JMLR: Workshop and Conference Proceedings 13: 315-330
2nd Asian Conference on Machine Learning (ACML2010), Tokyo, Japan, Nov. 8–10, 2010.

An Approach to Inference in Probabilistic Relational Models using Block Sampling

Fabian Kaelin FABIAN.KAELIN@CS.MCGILL.CA
School of Computer Science
McGill University

Doina Precup DPRECUP@CS.MCGILL.CA
School of Computer Science
McGill University

Editor: Masashi Sugiyama and Qiang Yang

Abstract

We tackle the problem of approximate inference in Probabilistic Relational Models (PRMs) and propose the *Lazy Aggregation Block Gibbs* (LABG) algorithm. The LABG algorithm makes use of the inherent relational structure of the ground Bayesian network corresponding to a PRM. We evaluate our approach on artificial and real data, and show that it scales well with the size of the data set.

Keywords: Probabilistic Relational Models, approximate inference, Gibbs sampling, aggregation

1. Introduction

Researchers designing graphical models often make the assumption that their data are independent and identically distributed. In reality this is rarely the case - usually the data has an inherent structure. The central idea of statistical relational learning is to exploit this structure when learning models to represent the data. Numerous approaches have been introduced in recent years - for example Probabilistic Relational Models (PRMs), Relational Markov networks (RMNs) and Markov Logic; (see the textbooks of Getoor & Taskar, 2007 and Koller & Friedman, 2009 for details).

Probabilistic Relational Models (Getoor, 2000) define a language that can be used to describe the relationships - structural and probabilistic - between classes and variables, and thus allows representing dependencies between sets of objects. Heckerman et al. (2004) introduced the directed acyclic probabilistic entity-relationship (DAPER) model which is based on the entity-relationship model used to design relational databases. Our work builds on the DAPER framework; more precisely, we present a framework for PRM specification which provides easy support for aggregation and an efficient approximate inference algorithm.

In general, the syntax of relational models resembles first-order logic (FOL) models for which inference is undecidable; hence, it is understandable that inference in relational models is a difficult problem. Undirected models such as Markov Logic have the advantage of being more expressive than directed models like PRMs; the drawback is increased complexity during inference. Markov Logic extends standard FOL formulae with uncertainty, current inference approaches include LazySAT for MAP estimates and MC-SAT for conditional posterior distributions. Both methods are based on the satisfiability (SAT) problem, where the 'lazy' aspect is exploiting the natural

sparsity of relational domains. Recent results are very promising, but to our knowledge it is unclear how well this approach scales with the dataset size. In directed models such as PRMs, much of the existing work involves constructing a *ground Bayesian network* (GBN) and performing inference in this model. This is similar to translating first-order formulae into propositional ones; hence, it is easy to see that the network generated can be very large, making inference very expensive. Aggregation (Zhang and Poole, 1996) is an approach to this problem based on the idea that the value of the child node can be computed as a function of the values of the parent nodes, whose computation does not require knowing the number of parents in advance. Recent work (e.g. Milch et al., 2008, Kisynski & Poole, 2009) has proposed algorithms for exact inference in lifted (i.e. non-grounded) models using aggregation. In this case, aggregation is used to avoid creating the ground Bayes net for as long as possible. Structured variable elimination introduced by Pfeffer and Koller (2000) is extension of the original variable elimination method (Zhang and Poole, 1996) to the relational domain. However, exact inference computationally expensive, and in the Bayesian networks community, it is often replaced by approximate inference methods, which can handle large models much better. In this paper, we propose an approach for approximate inference in PRMs based on MCMC, which leverages the relational structure of the model for aggregation and sampling order. In our framework, the ground Bayes net is constructed recursively and incrementally, only to the extent that it is needed to answer the query at hand. Aggregation of parent nodes is used to keep the parametrization of the network as small as possible. We propose a block Gibbs sampling algorithm that can answer the desired query; this algorithm takes full advantage of the special structure of the ground Bayes net, in order to sample as many variables in parallel as possible. We present scaling experiments in an artificial data set, showing that the running time of the proposed approach scales very well with the size of the ground Bayesian network. We also present results of an analysis of political campaign contributions data in the United States to showcase the ability of the proposed framework to model real data sets.

The paper is structured as follows. In Sec. 2 we review PRMs and introduce the framework that we developed to specify such models. Sec. 3 describes the Lazy Aggregation Block Gibbs (LABG) algorithm that we propose. In Sec. 4 we provide an empirical evaluation, and Sec. 5 contains conclusions and future work.

2. Probabilistic Relational Models

2.1 Entity-Relationship Model

In database design, the entity-relationship model is a representation of the structure of a data set, and the DAPER framework (which is the basis of our work) is very related to this model. It consists of a set of entities $E \in \mathcal{E}$, where E represents a class of objects, a set of relationships $R \in \mathcal{R}$, where R is linking a set of entity classes $R(E_1, ..., E_n)$ and a set of attributes $A \in \mathcal{A}$, where each attribute A is associated with a class $X \in \mathcal{X} \equiv \mathcal{E} \cup \mathcal{R}$. We denote $\mathcal{A}(X)$ as the set of attributes associated with X. A DAPER model can be represented graphically by using rectangles to represent entities, diamonds to represent relations and ellipses to represent attributes; we use dashed lines to connect entities to relationships and attributes. An example (which we discuss in detail later) is shown in Fig. 1.

Every entity $E \in \mathcal{E}$ and every relationship $R \in \mathcal{R}$ contain a set of objects that are instantiations of that specific class; we refer to these sets as entity sets and relationships sets respectively. We denote by $\sigma_{\mathcal{ER}}$ the set that contains all objects in our data set (also called the *skeleton*), by $\sigma_{\mathcal{E}}$

and $\sigma_{\mathcal{R}}$ the sets of all entity objects and all relation objects, and by $\sigma_{\mathcal{ER}}(X)$ the set of objects in $X \in \mathcal{X}$. Every object $x \in \sigma_{\mathcal{ER}}(X)$ is associated with a set of attributes $\mathcal{A}(x)$. We use $x.A$ to denote an attribute of object x and $X.A$ to denote an attribute class associated with class X. Each $x.A \in \mathcal{A}(\sigma_{\mathcal{ER}})$ has a domain $\mathcal{V}(A)$ of possible values. Finally, an instance of an entity-relationship model, $\mathcal{I}_{\mathcal{ERA}}$, consists of a skeleton $\sigma_{\mathcal{ER}}$ where for each $x \in \sigma_{\mathcal{ER}}$ and $A \in \mathcal{A}(x)$, $x.A$ is assigned a valid value in $\mathcal{V}(A)$.

Making the connection to a relational database, a table corresponds to an entity or relationship class $X \in \mathcal{E} \cup \mathcal{R}$ and the rows of the table to the skeleton objects $\sigma_{\mathcal{ER}}(X)$. The columns of the table are the set of attributes $X.A \in \mathcal{A}(X)$ where the entry in row x and column A is denoted $x.A$. The relational structure is defined by the foreign keys of the relationship class tables, which are the set of linked entities $R(E_1, ..., E_n)$.

Example 1 *In the political system of the United States, money is an important factor. Recipients of political contributions are required by law to report details about each donation they receive. The domain is suitable for modelling using PRMs. Fig. 1 contains an example model. The set of entities is* $\mathcal{E} = \{Recipient, Contributor, State\}$ *and the set of relationships* $\mathcal{R}$ *contains the relationship* $Donation$*. The linked entities of* $Donation$ *are* $R(Recipient, Contributor, State)$ *and every donation object* $d \in \sigma_{\mathcal{R}}(Donation)$ *has a attribute object* $d.Amount$*.*

2.2 Probabilistic Structure

Probabilistic dependencies among the attributes $\mathcal{A}(\mathcal{X})$ are specified by solid arrows. Every attribute $X.A$ is associated with a set of *parents* denoted $\mathbf{pa}(X.A)$. Additionally, we may add a constraint $\mathcal{C}_{AB}$ on every dependency. The notion of a constraint is a generalization of what Friedman et al. (1999) calls a *slot chain*, and can be seen as a first-order expression that defines a subset of $\sigma_{\mathcal{X}}(X.A) \times \sigma_{\mathcal{X}}(Y.B)$ for which the probabilistic dependency is active. Note that a constraint is more expressive than a *slot chain* as there can be multiple paths connecting two attributes; a slot chain is limited to one path.

Given an entity-relationship model with probabilistic dependencies and a skeleton $\sigma_{\mathcal{ER}}$, an *unrolling* process can be used to generate a standard "ground" Bayesian network. A node is created for every attribute instance of every object $x.A \in \mathcal{A}(\sigma_{\mathcal{ER}})$. Then an arc is added to every pair $x.A$ and $y.B$, $y.B \in \mathcal{A}(\sigma_{\mathcal{ER}})$, if $X.A \in \mathbf{pa}(Y.B)$ and if $x.A$ and $y.B$ satisfy the constraint $\mathcal{C}_{AB}$. The resulting directed acyclic graph (DAG) is called the *Ground Bayesian network* (GBN).

Next a local probability distribution is defined for each attribute $X.A \in \mathcal{A}(\mathcal{X})$. This local distribution class - $P(X.A \mid \mathbf{pa}(X.A))$ - is shared by every object of that attribute class $x.A \in \mathcal{A}(\sigma_{\mathcal{ER}}(X))$.

Example 2 *In the political contribution domain there is a probabilistic dependency between the* $a = Amount$ *attribute of the* $Donation$ *relationship and the* $cat = Category$ *attribute associated with the* $Contributor$ *entity. The constraint* $\mathcal{C}_{Amount,Category}$ *will activate the dependency only between* $Contributor$ *objects for which the* $Donation$ *object is intended (*$contributor[Category] = contributor[Amount]$*). Similar dependencies exist for* $dem_{rec} = Recipient.Democratic$*,* $dem_{state} = State.Democratic$ *and* $inc = State.Income$*. Thus the shared local distribution of the amount is* $P(a \mid cat, dem_{rec}, dem_{state}, inc)$*.*

Note that during the unrolling process, the attribute instances $x.A$ can have a varying number of parents. But since the local distribution is shared among all $X.A$, the notion of *aggregation* has to

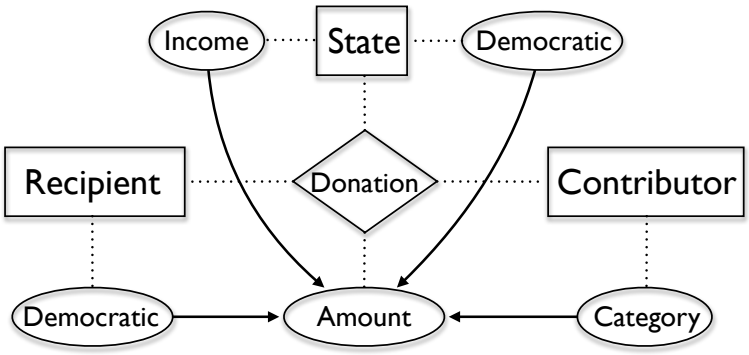


Figure 1: The DAPER model for the political contributions domain. The constraints $\mathcal{C}$ of the dependencies are not displayed for simplicity; see also example 2

be introduced. If the constraint $\mathcal{C}_{AB}$ of a dependency $X.A \leftarrow Y.B$ applied to $x.A$ returns a multiset $\{y.B\}$, then the function $\gamma : \{y.B\} \rightarrow y.B_\gamma$ returns a summary of the multiset. $\gamma(\{y.B\})$ can be any function that does not depend on the actual number of elements in the set, e.g. the average, mean, mode, min, max, logical "and" (for binary variables) etc.

A DAPER model $\mathcal{D}$ consists of an entity-relationship model with a probabilistic dependency structure and a set of local distributions (one for each $X.A$). The joint probability distribution in a DAPER model can be written as:

$$P(\mathcal{I}_{\mathcal{ERA}} \mid \mathcal{D}, \sigma_{\mathcal{ER}}) = \prod_{x.A \in \mathcal{A}(\sigma_{\mathcal{ER}})} P(x.A \mid \mathbf{pa}(x.A))$$
$$= \prod_{X \in \mathcal{E} \cup \mathcal{R}} \prod_{x \in \sigma_{\mathcal{ER}}(X)} \prod_{A \in \mathcal{A}(x)} P(x.A \mid \mathbf{pa}(x.A))$$

2.3 Parameter Estimation

In general, neither the local distributions nor the actual probabilistic dependencies are readily available. In the relational setting the uncertainty can also extend to the structure of the data itself, known as reference and identity uncertainty (Getoor and Taskar, 2007; Pasula and Russell, 2001), but for this paper we assume the full skeleton $\sigma_{\mathcal{ER}}$ is known, and the probabilistic dependencies are also specified by the designer of the model. Only the conditional probability distributions $P(X.A \mid \mathbf{pa}(X.A))$ for each attribute $X.A$ have to be learned from the data. Our software package use maximum likelihood estimation (MLE), and at the moment supports discrete variables. It is worth noting that the estimation is very easy to do using a relational data base, because the number of objects with a certain configuration can usually be computed with a single query.

3. Approximate Inference

Like in Bayes nets, probabilistic inference can be viewed as a process by which *influence* flows through the network. But instead of constraining that flow to be between the random variables of one instance, like in Bayesian networks, PRMs allow flow between interrelated objects as well. In

large domains, exact inference quickly becomes intractable and approximate inference is necessary. In this section we describe our inference method: we construct the ground Bayes net in a lazy manner, in order to capture just the variables needed to answer the query, then we use an efficient Gibbs sampling method for approximate inference.

3.1 Querying

Given a model $\mathcal{D}$, a query $\mathbb{Q} = (\mathbb{Y}, \mathbb{E})$ is defined by a set of event variables $\mathbb{Y} \subseteq \mathcal{A}(\sigma_{\mathcal{ER}})$ and a set of evidence variables $\mathbb{E} \subseteq \mathcal{A}(\sigma_{\mathcal{ER}})$. The set of all classes in the query is $\mathbb{Q}\langle\mathcal{X}\rangle = \mathbb{Y}\langle\mathcal{X}\rangle \cup \mathbb{E}\langle\mathcal{X}\rangle$ and the set of all objects of a class $X \in \mathbb{Q}\langle\mathcal{X}\rangle$ in the event and the evidence is denoted, respectively, by $\mathbb{Y}(X)$ and $\mathbb{E}(X)$. Finally, $\mathbb{Y}(X.A)$ designates the set $\{x.A\} \subseteq \mathbb{Y}(X)$ associated with the same attribute class $X.A$. The goal is to infer the posterior $P(\mathbb{Y} \mid \mathbb{E})$.

Example 3 *In the political contribution domain, we might be interested in predicting the political affiliation of a recipient based on the information about the donations, the contributors and the state information. The query $\mathbb{Q}$ would consist of $\mathbb{Y} = \{Recipient.Democratic_i\}$, where i references the $Recipient$ object of a specific politician and the evidence $\mathbb{E}$ would contain all information about the donations, the contributors and the states.*

3.2 Lazy Unrolling

A query can be answered in a Bayes net by taking into account only the subgraph that contains all event nodes and is d-separated from the full graph given the evidence nodes (see section 3.3). The d-separated Ground Bayesian network generated by the unrolling process for query $\mathbb{Q}$ should therefore satisfy

$$(GBN_{\text{d-sep}} \perp\!\!\!\perp GBN_{\text{full}}) \mid \mathbb{E},$$

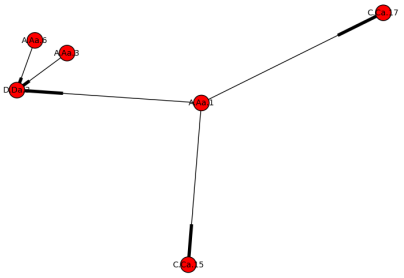
where GBN_{full} refers to the ground Bayes net induced by the full model. In the following we refer to $GBN_{\text{d-sep}}$ simply as Ground Bayes net. For traditional BNs there exist algorithms for this task - e.g. Bayes-Ball, proposed by Shachter (1998) and described in detail in (Koller and Friedman, 2009, page 75). When using PRMs, the structure of the GBN is stored in a first-order representation rather than explicitly, therefore a different approach is needed.

This observation allows the design of an recursive algorithm, similar to the First-order Bayes-ball algorithm (Meert et al., 2010), that constructs a partial Ground Bayesian network on an 'as needed' basis. Starting off with the set of event variables $\mathbb{Y}$, the parents and children are iteratively loaded subject to the rules that open/break a probabilistic path in the graph (Shachter, 1998). Eventually the evidence variables $\mathbb{E}$ should break all paths, at which point the partial GBN will be d-separated. As the probabilistic dependencies are usually between attributes of different classes, the structure of the resulting GBN depends on the relational model. Examples of resulting GBNs can be found in Fig. 2.

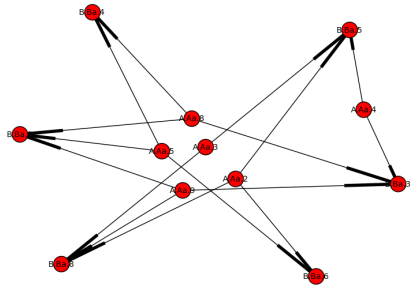
The GBN induced by a query $\mathbb{Q}$ will consist of the evidence nodes in $\mathbb{E}$ whose values are fixed to the observed values; all other nodes $\mathbb{S} = \mathcal{A}(\text{GBN})\backslash\{\mathbb{E}\}$ are not observed and therefore must be sampled during the inference procedure.

3.3 Block Gibbs Sampling

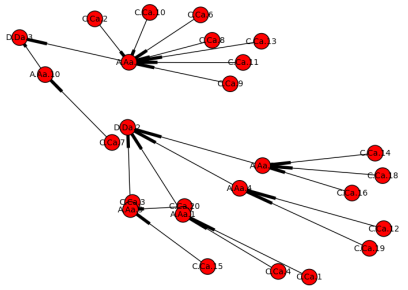
The underlying independence assumptions in Bayesian networks make Gibbs sampling a natural choice. It is well known that in a Bayesian network, samples from the proposal distribution for



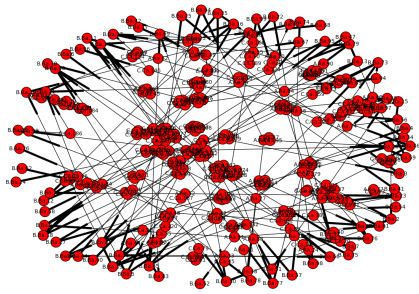
(a) A simple 1:n (A-C) and n:1 (A-D) relationship



(b) A m:n relationship



(c) Two linked relationships (C-A-D)



(d) A larger GBN based on a more complex query

Figure 2: The structure of the resulting of the ground Bayesian network for different probabilistic relationships

a variable X_i can be computed based only on the assignment of $\mathbf{x}_{(-i)}$ to the Markov blanket of X_i (Pearl, 1988); as a result, samples can be repeatedly drawn for all variables, generating in the end true samples from the desired conditional distribution. In the context of PRMs it is possible to leverage the structure of the GBN to reduce the computational complexity.

All sampling nodes in the GBN are associated with attribute objects $x.A_i \in \mathbb{S}$ and all $x.A_i \in \mathbb{S}(X.A)$ for some $X \in \mathbb{S}\langle\mathcal{X}\rangle$ share the same local distribution $P(X.A \mid \mathbf{pa}(X.A))$. Each $x.A_i$ can have a varying number of parents and children, therefore the full conditional distribution is $P_\phi(x.A_i \mid x.A_{(-i)})$, where $x.A_{(-i)}$ is an assignment to $\mathcal{A}(\sigma_{\mathcal{ER}})\backslash\{x.A_i\}$.

Let $\mathcal{C} = x.A_i \cup \text{Children}(x.A_i)$; then:

$$P_\phi(x.A_i \mid x.A_{(-i)}) = \frac{P(x.A_i, x.A_{(-i)})}{\sum_{x.A_i} P(x.A_i, x.A_{(-i)})} = \frac{\prod_{x.A \in \mathcal{A}(\sigma_{\mathcal{ER}})} P(x.A \mid \mathbf{pa}(x.A))}{\sum_{x.A_i} \prod_{x.A \in \mathcal{A}(\sigma_{\mathcal{ER}})} P(x.A \mid \mathbf{pa}(x.A))}$$

$$= \frac{\prod_{x.A \in \mathcal{C}} P(\cdot \mid \cdot) \prod_{x.A \notin \mathcal{C}} P(\cdot \mid \cdot)}{\sum_{x.A_i} \prod_{x.A \in \mathcal{C}} P(\cdot \mid \cdot) \prod_{x.A \notin \mathcal{C}} P(\cdot \mid \cdot)} \propto P(x.A_i \mid \mathbf{pa}(x.A_i)) \prod_{y.B \in \text{Children}(x.A_i)} P(y.B \mid \mathbf{pa}(y.B))$$

The contribution of each child $y.B$ is therefore the likelihood $L(x.A_i \mid y.B, \mathbf{pa}(y.B)\backslash\{x.A_i\})$. As $x.A_i$ is in $\mathbf{pa}(y.B)$ for each $y.B$, the influence is flowing 'upwards' without the need for aggregation. Furthermore, if $\mathbf{pa}(y.B)\backslash\{x.A_i\} \neq \emptyset$ (e.g. $y.B$ has other parents besides $x.A_i$) there will

also be influence flowing through the resulting V-structure. The other contribution to P_ϕ is the factor $P(x.A_i \mid \mathbf{pa}(x.A_i))$ which is influence flowing 'downwards', in aggregated form if necessary.

In general, if the number of parent attribute classes of any $x.A$ is smaller than the number of parent attribute objects, there is at least one parent attribute class for which aggregation has to be performed, because the shared local distribution constrains each parent to be single-valued. On the other hand, influence from child to parent is not aggregated since the above equation contains a product of the likelihoods $L(\cdot \mid \cdot)$ of all children nodes.

This observation allows for "lazy" computation of aggregations. Algorithm 1 presents our approach. The sampling nodes $\mathbb{S}$ are partitioned into blocks, where each block contains all attribute objects of the same attribute class $X.A$. Then an attribute class is randomly selected with probability proportional to the size of its block to ensure that each attribute object is equally likely to be sampled. After selecting a sampling attribute class, only attribute objects of that type will be sampled in that step. In the *LazyAggregation()* step we precompute all aggregation values of parent attributes for which two attribute objects $x.A_i, x.A_j \in \mathbb{S}(X.A)$ are conditionally independent. This is the case for all parent attributes $\mathbf{pa}(X.A)$ since $(x.A_i \perp\!\!\!\perp x.A_j) \mid \mathbf{pa}(X.A)$ as well as for the parents $\mathbf{pa}(Y.B)\backslash X.A$ of the children attribute objects $y.B$ except for $X.A$ itself. In this case, because $x.A_i$ and $x.A_j$ would not be mutually independent given a common child attribute object, the aggregation is computed in the *Aggregation()* step.

Algorithm 1 generates a Gibbs trajectory guaranteed to converge to $P(\mathbb{S} \mid \mathbb{E})$ if the PRM model satisfies the standard constraints defined by Getoor (2000). The desired marginal posterior $P(\mathbb{Y} \mid \mathbb{E})$ can be found by marginalizing the latent variables $\mathbb{S}\backslash\{\mathbb{Y}\}$ since $\mathbb{Y} \subseteq \mathbb{S}$.

Algorithm 1 Lazy Aggregation Block Gibbs (LABG)

Input:
Query $\mathbb{Q} = (\mathbb{Y}, \mathbb{E})$
Number of samples N

$\mathbb{S} \leftarrow$ Unroll GBN for $\mathbb{Q}$
$P_\phi \leftarrow$ Compute Full Conditional for $x.A \in \mathbb{S}$
$\mathbf{s}^{(0)} \leftarrow$ Sample initial state
for $t = 1$ to N **do**
  $\mathbf{s}^{(t)} \leftarrow \mathbf{s}^{(t-1)}$
  $X.A \leftarrow$ Select attribute class in $\mathcal{A}(\mathbb{S})$
  LazyAggregation($X.A$) , if necessary
  for all $x.A \in \mathbb{S}(X.A)$ **do**
    Aggregation($x.A$) , if necessary
    $\mathbf{s}^{(t)}\langle x.A\rangle \leftarrow$ Sample $P_\phi(x.A)$
  end for
end for
$P(\mathbb{S} \mid \mathbb{E}) \leftarrow$ Density Estimate of $\{\mathbf{s}^{(0)},...,\mathbf{s}^{(N)}\}$
$P(\mathbb{Y} \mid \mathbb{E}) \leftarrow$ Marginalize $\mathbb{S}\backslash\{\mathbb{Y}\}$ from $P(\mathbb{S} \mid \mathbb{E})$
return $P(\mathbb{Y} \mid \mathbb{E})$

4. Experiments

The algorithm we presented only samples new values for a subset of all sampling variables during a Gibbs step. In order to compare convergence and performance properties, we compare LABG against two other samplers. The *Lazy Aggregation Standard Gibbs* (LASG) sampler makes use of the GBN structure when computing aggregate values, but samples all inference variables in $\mathbb{S}$. We also use a traditional Gibbs sampler that does not make use of the GBN structure and thus recomputes aggregations redundantly.

4.1 Artificial Dataset

To illustrate the computational performance of our algorithm, we use an artificial data set whose associated relational and probabilistic model are depicted in Fig. 3; the local distributions are given in Table 1. The set of entities and relationship classes are $\mathcal{E} = \{A, B, C, D\}$ and $\mathcal{R} = \{AB, AC, AD\}$ respectively and all attribute classes are Bernoulli distributed. The model is designed to cover basic set of possible configurations, namely one 1:n relationship ($A \to C$), one n:1 relationship ($A \to D$) and one m:n relationship ($A \to B$). The constraint $\mathcal{C}$ for all dependencies is the traditional slot chain, e.g. $A.a[A] = A.a[AB], B.a[AB] = B.a[B]$ for the dependency $A.a \to B.a$. Where necessary, the aggregation function used is the average. We are assessing the quality of the posterior (besides checking the convergence) by performing a simple classification query $P(A.a \mid B.A, C.A, D.A)$ for 19 query variables of attribute class $A.a$. The unrolled GBN is of moderate size with 19 sampling nodes and a total of 526 nodes. Figure 4 shows the cumulative mean of the LABG algorithm. Convergence was fast; we used a burn-in of 100 samples and then collected 200 samples from three parallel Gibbs trajectories. This proved sufficient to classify 17 out of the 19 variables correctly using a simple MAP estimate.

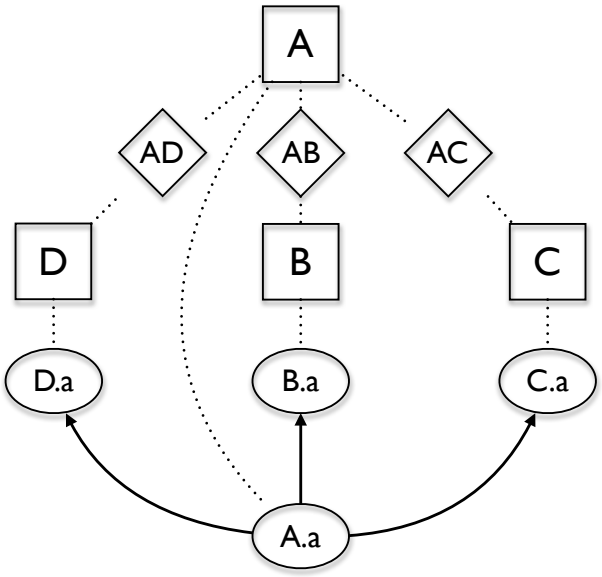


Figure 3: A DAPER model based on the artificial dataset

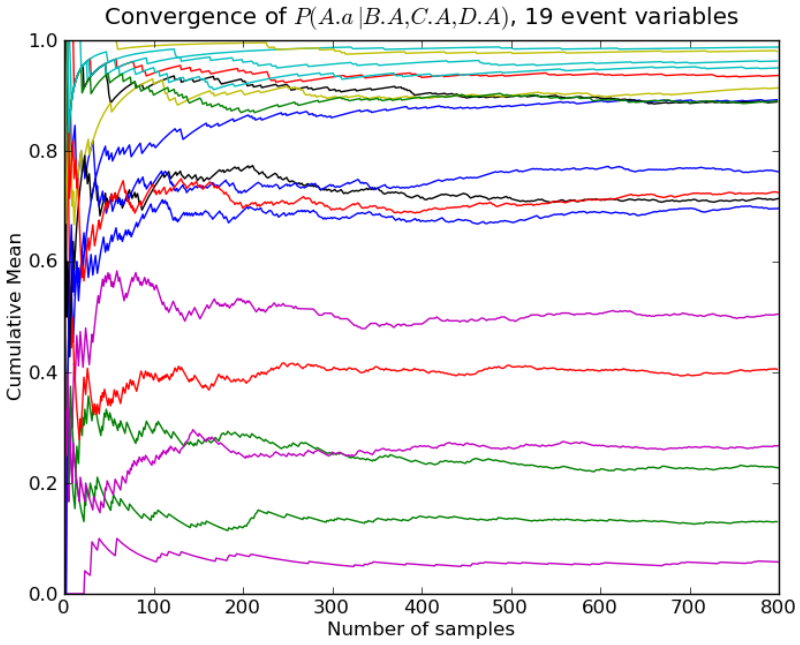


Figure 4: Inference on 19 $A.a$ attribute objects shows solid convergence

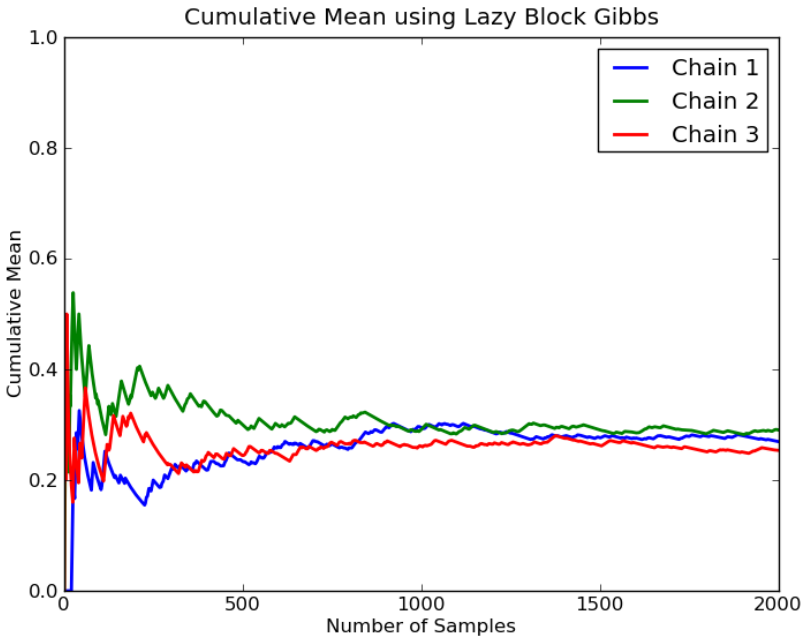


Figure 5: Convergence plot of three parallel chains for the Lazy Aggregation Block Gibbs sampler

	A^0	A^1
	0.29	0.71

	D^0	D^1
A^0	0.34	0.66
A^1	0.6	0.4

	C^0	C^1
A^0	0.27	0.73
A^1	0.42	0.58

	B^0	B^1
A^0	0.59	0.41
A^1	0.31	0.69

Table 1: The local distributions for $P(A.a)$, $P(D.a \mid A.a)$, $P(C.a \mid A.a)$, $P(B.a \mid A.a)$

	Table size
A	100
B	50
C	2000
D	500
AB	200

	Table size
A	500
B	10
C	10000
D	50
AB	1000

Table 2: A small and a large instance of artificial datasets generated using the same local distributions (Table 1)

The model presented above allows a meaningful comparison of the proposed inference methods both in terms of convergence as well as performance. The query selected to ensure that the structure of the GBN is the same for all experiments performs inference on one attribute object $\mathbb{Y} = \{A.a_1\}$ given all attribute objects $\mathbb{E} = \{C.a_i\}$. The structure induced by the 1:n dependency $(A \rightarrow C)$ is a Naive Bayes since the nodes of class C can only have one parent which is already in the event set. Every child node of class D of the n:1 dependency $(A \rightarrow D)$ can have multiple parents in A, and since neither A or D are in the evidence, all loaded nodes of type A and D will be sampling nodes. Hence, the lazy construction of the GBN will also load all the children nodes C of the A nodes that have already been loaded as parents of D. The same recursive mechanism loads nodes of type B as well. The resulting GBN will quickly grow in size and it will contain many children of type B and D with multiple parents in A. This type of query is well suited for the proposed *Lazy Aggregation* method.

To assess the convergence properties, three parallel Gibbs trajectories were run for both LABG and LASG. As the LABG algorithm samples only the variables of one randomly selected attribute class during one Gibbs step, the auto-covariance of the posterior samples is larger than when using LASG. This in turn leads to a slower traversal of the posterior probability mass as the Gibbs jumps in the state space are smaller. Hence, the LABG algorithm needs more samples to approximate the posterior. These effects are illustrated in Figures 5 and 6: LABG needs around 700 samples for convergence whereas LASG converges after around 400 samples. LASG also displays a lower inter-chain variance, because it is more 'guided' than LABG. These effects can be seen in Figure 7, which shows two chains respectively of LABG and LASG.

The *Lazy Aggregation* makes use of the GBN structure while the *Block Gibbs* is a compromise in regards to convergence speed and precision. Both mechanisms are increasing the computational

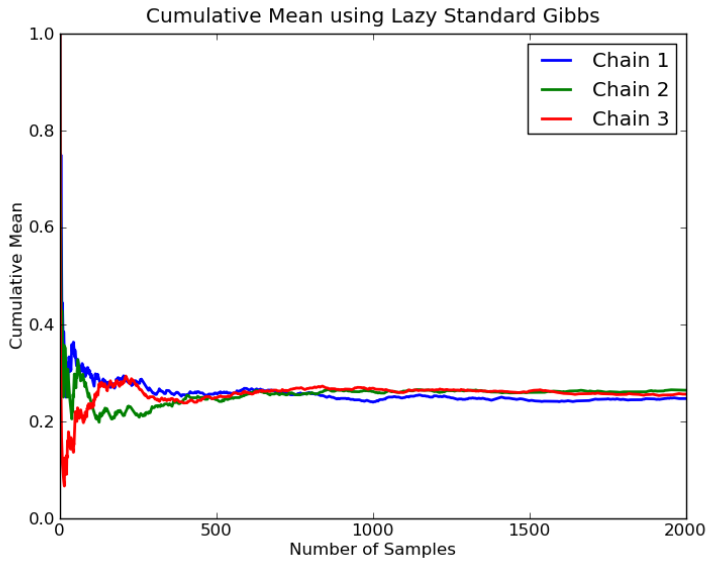


Figure 6: Convergence plot of three parallel chains for the Lazy Aggregation Standard Gibbs sampler

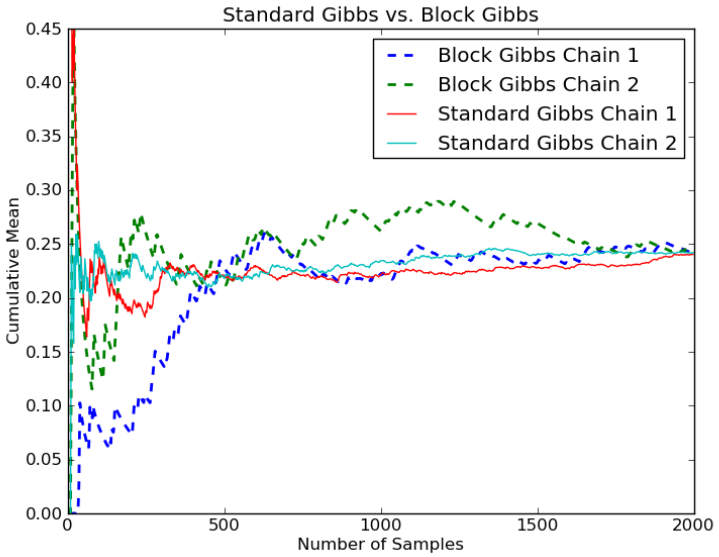


Figure 7: Comparison of convergence between the LABG and LASG samplers. Note that the limit of the y-axis has been changed to allow greater detail in the plot

efficiency of the LABG algorithm (Figure 8): the former by minimizing the number of aggregations that need to be computed, and the latter by sampling fewer variables. This tradeoff seems to pay off, as LABG is about three times faster than LASG, but convergence only takes roughly twice as many samples. Furthermore, *Lazy Aggregation* seems to have a greater effect when the size of the GBN increases, which confirms that we avoid more computation if we have more nodes that share the same children. We conclude that the proposed algorithm scales well with larger queries or when increasing the data set.

Thus the compromise between speed and accuracy is flexible and the algorithm can be adapted depending on the precision needed for the inferential goals. The best results are typically generated using the LASG sampler during the burnin phase for fast convergence, while the samples are being collected efficiently using LABG.

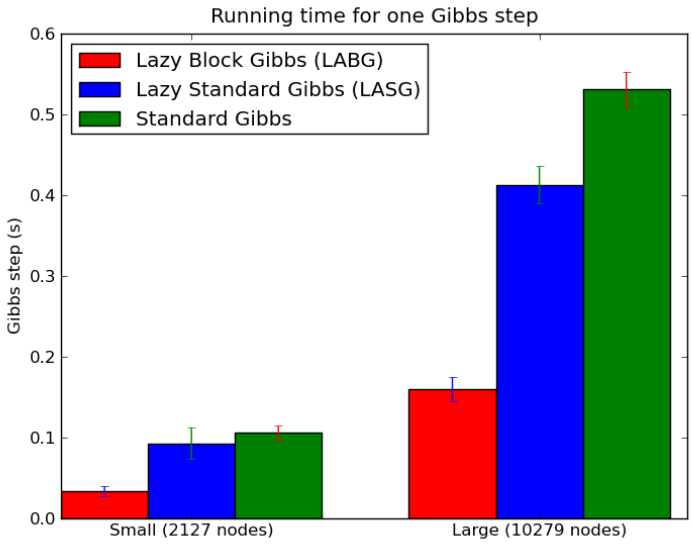


Figure 8: For both the big and small artificial dataset, a comparison of the average time needed to compute one Gibbs step for the three discussed Gibbs samplers.

4.2 Political Contributions

In the United States, running a political campaign to be elected into Congress is an expensive endeavour. The amount of money at the disposal of a candidate is an important factor in a successful campaign. Recognizing this influence and the problems that come with it - corrupt lobbying, vested corporate interests - the recipient of a political contribution is required by law to report the donation. As a consequence of the recent trend towards government transparency and data digitalization, this data is now publicly available for bulk download[1] .

In order to model the data with a PRM, we considered a subset of the data, consisting of the federal contributions for the cycle 2007-2008. The recipients are either individuals running for Congress (House or Senate), Political Action Committees (PACs) or presidential candidates (Barack

1. http://www.transparencydata.com/

Obama, John McCain). To guarantee statistical significance, only recipients who received more than 1000 contributions are included in the model. The political affiliation of candidates for Congress is easily available. PACs on the other hand usually redistribute their donations to candidates of both parties, which makes it harder to determine their affiliation. Each contributor is associated with a name, the US state where the donation was made, and a industry category (e.g. Oil & Gas, Gun Rights, Retired). The size of the dataset and the cardinality of the attributes are displayed in Table 3. We augmented the data with information about the contributor state: a binary variable indicating the income level (above or below the US average) and a binary variable for the political affiliation of the state based on the outcome of the previous presidential election.

Class	Size
Recipient	430
State	50
Donation	~ 2300000
Contributor	~ 2300000

Attribute	Cardinality
Category	115
Recipient.Democratic	2
Amount	6
Income	2
State.Democratic	2

Table 3: The number of objects in the entities and relationships of the PRM and the cardinality of the attributes

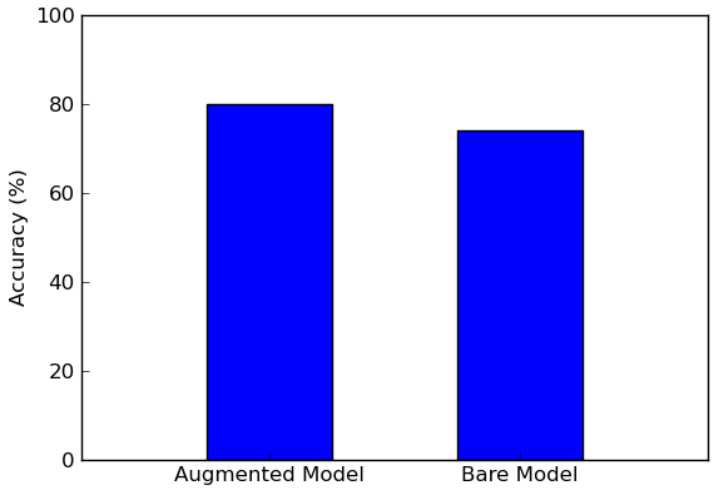


Figure 9: The model making use of the information about the state of the contributor is performing better (80%) than the bare model only making use of the industry category of the contributor (74%)

The query from example 2 attempts to predict the political affiliation based on the donation, contributor and state information. As mentioned above, there is no clear ground truth available for PACs. To examine the quality of the model, we split the data about the individual recipients in a training and test set. The test set contains 30 democratic and 30 republican individual candidates for Congress, randomly selected; the model is trained using the contributions for the remaining 370

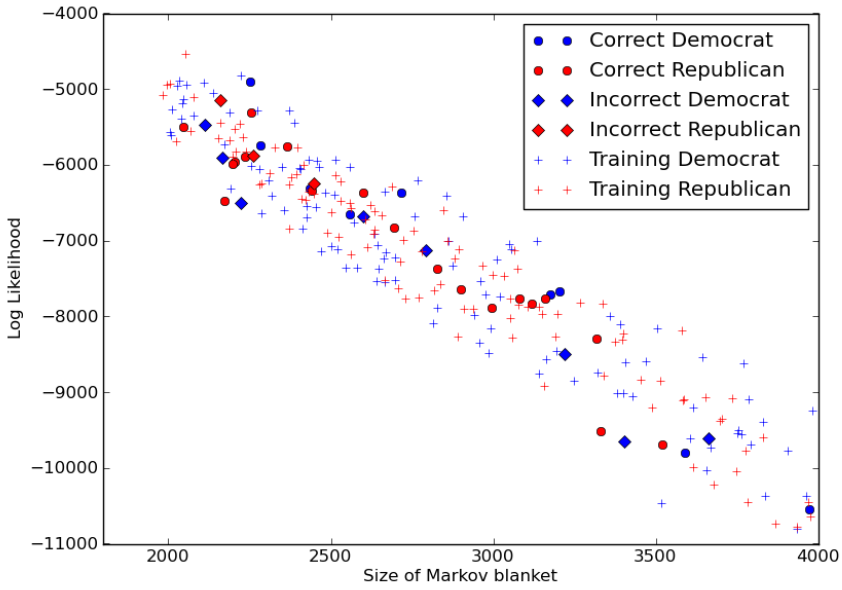


Figure 10: A scatter plot of the size of the Markov blanket and the log likelihood of the recipient. The red and blue colours indicate republican and democratic affiliations respectively. A circle means the recipient has been correctly classified whereas a diamond indicates a misclassification.

recipients. To compute the accuracy of the learned PRM, the proposed inference method (Algorithm 1) is run for each recipient in the test set. The structure of the GBN is different for each query. The number of nodes in a specific GBN depends on the number of contributions the recipient has received. The size of the resulting GBNs and the performance of the algorithm is presented in Table 4.

	GBN Size (nodes)
Min	2045
Max	23201
Average	4482

	Running Time (s)		
	Average	Min	Max
Unrolling GBN	235	209	303
Gibbs Sampling	4.8	2.8	13.3

Table 4: Statistics about the size of the GBNs and the running time of the algorithm. During inference, 200 Gibbs samples showed to be sufficient.

Fig. 9 shows the accuracy of the augmented model compared to a simplified model that did not make use of the additional information about the $State$. The inclusion of the additional information increases the quality of the model. Among the 12 recipients that were classified incorrectly, nine are Democrats and three are Republicans. To gain more insight into the results, we also examine the log-likelihood information. Specifically, we are interested in the log-likelihood of the individual recipients, which can be computed from the Markov blanket of the recipient object. As the number of nodes in the Markov blanket depends on the number of donations a recipient receives, the log-

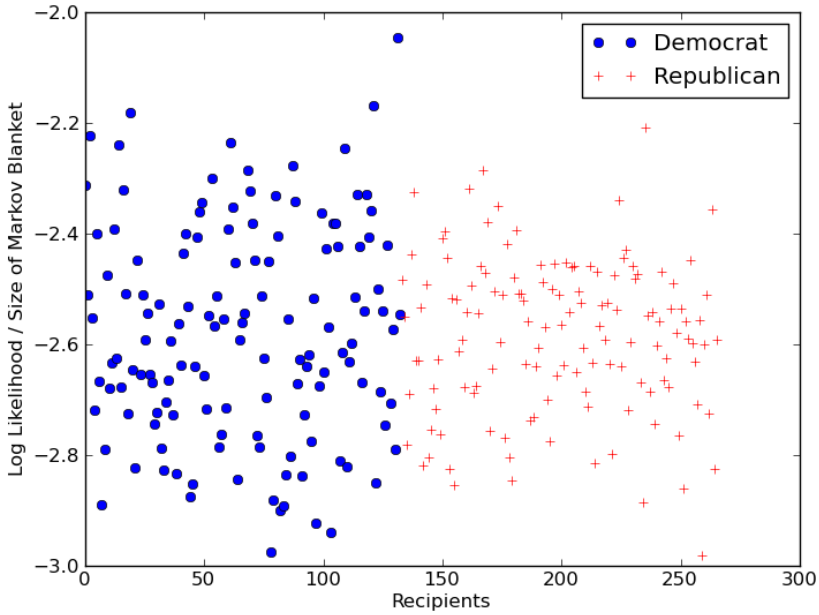


Figure 11: The ratio between the log-likelihood and the size of the Markov blanket allow for a direct comparison of recipients. The mean of the democratic and republican recipients is similar, but the democratic recipients display a higher variance.

likelihood values of the individual recipients cannot be compared directly. Fig. 10 illustrates this correlation; we note that most misclassifications are found in the top-left region where the log-likelihood is high and the number of donations low. By "normalizing" the log-likelihood by the number of nodes in the Markov blanket, we obtain scaled values for each recipient, which are more directly comparable. In Fig. 11 the model's struggle with democratic recipients becomes apparent; even though the mean is almost the same ($dem = -2.57$, $rep = -2.58$), the variance of the scaled likelihood is larger ($dem = 0.041$, $rep = 0.015$) for the democratic recipients. We conclude that the pattern of contributions of democratic recipients is more complex to model than for republican contributions.

5. Conclusions and future work

We presented an approximate inference method for probabilistic relational models which scales well with the size of the ground Bayesian network. Note that we have an implemented software package that allows the specification of a DAPER model as presented, the learning of its parameters as well as query specification and answering. An advantageous sampling order and aggregation are central parts of this framework, and we leverage the special structure of the GBN as much as possible, to increase efficiency. The framework is able to efficiently learn a model from real data and its flexibility makes it suitable for exploratory data analysis. We are currently testing the package on much larger data sets, as well as on some models where ground truth is available, and where we can

assess precisely the accuracy of the predictions. We are also augmenting the software package with other types of distributions and learning algorithms.

Acknowledgements

The authors gratefully acknowledge the financial support provided by NSERC for this research.

References

N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *IJCAI*, pages 1300–1309, 1999.

Lise Getoor. Learning probabilistic relational models. In *SARA '02: Proceedings of the 4th International Symposium on Abstraction, Reformulation, and Approximation*, pages 322–323. Springer-Verlag, 2000.

Lise Getoor and Ben Taskar. *Introduction to Statistical Relational Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2007.

D. Heckerman, C. Meek, and D. Koller. Probabilistic models for relational data. Technical Report MSR-TR-2004-30, Microsoft Research, 2004.

J. Kisynski and D. Poole. Lifted aggregation in directed first-order probabilistic models. In *IJCAI*, pages 1922–1929, 2009.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

W. Meert, N. Taghipour, and H. Blockeel. First-order Bayes-ball. In *ECML*, pages 369–384, 2010.

B. Milch, L. Zettlemoyer, K. Kersting, M. Haimes, and L.P. Kaelbling. Lifted probabilistic inference with counting formulas. In *AAAI*, pages 1062–1067, 2008.

H. Pasula and S. Russell. Approximate inference for first-order probabilistic languages. In *IJCAI*, pages 741–748, 2001.

J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, 1988.

A.J. Pfeffer and D. Koller. Semantics and Inference for Recursive Probability Models. In *AAAI*, pages 538–544, 2000.

R.D. Shachter. Bayes-ball: The rational pastime (for determining irrelevance and requisite information in belief networks and influence diagrams). In *UAI*, pages 480–487, 1998.

N.L. Zhang and D. Poole. Exploiting causal independence in bayesian network inference. *JAIR*, 5: 301–328, 1996.