$, we view each population variable as representing an independent selection from its associated population (thus variables for the same population represent i.i.d. selections). This defines a distribution over a-tuples of individuals from the appropriate population. Again, *any functor* $f(X_1, \ldots, X_a)$ *then defines a function of a random variable*, and hence is itself a random variable. Figure 2 illustrates the random selection interpretation for the functor $Friend(X, Y)$. As the random selection semantics is a key concept in this paper, we discuss two interpretations.

Random Individuals. Under random selection semantics, probability formulas can be interpreted as describing the properties of *random individuals.* For example, the formula $P(gender(X) = M) = 50\%$ can be interpreted as "the probability is 50% that a randomly selected person is a man". A related interpretation that is often helpful is in terms of typical individuals [3]; for instance, we may read the formula $P(Flies(B) = T) = 90\%$ 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 we can read $P(gender(X) = M) = 50\%$ as "the percentage of persons that are men is 50%". Similarly, we may read the formula $P(Flies(B) = T) = 90\%$ as "the percentage of birds that fly is 90%". We therefore refer to probabilities defined by random selection as **class-level probabilities**.

Random Selection Semantics for Bayes Nets. Random selection semantics can be applied to view Bayes nets 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 its node. These are probability assignments that refer to population variables rather than individuals. For example, the Bayes net structure of Figure 1 (b) entails 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 range. For example, the conditional probabilities given in Figure 1(c) entail the joint probability

$$P_B(g(X) = W, g(Y) = M, F(X,Y) = T, CDr(X) = T) = .028. \qquad (1)$$

The random selection semantics for the nodes provides an interpretation of these joint probabilities. For example, the joint probability (1) can be read as "if we randomly select two persons X and Y, the chance is 6% that one is a woman, the other is a man, they are friends, and the woman drinks coffee".

Since the random selection semantics is not defined in terms of a ground model, it can be applied even in the presence of cyclic dependencies in the data. For instance, grounding the Bayes net of Figure 1 shows a cyclic dependency between the genders of different users. The random selection semantics still applies, as shown in our examples, since it concerns class-level dependencies. The BN of Figure 1 illustrates that *the dependency structure among class-level random variables can be acyclic even when the dependency structure among ground instances is not.*

In logic, the term "semantics" is used in a different sense than in the context of statistical models, to refer to truth conditions for logical sentences [12, Ch.7.4.2]. In section 7, we show that the random selection semantics for Bayes

Table 1. An example computation of the pseudo-likelihood for the database and the Bayes net of Figure 1 (b). Columns for functors include the grounded value and its conditional probability. The dotted row indicates entries for combinations including the other instances of *Person*. The pseudo log-likelihood is the mean of the ln p column.

X	Y	g(Y)	F(X,Y)	g(X)	cd(X)	Joint p	ln p
anna	anna	W (0.5)	F (0.9)	W (0.5)	T (0.8)	0.180	-1.71
anna	bob	M (0.5)	T (0.1)	W (0.3)	T (0.8)	0.012	-4.42
bob	anna	W (0.5)	T (0.1)	M (0.3)	F (0.4)	0.006	-5.12
bob	bob	M (0.5)	F (0.9)	M (0.5)	F (0.4)	0.090	-2.41
...	...	...	...	...	...	...	...

nets that we have presented is in agreement with Halpern's truth-conditional type 1 semantics for probabilistic logic. We next turn to learning and describe a method for learning Bayes net parameters that represent class-level probabilities.

4 Parameter Learning For Class-Level Probabilities

In a nonrelational learning setting, it is common to view a data table as representing information about individuals in a finite subset drawn from a large, possibly infinite population. In relational learning, we view a **database** $\mathcal{D}$ as representing a finite substructure of a large, possibly infinite relational structure. We make the standard assumption that the database is complete [14, 15]: 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. Note that completeness is a statement about the information about individuals in the database. A complete database may contain only a subset of the individuals in the population about which we wish to make inferences.

A fundamental question for statistical model selection is how to measure the fit of the model. That is, we must specify a likelihood function $P_B(\mathcal{D})$ for a given database. The random selection semantics can be used to define a tractable *pseudo-likelihood* [1]: The pseudo log-likelihood is the expected log-probability assigned by the Bayes net to the links and attributes for a *random instantiation* of the population variables. This is defined by the following procedure.

1. Randomly select a grounding for *all* population variables that occur in the Bayes net. The result is a ground graph with as many nodes as the original.
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 this random instantiation.
3. The expected value of this log-likelihood—the mean over all instantiations—is the *pseudo log-likelihood* of the database given the Bayes net.

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 [1]. (1) Because it does not make reference to a single ground model, the measure is well defined even in the presence of cyclic dependencies in the data. For instance, grounding the Bayes net of Figure 1 shows a cyclic dependency between the genders of different users. Nonetheless a pseudo-likelihood value can be computed as in Table 1. The pseudo-likelihood can be used to effectively learn cyclic or recursive dependencies [13]. (2) It is closely related to relational pseudo-likelihood measures proposed for other graphical models (e.g., Markov random fields). (3) It is invariant under equivalence transformations of the logical predicates (database normalization). (4) For a fixed database $\mathcal{D}$ and Bayes net structure, *the parameter values that maximize the pseudo-likelihood are the conditional empirical frequencies observed in the database* [1, Prop.3.1]. We refer to these frequencies as the **database probabilities.** This result is exactly analogous to maximum likelihood estimation for i.i.d. data, where the empirical frequencies in an i.i.d. sample maximize the model likelihood.

The database probability distribution $P_{\mathcal{D}}$ of a node assignment is the number of instantiations of the population variables in the functor nodes that satisfy the assignment in the database, divided by the number of all possible instantiations. The formal definition is as follows [11]:

- Let $f_1 = v_1, \ldots, f_n = v_n$ be an assignment of appropriate values to n nodes.
- Let $X_1, \ldots, X_\ell$ be the population variables that appear in the nodes, with associated population sizes $|\mathcal{P}_1|, \ldots, |\mathcal{P}_\ell|$.
- Let $\#_{\mathcal{D}}(f_1 = v_1, \ldots, f_n = v_n)$ be the number of instantiations or groundings of the population variables that satisfy the assigned values in $\mathcal{D}$.
- Then the database probability of the value assignment is given by

$$P_{\mathcal{D}}(f_1 = v_1, \ldots, f_n = v_n) = \frac{\#_{\mathcal{D}}(f_1 = v_1, \ldots, f_n = v_n)}{|\mathcal{P}_1| \times \cdots \times |\mathcal{P}_\ell|}.$$

Example. Following Figure 1(b), where the complete person population is $\{anna, bob\}$, we have

$$P_{\mathcal{D}}(gender(X) = M) = 1/2$$

and

$$P_{\mathcal{D}}(Friend(X, Y) = T, gender(X) = M, gender(Y) = W) = 1/4.$$

In the next section we consider how to compute database probabilities.

5 Computing Relational Statistics

In this section we describe algorithms for computing probability estimates for the parameters $P_{\mathcal{D}}(child\_value, parent\_values)$ in a functor Bayes net. The parameters can be estimated independently for each child node. We refer to a joint specification of values $(child\_value, parent\_values)$ as a **family state**. Consider

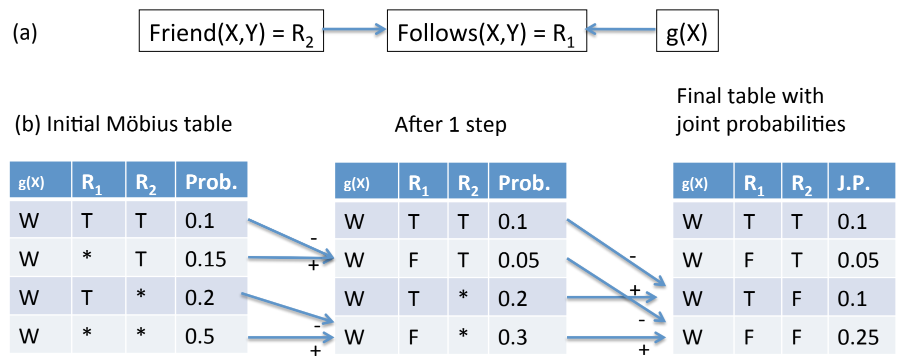


Fig. 3. (a) A Bayes net with two relationship nodes. (b) An illustrative trace of the inverse Möbius transform (see text).

a child node specifying a relationship R_1 whose parents comprise relationship nodes $R_2, \ldots, R_m$, and attribute nodes $f_1, \ldots, f_j$. The algorithms below can be applied in the same way in the case where the child node is an attribute node. The number of family states r is the cardinality of the Cartesian product of the ranges for every node in the family. The Bayes net parameters are r conditional probabilities of the form (we separate relationships from attributes by a semicolon)

$$P(R_1 = b_1 | R_2 = b_2, \ldots, R_m = b_m; f_1 = v_1, \ldots, f_j = v_j), \tag{2}$$

where the b_i values are Boolean and each v_j is from the domain of f_j. Figure 3 (Top) provides an example with $R_1 = \mathit{Friend}(X, Y)$, $R_2 = \mathit{Follows}(X, Y)$, and $f_1 = \mathit{gender}(X)$. In this example, all nodes are binary, so the CP-table requires the specification of $r = 2 \times 2 \times 2 = 8$ values.[1]

Conditional probabilities can be easily estimated from the *sufficient statistics*, which are the database probabilities for a family state:

$$P_{\mathcal{D}}(R_1 = b_1, R_2 = b_2, \ldots, R_m = b_m; f_1 = v_1, \ldots, f_j = v_j). \tag{3}$$

For the rest of this section, we focus on computing these joint probabilities. So long as a database probability involves only positive relationships, the computation is straightforward. For example, in $P_{\mathcal{D}}(\mathit{gender}(X) = M, \mathit{Friend}(X, Y) = T)$, the value $\#_{\mathcal{D}}(\mathit{gender}(X) = M, \mathit{Friend}(X, Y) = T)$, the count of friendship pairs (x, y) where x is male and the *Friend* relationship is true, can be computed by regular table joins or optimized virtual joins [16].

[1] Although only 4 of these values are independent, we take the number of parameters to be $r = 8$ for simplicity of exposition of the inverse Moebius transform in the next section.

Computing joint probabilities for a family containing one or more negative relationships is harder. 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). In their work on learning Probabilistic Relational Models with existence uncertainty, Getoor et al. provided a "1-minus trick" for the special case of estimating a conditional probability with only a single negated relationship [10, Sec.5.8.4.2]. They did not treat parameter learning with multiple negated relationships.

5.1 Statistics With Multiple Negated Relationships: The Fast Möbius Transform

The general case of multiple negative relationships can be efficiently computed using the **fast inverse Möbius transform** (IMT), or Möbius transform for short. We compute the joint probabilities (3) for r family states by first computing the r Möbius parameters of the joint distribution, then using the inverse Möbius transform to transform the Möbius parameters into the desired joint probabilities. Figure 4 provides an overview of the computation steps. Because the Möbius parameters involve probabilities for events with *positive relationships only*, they can be estimated directly from the data. We next define the Möbius parameters, then explain the IMT.

Let $\mathbb{B} = B_1, \ldots, B_m$ be a set of binary random variables with possible values 0 or 1, and P be the joint distribution that specifies 2^m probabilities, one for each possible assignment of values to the m binary variables. For any subset $\mathbf{B} \subseteq \mathbb{B}$ of the variables, let $P(\mathbf{B} = \mathbf{1})$ denote the probability that the variables in $\mathbf{B}$ are assigned the value 1, leaving the value of the other variables unspecified. The **Möbius parameters** of the distribution P are the values $P(\mathbf{B} = \mathbf{1})$ for all subsets $\mathbf{B} \subseteq \mathbb{B}$[17, Sec.3]. There are 2^m Möbius parameters for m binary variables, with $0 \leq P(\mathbb{B} = \mathbf{1}) \leq P(\mathbf{B} = \mathbf{1}) \leq P(\emptyset = \mathbf{1}) = 1$.

For a family state, if we fix the values $v_1, \ldots, v_j$ of the attribute atoms, the sufficient statistics correspond to a joint distribution over m Boolean relationship random variables:

$$P(R_1 = \cdot, R_2 = \cdot, \ldots, R_m = \cdot; f_1 = v_1, \ldots, f_j = v_j).$$

We refer to the Möbius parameters of this joint distribution as the **Möbius parameters for the attribute values** $v_1, \ldots, v_j$.

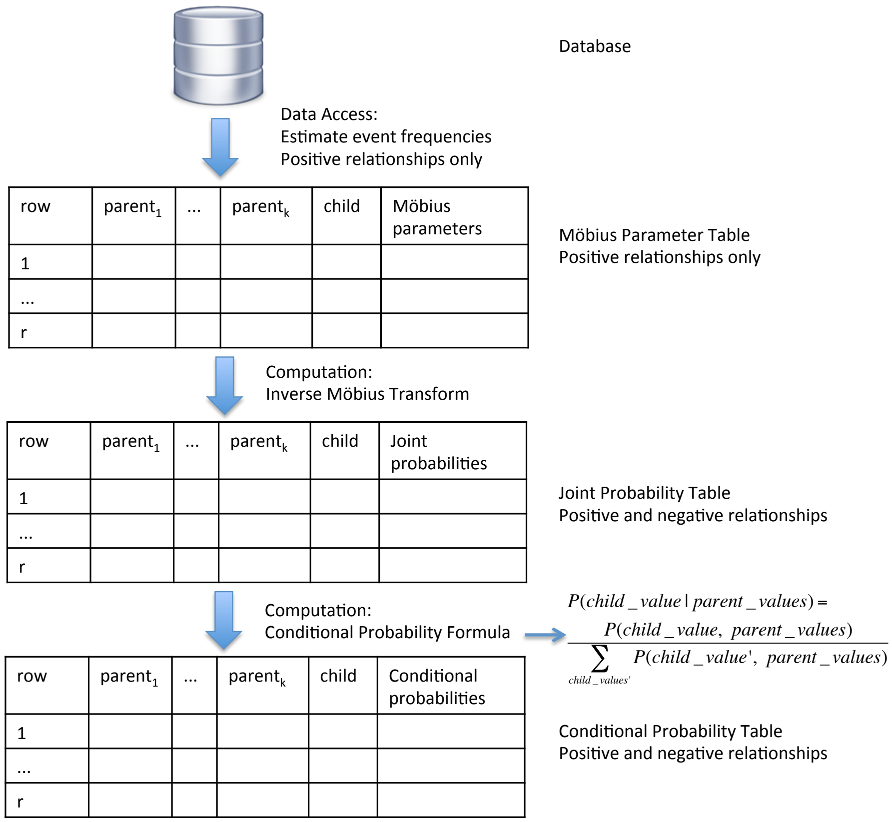
Example. For the Bayes net of Figure 3 (Top), fix the attribute condition $gender(X) = W$. The four Möbius parameters for this attribute condition are

$P(gender(X) = W)$

$P(Friend(X, Y) = T; gender(X) = W)$

$P(Follows(X, Y) = T; gender(X) = W)$

$P(Friend(X, Y) = T, Follows(X, Y) = T; gender(X) = W)$



$$P(child\_value \mid parent\_values) = \frac{P(child\_value,\ parent\_values)}{\sum_{child\_values'} P(child\_value',\ parent\_values)}$$

Fig. 4. Computation of joint probabilities in a relational database. (1) Estimate Möbius parameters using standard table join operations. (2) Transform the Möbius parameters into joint probabilities. (3) Compute conditional probabilities from joint probabilities. Only the first step involves data access.

5.2 The Inverse Möbius Transform

It is clear that the Möbius parameters can be computed from the joint probabilities by marginalization. For example, for two binary random variables, we have $P(B_1 = 1) = P(B_1 = 1, B_2 = 1) + P(B_1 = 1, B_2 = 0)$. Conversely, the Möbius inversion lemma entails that *the joint probabilities can be computed from the Möbius parameters.*[2] The Inverse Möbius Transform is an optimal algorithm for carrying out this computation, using the local update operation

$$P(R = F, \mathbf{R}) = P(\mathbf{R}) - P(R = T, \mathbf{R}) \tag{4}$$

[2] There are several other ways to parametrize binary joint distributions, such as canonical parametrizations [18, Sec.4.4.2.1]. See also Buchman *et al.* [19].

where **R** is a conjunction of relationship specifications, possibly with both positive and negative relationships. The equation holds for any fixed set of attribute conditions $f_1 = v_1, \ldots, f_j = v_j$. Eq. 4 generalizes the 1-minus trick: the joint probabilities on the right hand side each involve exactly one less false relationship than the joint probability on the left.

As illustrated in Figure 4, our implementation of the transform uses two data structures. *Middle table:* a joint probability table with r rows and $k+2$ columns. The first $k = j + m$ columns are for the parent nodes, column $k + 1$ is for the child node, and column $k + 2$ is the joint probability for the row. The r rows enumerate all possible combinations of the child and parent node value. Thus the joint probability table is just like a conditional probability table, but with joint probabilities instead of conditional ones. *Top table:* the **Möbius table** is just like the joint probability table, but with entries T and * instead of T and F for the Boolean relationship values. The value * represents "unspecified". The entries in the Möbius table do not involve negated relationships—all relationship nodes have the value T or $*$.

The IMT initializes the Möbius parameter values with frequency estimates from the data (top table). It then goes through the relationship nodes $R_1, \ldots, R_m$ in order, at stage i replacing all occurrences of $R_i = *$ with $R_i = F$, and applying the local update equation to obtain the probability value for the modified row. At termination, all $*$ values have been replaced by F and the table specifies all joint frequencies (middle table). In the final step, with complexity linear in r, we compute conditional probabilities (bottom table).

This algorithm is repeated for all possible assignments of values to attribute node, for every child node. Algorithm 1 gives pseudo code and Figure 3 presents an example of the transform step.

5.3 Complexity Analysis.

Kennes and Smets [9] analyze the complexity of the IMT. We summarize the main points. Recall that m is the number of binary relationship nodes in a single node family.

1. The IMT only accesses *existing* links, never nonexisting links.
2. The inner loop of Algorithm 1 (lines 3–6) performs $O(m2^{m-1})$ updates [9]. This is optimal [9, Cor.1].
3. If the joint probability table for a family contains r rows, the algorithm performs $O(m \times r)$ updates in total.

In practice, the number m is small, often 4 or less.[3] We may therefore treat the number of updates as proportional to the number r of Bayes net parameters.Our simulations confirm that the cost of the IMT updates is dominated by the data access cost for finding the Möbius parameter frequencies.

[3] For general m, the problem of computing a sufficient statistic in a relational structure—a joint probability of the form (3)—is #P-complete [20, Prop.12.4].

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

Input: database $\mathcal{D}$; a set of nodes divided into attribute nodes $f_1, \ldots, f_j$ and relationship nodes $R_1, \ldots, R_m$.
Output: joint probability table specifying the data frequencies for each joint assignment to the input nodes.

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

6 Evaluation

We evaluated the learning times (Section 6.2) and quality of inference (Section 6.3) of our approach to class-level semantics, as well as comparing the inference accuracy of our methods to Markov Logic Networks. Another set of simulations evaluates the accuracy of parameter estimates (conditional probabilities). All experiments were done on a QUAD CPU Q6700 with a 2.66GHz CPU and 8GB of RAM. The datasets and code are available on the Web [21].

6.1 Datasets

We used four benchmark real-world databases, with the modifications described by Schulte and Khosravi [13]. See that article for details.

Mondial Database. A geography database, featuring one self-relationship, *Borders*, that indicates which countries border each other. The data are organized in 4 tables (2 entity tables, 2 relationship tables, 10 descriptive attributes).

Hepatitis Database. A modified version of the PKDD'02 Discovery Challenge database. The data are organized in 7 tables (4 entity tables, 3 relationship tables and 16 descriptive attributes).

Financial A dataset from the PKDD 1999 cup. The data are organized in 4 tables (2 entity tables, 2 relationship tables, 13 descriptive attributes).

MovieLens. A dataset from the UC Irvine machine learning repository. The data are organized in 3 tables (2 entity tables, 1 relationship table, and 7 descriptive attributes).

To obtain a Bayes net structure for each dataset, we applied the learn-and-join algorithm [13] to each database with the implementation provided by the authors. This is the state-of-the-art structure learning algorithm for FBNs; for an objective function, it uses the pseudo-likelihood described in Section 4.

6.2 Learning Times

Table 2 shows the run times for computing parameter values. The Complement method uses SQL queries that explicitly construct tables for the complement of relationships (tables representing tuples of *unrelated* entities), while the IMT method uses the inverse Möbius transform to compute the conditional probabilities. The IMT is faster by factors of 15–237.

Table 2. Learning times (sec) for algorithms that construct explicit complement tables or use the inverse Möbius transform.

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

6.3 Inference

The evaluation of parameter learning for Bayes nets is based on query accuracy of the resulting nets [23] because answering probabilistic queries is the basic inference task for Bayes nets. Correct parameter values will support correct query results, so long as the given Bayes net structure matches the true distribution (i.e., is an I-map). We evaluate our algorithm by formulating random queries and comparing the results returned by our algorithm to the ground truth, the actual database frequencies.

We randomly generate queries for each dataset according to the following procedure. First, randomly 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 randomly the number k of conditioning variables, ranging from 1 to 3. Make a random selection of k variables $V_1, \ldots, V_k$ and corresponding values $a_1, \ldots, a_k$. The query to be answered is then $P(V = a|V_1 = a_1, \ldots, V_k = a_k)$.

As done by Getoor et al. [2], we evaluate queries after learning parameter values on the entire database. 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 [22]. Figure 5(left) shows the query performance for each database. A point (x, y) on a curve indicates 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 difference of less than 1% (inset numbers).

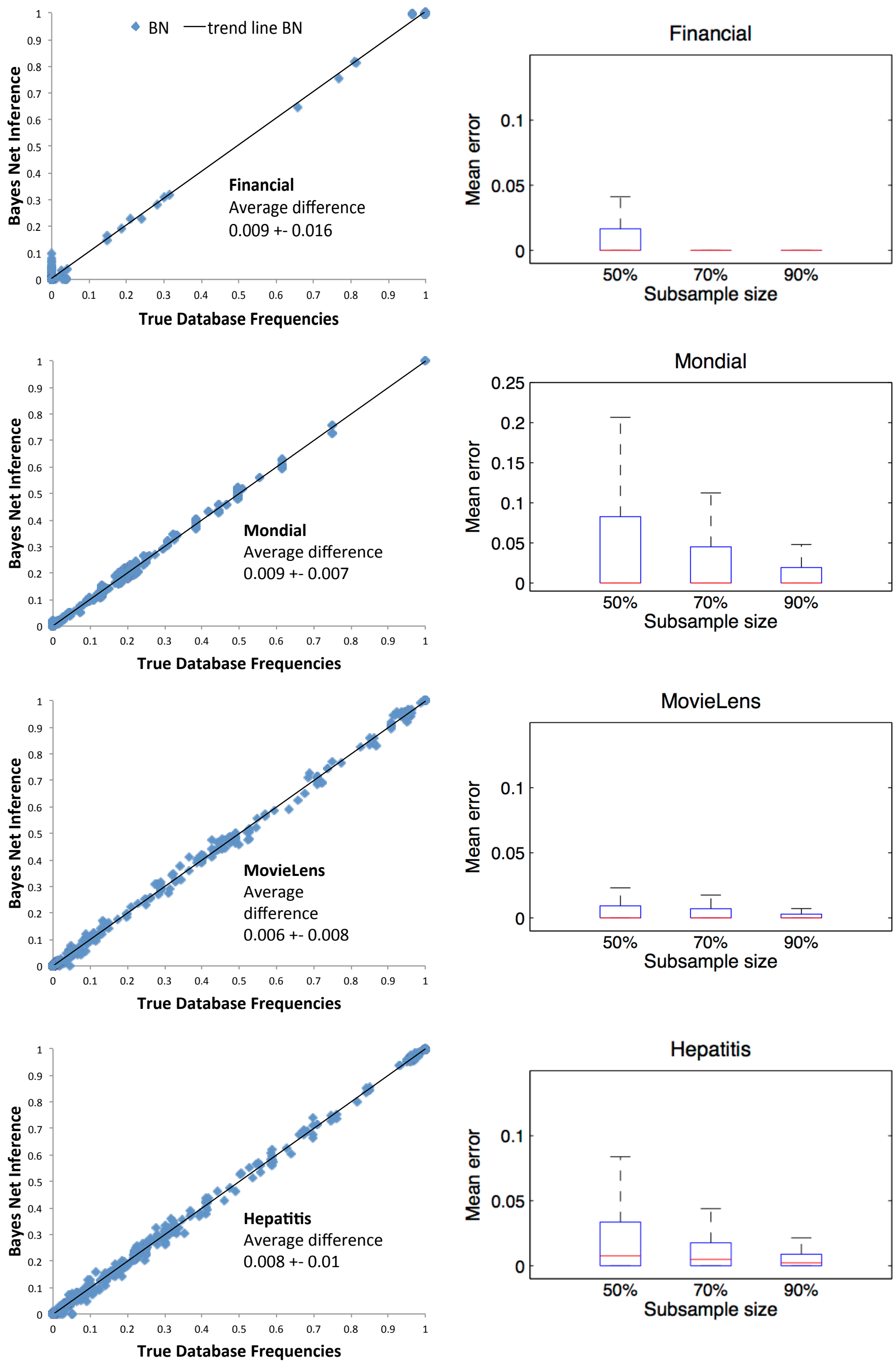


Fig. 5. *Left*: Query performance (Section 6.3): Estimated vs. true probability, with average error and standard deviation. Number of queries/average inference time per query: Mondial, 506/0.08sec; MovieLens, 546/0.05sec; Hepatitis, 489/0.1sec; Financial, 140/0.02sec. *Right:* Error (Section 6.4): Absolute difference in estimated conditional probability parameters, averaged over 10 random subdatabases and all BN parameters. Whiskers indicate the 95th percentile. The 70% and 90% subsamples of the Financial database had so little variance that the whiskers are not visible.

Comparison With Markov Logic Networks. To benchmark our results, we compare the average error of FBN inferences with frequency estimates from Markov Logic Network (MLN)s. MLNs are a good comparison point because (1) they are currently one of the most active areas of SRL research; (2) the Alchemy system provides an open-source, state-of-the-art implementation for learning and inference [24]; (3) they do not require the specification of further components (e.g., a combining rule or aggregation function); and (4) their undirected model can accommodate cyclic dependencies.

Like most statistical-relational models, MLNs were designed for instance-level probabilities rather than class-level queries. We therefore need an extra step to convert instance-level probabilities into class-level probabilities. To derive class-level probability estimates from instance-level inference, the following method due to Halpern [3, fn.3] can be used: Introduce a new individual constant for each 1st-order variable in the relational model (e.g., random-student, random-course, and random-prof). Applying instance probability inference to these new individuals provides a query answer that can be interpreted as a class-level probability. To illustrate, if the only thing we know about Tweety is that Tweety is a bird, then the probability that Tweety flies should be the frequency of flyers in the bird population [25]. That is, the class-level probability $P(Flies(B) = T)$ can be obtained from the instance-level marginal probability $P(Flies(tweety) = T)$ where *tweety* is a new constant not featured in the data used for learning. In the experiments below, we apply MLN inference to new constants as described to obtain class-level probability estimates. We compare these learning algorithms [26]:

BN: Bayes net parametrized with maximum pseudo likelihood estimates.
LHL: A structure learning algorithm that produces a parametrized MLN, with Alchemy's MC-SAT for inference.
LSM: Another structure learning algorithm that produces a parametrized MLN, with Alchemy's MC-SAT for inference. In experiments by Kok and Domingos, LSM outperformed previous MLN learners.

As shown in Table 3, the Bayes net models provide an order of magnitude more accurate class-level estimates than the MLN models.

Table 3. Class-level query performance of Bayes nets vs. two algorithms for learning Markov Logic Networks, as indicated by average absolute error between the predicted frequency and the true database frequencies. NT denotes non-termination within the system resources.

	Average error		
Dataset	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

6.4 Conditional Probabilities

In the previous inference results, the correct query answers are defined by the entire population, represented by the benchmark database, and the model is also trained on the entire database. To study parameter estimation at different sample sizes, we trained the model on $N\%$ of the data for varying N while continuing to define the correct value in terms of the entire population. Conceptually, we treated each benchmark database as specifying an entire population, and considered 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 [27, 28], 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. 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.

The right side of Figure 5 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 quartile of estimation error is bound within 10% on Mondial, the worst case, and within less than 5% on the other datasets.

7 Probability Logic

First-order logic was extended to incorporate statements about probabilities in Halpern and Bacchus's classic AI papers [3, 4]. Formalizing probability within logic provides a formal semantics for probability assertions and allows principles for probabilistic reasoning to be defined as logical axioms. This section provides a brief outline of probability logic. We describe how the random selection concept has been used to define a truth-conditional semantics for probability formulas [3, 4]. This semantics is equivalent to our class-level interpretation of functor random variables.

7.1 Syntax

We adopt Chiang and Poole's version of predicate logic for statistical-relational modelling [11]. We review only the concepts that are the focus of this paper.

Individuals are represented by **constants**, which are written in lower case (e.g., *bob*). A **type** is associated with each individual, for instance *bob* is a *person*. The **population** associated with type τ, a set of individuals, is denoted by $\mathcal{P}(\tau)$.

We assume that the populations are specified as part of the type description. A **logical variable** is capitalized and associated with a type. For example, the logical variable *Person* is associated with the population *person*. We use the definition of functor given in Section 2.1; we assume that the language contains functor symbols such as f, f', g. A **literal** is of the form $f(\sigma_1, \ldots, \sigma_a) = v$, where v is in the range of the functor, and each σ_i is either a constant or a population variable. The types of $\sigma_1, \ldots, \sigma_a$ must match the argument types of f. If all the $\sigma_1, \ldots, \sigma_a$ are constants, then $f(\sigma_1, \ldots, \sigma_a) = v$ is a **ground literal**. A ground literal is the basic unit of information. Literals can be combined to generate *formulas*. In this paper we consider only formulas that are **conjunctions** of literals, denoted by the Prolog-style comma notation $f(\sigma_1, \ldots, \sigma_a) = v, \ldots, f'(\sigma'_1, \ldots, \sigma'_{a'}) = v'$. The work of Halpern and Bacchus shows how to extend the random selection semantics to the full syntax of first-order logic.

A **probability literal** is of the form

$$P(\phi) = p$$

where each p is a term that denotes a real number in the interval [0,1] and ϕ_i is a conjunctive formula as defined above. A probability formula is a conjunction of probability literals. We refer to predicate logic extended with probability formulas as *probability logic.*

Let $\{X_1, \ldots, X_k\}$ be a set of logical variables of types $\tau_1, \ldots, \tau_k$. A **grounding** γ is a set $\gamma = \{X_1 \backslash x_1, \ldots, X_k \backslash x_k\}$ where each x_i is a constant of the same type as X_i. A grounding γ for a formula ϕ is a grounding for the variables that appear in the formula, denoted by $\phi\gamma$.

Examples. Consider again the social network model of Figure 1 (a), with a single type, $\tau = \mathit{person}$, associated with two logical variables X and Y. A conjunction of nonground literals is $\mathit{gender}(X) = W$, $\mathit{gender}(Y) = M$, $\mathit{Friend}(X, Y) = T$. Applying the grounding $\{X \backslash \mathit{anna}, Y \backslash \mathit{bob}\}$ produces the conjunction of ground literals $\mathit{gender}(\mathit{anna}) = W$, $\mathit{gender}(\mathit{bob}) = M$,$\mathit{Friend}(\mathit{anna}, \mathit{bob}) = T$.

An example of a nonground probability formula is

$$P(\mathit{gender}(X) = W, \mathit{gender}(Y) = M, \mathit{Friend}(X, Y) = T) = 1/4,$$

while an example of a ground probability formula is

$$P(\mathit{gender}(\mathit{anna}) = W, \mathit{gender}(\mathit{bob}) = M, \mathit{Friend}(\mathit{anna}, \mathit{bob}) = T) = 1/4.$$

7.2 Semantics.

Halpern defines different semantics for probability formulas depending on whether they contain only nonground probability literals (type 1), only ground probability literals (type 2), or both (type 3). Relational statistics are represented by nonground probability formulas, so we give the formal definition for the type 1 semantics only. The type 2 semantics for probability formulas with ground literals is is well-known in the statistical-relational learning community [29, 30],[12, Ch.14.6.1]. An example of a mixed type 3 case is the connection between the class-level probabilities and individual marginal probabilities that we utilized in the Markov Logic Network experiments reported in Table 3. Schulte [25] discusses the importance of type 3 semantics for statistical-relational learning.

A **relational structure** $\mathcal{S}$ specifies (1) a population for each type, and (2) for each functor f a set of tuples $\mathcal{S}_f = \{\langle x_1, \ldots, x_a, v\rangle\}$, where the tuple $\langle x_1, \ldots, x_a\rangle$ match the functor's argument types, and the value v is in the range of f. Each tuple $\langle x_1, \ldots, x_a\rangle$ appears exactly once in $\mathcal{S}_f$.

A relational structure defines truth conditions for ground literals, conjunctive formulas, and probability formulas as follows. 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

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

A conjunction is true if all its conjuncts are true. All ground literals listed above are true in the structure of Figure 1 (b). False literals include $\mathit{CoffeeDrinker}(\mathit{bob}) = T$ and $\mathit{Friend}(\mathit{anna}, \mathit{anna}) = T$.

A relational structure does not determine a truth value for a nonground literal such as $\mathit{gender}(X) = M$, since a gender is not determined for a generic person X. We follow database theory and view a nonground formula as a *logical query* [31]. 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 S as $\mathbb{R}_{\mathcal{S}}(\phi)$, defined by

$$\mathbb{R}_{\mathcal{S}}(\phi) \equiv \{\langle x_1, \ldots, x_k\rangle : \mathcal{S} \models \phi\{X_1 \backslash x_1, \ldots, X_k \backslash x_k\}\rangle$$

To define truth conditions for nonground probability formulas, a relational structure is augmented with a distribution P_τ over each population $\mathcal{P}(\tau)$. There is no constraint on the distribution. Halpern refers to such augmented structures as **type 1 relational structures**; we continue to use the symbol S for type 1 structures. Given a type 1 structure, we may view a logical 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. The random selection semantics states that the probability of a formula in a type 1 structure is the probability of the set of groundings that satisfy the formula. In other words, it is the probability that the formula is satisfied by a randomly selected grounding. The formal definitions are as follows.

1. $P_{\mathcal{S}}(X_1 = x_1, \ldots, X_a = x_a) =_{df} P_{\tau_1}(x_1) \times \cdots \times P_{\tau_a}(x_a)$ where τ_i is the type of variable X_i and constant x_i.
2. $\mathcal{S} \models P(\phi) = p$ if and only if $p = \sum_{\langle x_1, \ldots, x_a\rangle \in \mathbb{R}_{\mathcal{S}}(\phi)} P_{\mathcal{S}}(x_1, \ldots, x_a)$.

Each joint probability specified by a functor Bayes net is a conjunction in probability logic (like Equation 1). The net therefore specifies a set of nonground probability formulas. The logical meaning of these formulas according to the type 1 semantics is the same as the meaning of class-level probabilities according to the random selection semantics of Section 3. In the terminology of Bacchus, Grove, Koller, and Halpern, a set of nonground probability formulas is a *statistical knowledge base* [32]. 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.

8 Related Work and Discussion

Statistical Relational Models. To our knowledge, the Statistical Relational Models (SRMs) of 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, but SRMs have compared favorably with benchmark methods for estimating the cardinality of a database query result [2] (Sec. 1).

SRMs differ from FBNs and other statistical-relational models in several respects. (1) SRMs are 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: The queries that can be formulated using the nodes in an SRM cannot express general combinations of positive and negative relationships [8, Def.6.6]. 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 (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 to include drawing random tuples from complement tables, but this entails the large increase in storage and processing described above.

Class-level vs. Instance-level Probabilities. Most previous work on statistical-relational learning has been concerned with instance-level probabilities for predicting the attributes and relationships of *individuals* [33, 12]. Examples of instance-level queries include the following.

- Given that Tweety is a bird, what is the probability that Tweety flies?
- Given that Sam and Hilary are friends, and given the genders of all their other friends, what is the probability that Sam and Hilary are both women?
- What is the probability that Jack is highly intelligent given his grades?

Probability logic represents such instance-level probabilities by formulas with ground literals (e.g., $P(Flies(tweety)) = p$). In graphical models, instance level probabilities are defined by a *template semantics* [33]: the functors associated with every node are grounded with every appropriate constant. Instance-level probabilities can be computed by applying inference to the resulting graph.

A key difference between class- and instance-level queries is that instance-level queries can involve an unbounded number of random variables. For instance, if m has 100 Facebook friends $x_1, \ldots, x_{100}$, an instance-level query to predict the gender of m given the genders of her friends would involve 200 ground literals of the form

$$P(\mathit{gender}(m)|\mathit{gender}(x_1), \mathit{Friend}(m, x_1), \ldots, \mathit{gender}(x_{100}), \mathit{Friend}(m, x_{100})).$$

In contrast, class-level queries are restricted to the set of random variables in the class-level model, which is relatively small. For instance, the Bayes net of Figure 1 (b) contains four class-level random variables. This allows a query like

$$P(\mathit{gender}(X)|\mathit{gender}(Y), \mathit{Friend}(X, Y)),$$

which predicts the gender of a single generic user, given the gender of a single generic friend.

While we do not treat learning models for instance-level probabilities in this paper, we note that the Bayes net structures that we use in this paper also perform well for instance-level inference [13, 28]. The structures are learned using the random selection pseudo likelihood as the main component of the objective function. These evaluations suggest the pseudo likelihood guides learning towards models that perform well for *both* class-level and instance-level inferences.

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]. For instance, the inference procedure would count the number of male friends that m has, rather than repeat the same computation for each male friend. Based on the similarity with computing event counts in a database, Schulte and Khosravi refer to learning with the pseudo-likelihood as *lifted learning* [13]. 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 compute instance-level predictions (e.g., predict the gender of m). 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.

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" [12, 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*. For instance, we have made the complete-data assumption that if a link between two individuals is not listed in a database, it does not exist. This entails 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 [34, 35]. Another approach is downsampling existing links by adding random negative links [15, 36]. Many methods for dealing with missing data employ complete-data learning as a subroutine (e.g., the Expectation Maximization algorithm; cf. [14, Ch.4.3]). Given the speed and accuracy of pseudo-likelihood Bayes net learning in the complete-data case, we expect that it will be useful for learning with missing data as well.

Another 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.

9 Conclusion

Class-level probabilities represent relational statistics about the frequencies or rates of generic events and patterns. Representing these probabilities has been a long-standing concern of AI research. Class-level probabilities represent informative statistical dependencies in a relational structure, supporting such applications as strategic planning and query optimization. The class-level interpretation of Bayes nets is based on the classic random selection semantics for probability logic. Parameters are learned using the empirical frequencies, which are the maxima of a pseudo-likelihood function. The inverse Möbius transform makes the computation of database frequencies feasible even when the frequencies involve 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 Bayes net models derived with maximum pseudo-likelihood parameter estimates provide excellent estimates of class-level probabilities.

Acknowledgements

This research was supported by an NSERC Discovery Grant to Schulte. Lise Getoor's work on Statistical Relational Models inspired us to consider class-level modelling with Parametrized Bayes nets; we thank her for helpful comments and encouragement. Preliminary versions of this paper were presented at the SRL workshop at ICML 2012 and the ILP 2012 conference. We thank the organizers for providing a discussion venue, and the referees for constructive criticism.

References

[1] Schulte, O.: A tractable pseudo-likelihood function for Bayes nets applied to relational data. In: SIAM SDM. (2011) 462–473

[2] Getoor, L., Taskar, B., Koller, D.: Selectivity estimation using probabilistic models. ACM SIGMOD Record **30**(2) (2001) 461–472
[3] Halpern, J.Y.: An analysis of first-order logics of probability. Artificial Intelligence **46**(3) (1990) 311–350
[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., Oktay, H., Jensen, D.: Learning causal models of relational domains. In: AAAI. (2010)
[6] 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
[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. [37] chapter 5 129–173
[11] Chiang, M., Poole, D.: Reference classes and relational learning. Int. J. Approx. Reasoning **53**(3) (2012) 326–346
[12] Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Prentice Hall (2010)
[13] Schulte, O., Khosravi, H.: Learning graphical models for relational data via lattice search. Machine Learning **88**(3) (2012) 331–368
[14] Domingos, P., Lowd, D.: Markov Logic: An Interface Layer for Artificial Intelligence. Morgan and Claypool Publishers (2009)
[15] Khot, T., Natarajan, S., Kersting, K., Shavlik, J.W.: Learning Markov logic networks via functional gradient boosting. In: ICDM, IEEE Computer Society (2011) 320–329
[16] Yin, X., Han, J., Yang, J., Yu, P.S.: Crossmine: Efficient classification across multiple database relations. In: ICDE, IEEE Computer Society (2004) 399–410
[17] Drton, M., Richardson, T.S.: Binary models for marginal independence. Journal of the Royal Statistical Society: Series B (Statistical Methodology) **70**(2) (2008) 287–309
[18] Koller, D., Nir: Probabilistic Graphical Models. MIT Press (2009)
[19] Buchman, D., Schmidt, M.W., Mohamed, S., Poole, D., de Freitas=, N.: On sparse, spectral and other parameterizations of binary probabilistic models. Journal of Machine Learning Research - Proceedings Track **22** (2012) 173–181
[20] Domingos, P., Richardson, M.: Markov logic: A unifying framework for statistical relational learning. [37]
[21] Khosravi, H., Man, T., Hu, J., Gao, E., Schulte, O.: Learn and join algorithm code. URL = `http://www.cs.sfu.ca/~oschulte/jbn/`.
[22] The Tetrad Group: The Tetrad project (2008) http://www.phil.cmu.edu/projects/tetrad/.
[23] 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
[24] 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.

[25] Schulte, O.: Challenge paper: Marginal probabilities for instances and classes. ICML-SRL Workshop on Statistical Relational Learning. (June 2012)
[26] Kok, S., Domingos, P.: Learning Markov logic networks using structural motifs. In: ICML. (2010) 551–558
[27] Frank, O.: Estimation of graph totals. Scandinavian Journal of Statistics **4:2** (1977) 81–89
[28] 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
[29] Cussens, J.: Logic-based formalisms for statistical relational learning. [37]
[30] 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
[31] Ullman, J.D.: Principles of Database S¡ystems. 2 edn. W. H. Freeman & Co. (1982)
[32] Bacchus, F., Grove, A.J., Koller, D., Halpern, J.Y.: From statistics to beliefs. In: AAAI. (1992) 602–608
[33] Getoor, L., Taskar, B.: Introduction. [37] 1–8
[34] Hoff, P.D.: Multiplicative latent factor models for description and prediction of social networks. Computational and Mathematical Organization Theory (2007)
[35] 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
[36] 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
[37] Getoor, L., Taskar, B.: Introduction to Statistical Relational Learning. MIT Press (2007)