Approximate inference for first-order probabilistic languages

Hanna Pasula and Stuart Russell
Computer Science Division, University of California
387 Soda Hall, Berkeley, CA 94720-1776
{pasula,russell}@cs.berkeley.edu

Abstract

A new, general approach is described for approximate inference in first-order probabilistic languages, using Markov chain Monte Carlo (MCMC) techniques in the space of concrete possible worlds underlying any given knowledge base. The simplicity of the approach and its lazy construction of possible worlds make it possible to consider quite expressive languages. In particular, we consider two extensions to the basic relational probability models (RPMs) defined by Koller and Pfeffer, both of which have caused difficulties for exact algorithms. The first extension deals with uncertainty about relations among objects, where MCMC samples over relational structures. The second extension deals with uncertainty about the identity of individuals, where MCMC samples over sets of equivalence classes of objects. In both cases, we identify types of probability distributions that allow local decomposition of inference while encoding possible domains in a plausible way. We apply our algorithms to simple examples and show that the MCMC approach scales well.

1 Introduction

Recent work in AI has made clear the advantages to be derived from combining probability theory with (at least some of) the expressive power of first-order logic [Wellman *et al.*, 1992]. We will call languages that exhibit such a combination *first-order probabilistic languages* (FOPLs). The ability to handle objects, relations, and quantification gives such languages a huge advantage in representational efficiency in certain situations. To take a purely logical example: the rules of chess can be written in about one page of Prolog but require perhaps millions of pages in a propositional language. A recent thesis on first-order probabilistic languages [Pfeffer, 2000] describes a battlespace management system involving potentially thousands of objects whose relationships are unknown and changing. [Pasula *et al.*, 1999] describe a freeway traffic surveillance involving probabilistic inference about the identities and properties of thousands of vehicles. These applications would be infeasible without some ability to specify and reason with FOPL knowledge bases.

The semantics of FOPLs are based on the idea that each model of a FOPL knowledge base should be viewed as a probability measure over the possible worlds (logical models) defined by the constant, function, and predicate symbols of the knowledge base [Halpern, 1990]. Although "wildly undecidable" in full generality, highly restricted FOPLs appear to be practical, especially with finite models. Two threads have arisen, based on semantic networks (e.g., [Koller and Pfeffer, 1998]) and logic programming (e.g., [Sato and Kameya, 1997]).

In this paper, we focus on the family of *relational probability models* (RPMs) [Pfeffer, 2000], although our ideas apply equally to other languages. RPMs, like semantic networks, are based on *classes* containing *instances*, with each instance possessing *attributes*. (See Section 2 for details.) RPMs allow one to specify probability distributions over the attribute values of an instance, either directly or via inheritance from classes. These distributions may depend on other attribute values of the instance or of other instances. For example, a PhD student's success may depend on the fame of his or her advisor.

[Pfeffer *et al.*, 1999] describe an exact inference algorithm for RPM knowledge bases called *structured variable elimination* (SVE). Roughly speaking, SVE applies the variable elimination algorithm [Zhang and Poole, 1996] to a dynamically constructed Bayesian network whose nodes are all those ground propositional variables defined by the knowledge base and relevant to the current query. SVE derives an efficient variable ordering from the structure of the knowledge base and reuses computation results where possible. It is often able to answer queries involving hundreds of variables in a few seconds.

Despite SVE's excellent performance, its runtime is at least exponential in the size of the largest clique in the optimal triangulation of the network. The expressive power of RPMs makes it very easy to construct knowledge bases whose corresponding Bayesian networks have very large cliques. For example, [Pfeffer, 2000] describes a model for matches in a sports league. The knowledge base consists of a single, generic conditional distribution for the outcome of a match given the quality of the two teams, a single prior distribution for the quality of the teams, and the results of some matches. If every team plays every other, then the team qualities form a clique in the corresponding Bayesian network and the inference cost is exponential in the number of teams. Similar problems will arise in any application in which there are complex relationships among large numbers of objects—i.e., precisely those domains for which FOPLs are really necessary.

The situation is exacerbated when the RPM language is extended to allow for *structural uncertainty*—i.e., uncertainty about which probabilistic dependencies actually exist. We consider two forms of structural uncertainty:

reference uncertainty: the value of a relational attribute may be uncertain—e.g., we may not know which of two professors is the advisor of a certain student;

identity uncertainty: we may not know whether two objects in the knowledge base are the same—e.g., when we see a red bus at two different camera locations on the freeway.

Each of these extensions may lead to Bayesian networks whose size alone causes difficulties and whose high connectivity makes exact inference completely impractical. For example, the inference problem in the freeway surveillance application of [Pasula *et al.*, 1999] is known to be #P-hard, i.e., almost certainly exponential in the number of vehicles.

The solution proposed by [Pasula *et al.*, 1999] is to use a Markov chain Monte Carlo (MCMC) algorithm (see Section 3 for details). The algorithm samples from possible matchings among vehicles, converging (in some cases polynomially) to approximately correct probabilities. Often a few hundred samples suffice for a state space of 2^{1000} states. The states being sampled are essentially the possible worlds defined by the constant symbols (observed vehicles) and predicates (equality) of the knowledge base. This approach—sampling possible worlds with MCMC—can be turned into a general inference algorithm for first-order probabilistic languages, as suggested by [Russell, 1999; Pfeffer, 2000].

In this paper, we investigate MCMC on possible worlds as an inference method to handle reference uncertainty (Section 4) and identity uncertainty (Section 5). We show how the possible worlds may be constructed dynamically and how the transition probabilities may be computed efficiently. For the case of transitions involving a referentially uncertain relational attribute value, we identify a large family of conditional distributions that render the calculation independent of all but the particular values involved in the transition. We illustrate the algorithms using simple examples and give experimental results suggesting that the algorithms scale well.

2 Relational probability models

The following definitions are adapted from [Koller and Pfeffer, 2000]. A relational probability model, in its most basic form, consists of

- A set $\mathcal{C}$ of *classes* denoting sets of objects, related by subclass/superclass relations.
- A set $\mathcal{I}$ of *named instances* denoting objects, each an instance of one class.
- A set $\mathcal{A}$ of *complex attributes* denoting functional relations. Each complex attribute A has a domain type $Dom[A] \in \mathcal{C}$ and a range type $Range[A] \in \mathcal{C}$.

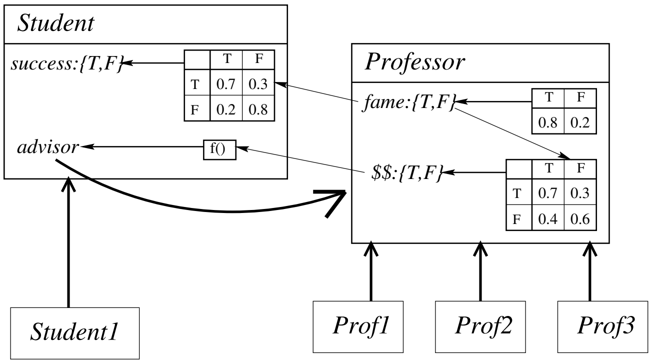

Figure 1: The simple RPM defined in the text, with its associated conditional probability models.

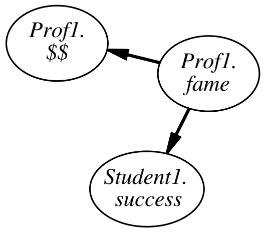

Figure 2: Bayesian network structure generated from the RPM in Figure 1. The *advisor* relationship, being certain, doesn't appear.

- A set $\mathcal{B}$ of *simple attributes* denoting functions. Each simple attribute B has a domain type $Dom[B] \in \mathcal{C}$ and a range that is a finite, enumerated set of values $Val[B]$.
- A set of conditional probability models $P(B|Pa[B])$ for the simple attributes. $Pa[B]$ is the set of B's *parents*, each of which is a nonempty chain of (appropriately typed) attributes $\sigma = A_1.\cdots.A_n.B'$, where B' is a simple attribute. Probability models may be attached to instances or inherited from classes.

For now, we will assume that the values of all the complex attributes are known (no reference uncertainty), and that every instance is distinct (unique names assumption, hence no identity uncertainty). Thus, a possible world is defined by the values of the *instance variables*—the simple attributes for all named instances.

Consider the following very simple RPM. There are two classes, *Student* and *Professor*, and two instances, *Student*$_1$ and *Prof*$_1$. There is one complex attribute, *advisor* (mapping *Student* to *Professor*) and the *advisor* of *Student*$_1$ is *Prof*$_1$. There are three simple attributes, all Boolean: *success* of a $Student$ and *fame* and *$$* (funding level) of a $Professor$. For any *Student* s, s.*success* has one parent, s.*advisor.fame*, with an appropriate conditional distribution. For any *Professor* p, p.*fame* has no parents and a simple prior distribution, while p.*$$* depends on p.*fame*. Figure 1 shows the knowledge base with probability distributions and Figure 2 shows the Bayesian network structure for the attribute variables definable from the knowledge base.

3 Markov chain Monte Carlo algorithms

MCMC [Gilks *et al.*, 1996] generates samples from a posterior distribution $\pi(x)$ over possible worlds x by defining a Markov chain whose states are the worlds x and whose *stationary distribution* is $\pi(x)$. In the *Metropolis–Hastings* method (henceforth M-H), transitions in the Markov chain are constructed in two steps:

- Given the current state x, a candidate next state is generated from the *proposal distribution* $q(x'|x)$, which may be (more or less) arbitrary.
- The transition to x' is not automatic, but occurs with an *acceptance probability* defined by

$$\alpha(x'|x) = min\left(1, \frac{\pi(x')q(x|x')}{\pi(x)q(x'|x)}\right)$$

It is not necessary that all the variables of state x be updated simultaneously, in a single transition function. *Single-component* M-H alters each variable in turn. It is also possible to factor q into separate transition functions for various subsets of variables. Provided that q is defined in such a way that the chain is ergodic, this transition mechanism defines a Markov chain whose stationary distribution is $\pi(x)$.

The Gibbs sampling algorithm for Bayesian networks [Pearl, 1988] is a special case of Metropolis–Hastings in which the proposal distribution samples a single variable X_i using the distribution $P(X_i|\mathbf{mb}(X_i))$, where $\mathbf{mb}(X_i)$ denotes the current values of the variables in the Markov blanket of X_i (its parents $Pa[X_i]$, children Y_j, and children's other parents). In this case, the acceptance probability is always 1. One can show easily that

$$P(X_i|\mathbf{mb}(X_i)) = \alpha P(X_i|Pa[X_i])) \prod_j P(Y_j|Pa[Y_j]) \quad (1)$$

Gibbs sampling is very simple and also *local*: transitions are generated referring only to parts of the model directly connected to the variable in question. Hence, *the cost per transition is typically independent of model size*. M-H sampling is also typically local because all the parts of the model that are not changed by the transition cancel in the ratio $\pi(x')/\pi(x)$. In particular, if the proposal concerns a single variable X_i, this ratio reduces to $P(x'_i|\mathbf{mb}(X_i))/P(x_i|\mathbf{mb}(X_i))$, where x'_i is the proposed value of X_i and x_i is its current value. The M-H algorithm, unlike Gibbs, has the added advantage that the transition may often be computed without referring to the other values of X_i at all, as we will see.

4 Reference uncertainty

Reference uncertainty arises whenever relations among objects, as described by complex attribute values, are not known with certainty. For example, we may be unsure as to which of three professors is *Student*$_1$'s advisor. We need to be able to describe this uncertainty and to specify the dependencies that influence it. The following definitions are adapted from [Pfeffer, 2000]:

- With each complex attribute A, we associate a simple *reference attribute* $ref[A]$, such that $Val[ref[A]]$ is a finite, enumerated set of named instances. (Where no confusion arises, we may drop the $ref[\cdot]$ and use the attribute name itself.) Dependencies are expressed as before by a conditional distribution $P(ref[A]|Pa[ref[A]])$. The parents of $ref[A]$ are those attributes or attribute chains that influence the choice of an instance as the value of attribute A.
- Reference uncertainty modifies the definition of attribute chains. Suppose that an attribute B depends on the parent chain $\sigma = A_1. \cdots .A_n.B'$. Any of the complex attributes in the chain may be uncertain. Then the parent variables for B are all the instance variables reached by the chain for all possible combinations of values of all the uncertain complex attributes, as well as all the reference variables for those attributes.

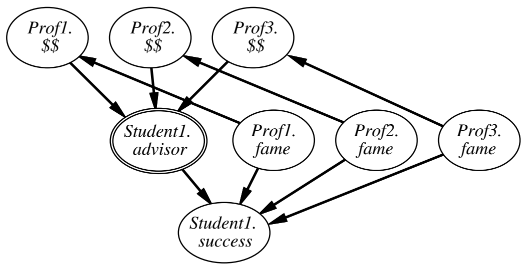

Figure 3: Bayesian network structure with reference uncertainty about *Student*$_1$*.advisor*. The reference attribute is shown in a double oval.

The simple example of Figure 1 can be extended to include reference uncertainty, if *Student*$_1$*.advisor* is unknown. We define a reference attribute ref[*Student*$_1$*.advisor*] with range (say) $\{Prof_1, Prof_2, Prof_3\}$. The choice of advisor depends (generically) on the funding (*Prof.$$*) of each candidate, which depends (generically) on *Prof.fame*. This gives the instance-variable network structure shown in Figure 3. Obviously, when reference attributes have many possible values, very large implicit network structures can result.

4.1 Exact inference with reference uncertainty

As mentioned above, the runtime of any variable elimination algorithm is exponential in the size of the largest clique in the optimally triangulated graph. Looking at Figure 3, it is apparent that a straightforward application leads to two "large" cliques: one containing *Student*$_1$*.success* and its parents, and one containing ref[*Student*$_1$*.advisor*] and its parents. In general, these will have $n + 2$ and $n + 1$ members respectively, where n is the number of possible values for the reference attribute. Thus, inference cost grows exponentially with this number.

[Pfeffer, 2000] observes that at least one of these cliques—the one associated with *Student*$_1$*.success*—can be decomposed. *Given a known value for* ref[*Student*$_1$*.advisor*], *Student*$_1$*.success* does *not* depend on the fame of other professors (Figure 4). This is an extreme form of context-specific independence, and allows the $n + 2$-variable factor to be replaced by a product of n 3-variable factors in the variable elimination process. This decomposition applies in general

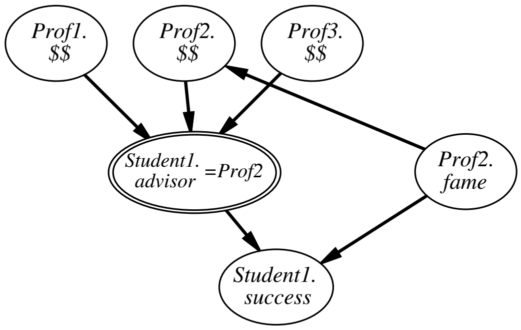

Figure 4: The network structure conditioned on *Student*$_1$.*advisor* = *Prof*$_2$.

to variables that have referentially uncertain parents.

Unfortunately, the clique associated with the reference attribute itself—ref[*Student*$_1$.*advisor*] in this case—cannot be decomposed. We are left with a network structure that seems (in general) to be intractable for exact inference. The situation becomes worse still as more complex interactions occur among several reference attributes.

4.2 MCMC inference with reference uncertainty

In our approach, we extend the MCMC algorithm commonly used on Bayesian networks by augmenting the Markov chain state space to include reference variables denoting relational uncertainty, and defining appropriate transition functions. Simple attributes, such as *Student*$_1$.*success*, use Gibbs sampling as usual—and context-specific independence can sometimes be used to simplify those steps further. Reference attributes, such as ref[*Student*$_1$.*advisor*], depend on a slightly more involved M-H step.

The Gibbs steps

In what follows, we abbreviate *Student*$_1$ to S_1, *advisor* to a, and so on. Let us consider an extended version of Figure 3 with n potential advisors. Let the current value of $S_1.a$ be P_i. Suppose we want to apply Gibbs sampling to $P_j.f$ (*Prof*$_j$.*fame*), where $j \neq i$. By Equation (1), the sampling distribution is

$$\begin{aligned} &\alpha P(P_j.f)P(P_j.\$\$|P_j.f)P(S_1.s|P_1.f,\ldots,P_n.f,S_1.a=P_i) \\ &= \alpha P(P_j.f)P(P_j.\$\$|P_j.f)P(S_1.s|P_i.f,S_1.a=P_i) \\ &= \alpha' P(P_j.f)P(P_j.\$\$|P_j.f) \end{aligned}$$

The first step of this derivation uses the fact that $S_1.s$ depends only on the fame of S_1's advisor, $P_i.f$, and not on other professors; the second step simply observes that $P(S_1.s|P_i.f, S_1.a = P_i)$ is constant w.r.t. $P_j.f$. Thus, when the reference attribute is instantiated, sampling operates exactly as if the links from non-selected parents were nonexistent. The same holds when sampling a child of the reference variable (e.g., $S_1.s$). Thus, while reference variables remain constant, the network has a simplified form, such as that shown in Figure 4. MCMC sampling on this simpler network should converge quickly. When the value of the reference variables changes, however, so does the effective structure of the network. We now address this issue.

The Metropolis–Hastings steps

Gibbs sampling for a reference variable with n values involves considering n possible network structures, so we apply M-H sampling instead. M-H proposes a single new value for the reference variable, and then decides whether to accept it. (We can ignore the proposal distribution, which we will assume for now is uniform and hence cancels.) For the transition from $S_1.a = P_i$ to $S_1.a = P_j$, then, we need (from Equation (1), and simplifying based on known values of the reference variable) the ratio

$$\frac{P(S_1.a=P_j|P_1.\$\$,\ldots,P_n.\$\$)P(S_1.s|P_j.f,S_1.a=P_j)}{P(S_1.a=P_i|P_1.\$\$,\ldots,P_n.\$\$)P(S_1.s|P_i.f,S_1.a=P_i)}$$

At first sight, it seems that calculating this ratio requires accessing the current values of $P_1.\$\$, \ldots, P_n.\$\$$, i.e., the funding levels of all possible candidate professors, even though the transition involves just two of them. (This is because the probability of picking any one advisor *does* depend on the funding levels of all candidates.) In turn, this requires that all those nodes be constructed and instantiated, which we prefer to avoid if possible.

It so happens, fortunately, that conditional distributions for "selecting" a value for a relational attribute, given properties of a set of candidates, may have some structural properties that simplify the task. Suppose, for example, that for all i,

$$P(S_1.a=P_i|P_1.\$\$,\ldots,P_n.\$\$) = \frac{f(P_i.\$\$)}{\sum_k f(P_k.\$\$)}$$

for some arbitrary function f. Then, in the transition probability ratio given above, the summations $\sum_k f(P_k.\$\$)$ cancel, leaving

$$\frac{f(P_j.\$\$)P(S_1.s|P_j.f,S_1.a=P_j)}{f(P_i.\$\$)P(S_1.s|P_i.f,S_1.a=P_i)}$$

which does not mention any values for the reference variable besides P_i and P_j. The property of conditional distributions that we require is satisfied by some well-known conditional distributions, including the *softmax* distribution:

$$P(X=i|Y_1=y_1,\ldots,Y_n=y_n) = \frac{e^{y_i}}{\sum_k e^{y_k}}$$

Softmax is indeed a reasonable model for selecting an advisor based on funding.

The complete algorithm

Once we put the two types of steps together, the complete algorithm alternates between ordinary Gibbs steps on a simplified network and M-H steps altering the network structure. It should be noted that different network structures may result in different sets of variables being, at any given time, relevant or irrelevant to the query. Our approach restricts computation to variables that are strictly relevant [Zhang and Poole, 1996] according to the current structure.

A further possible enhancement is *lazy construction* of the Bayesian network. The network can, e.g, be grown "from the query outwards", with nodes added as they are needed to sample a node currently in the network. (Structural variables are, like others, instantiated on creation, yielding a network

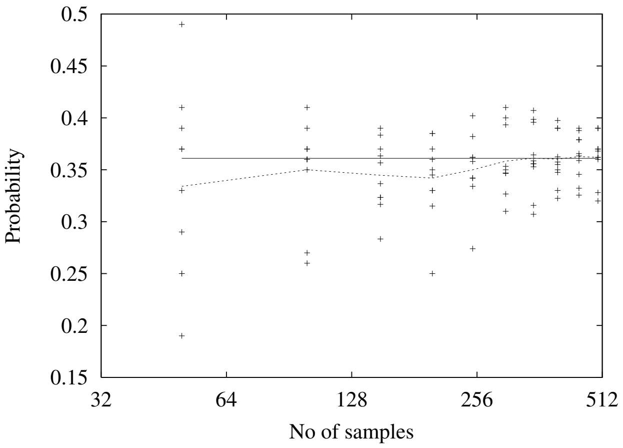

Figure 5: MCMC convergence on a simple example with three professors and one student. The exact probability is 0.361 (horizontal line). Each point represents the estimate from one of 10 Markov chains after a given number of samples (log scale). The dotted line shows the average of the 10 individual estimates.

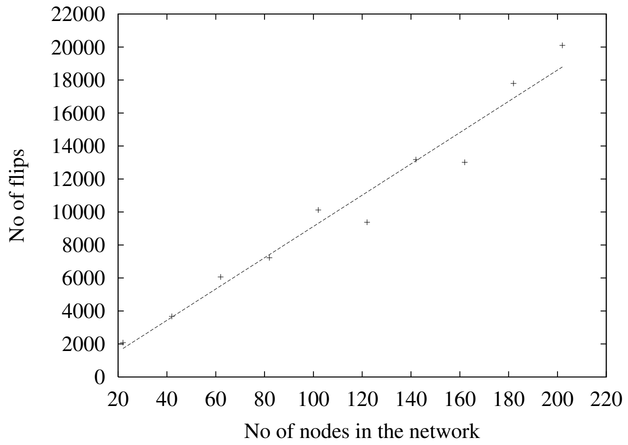

Figure 6: MCMC convergence as a function of network size. The y-axis measures the number of state transitions required to reach a preset convergence threshold for a standard diagnostic. The straight line is a regression fit to the data, which is averaged over 20 trials.

in the usual simplified format.) If necessary, nodes may then be discarded when they become irrelevant to the query under the current network structure. By using an "intelligent" M-H proposal, as discussed in Section 6, we can also focus computation on variables that are "important" to the query—perhaps visiting only an infinitesimal fraction of the potentially relevant variables—without sacrificing the guarantee of convergence to correct probabilities in the limit. This should permit us to apply our approach to very large, or even infinite, knowledge bases.

4.3 Experimental results

To check our algorithm, we applied it first to the knowledge base with reference uncertainty, no identity uncertainty, and three professors. In this case, the algorithm generates and evaluates the network shown in Figure 3. We set values for fame or funding for each professor, and queried the posterior probability of $Student_1.success$. In this tiny model, we can compute the exact value for comparison. The MCMC estimates in Figure 5 clearly converge to the correct value.

We also tested the scalability of the algorithm by trying it out on various types of networks of increasing size. As an example, let us consider networks with n professors and $4n + 1$ students, each of whom could have any one of the professors as advisor. We specified the success (or otherwise) of $4n$ students and queried the success of the last. Figure 6 shows the inference cost as a function of the total network size. Because we cannot compute the exact values for all n, we use a standard convergence diagnostic due to [Gelman, 1996], checking against the exact values for small n. The resulting graph appears to be linear in the size of the network. It would appear that, for this type of network, the sampling algorithm scales well. Note also that cost is measured in terms of *single-variable* state transitions performed by the algorithm, hence the number of transitions *per variable* to reach convergence is approximately constant, regardless of network size. This held in all our experiments, including experiments (not reported here) on genetic inheritance with uncertain parentage, where the number of cascaded reference variables increases with n.

5 Identity uncertainty

Standard relational probability models [Pfeffer, 2000] incorporate the unique names assumption, i.e., the assumption that each instance present in the knowledge base corresponds to a different object in any given possible world. When this assumption is removed, we must consider the possibility that several instances may denote the same object. Identity uncertainty is especially prevalent in settings where an agent perceives multiple objects over time [Pasula *et al.*, 1999], but also occurs in almost all real databases where "duplicate" records abound.

With identity uncertainty, a "possible world" must specify not only the attribute values for all objects, but also the mapping from instances in the knowledge base to objects in the possible world. (Without identity uncertainty, this mapping is one-to-one and hence no distinction between instances and objects is needed.) We represent this mapping using an equivalence relation—a set of equivalence classes, each of which contains all the instances that co-refer in that particular possible world.[1]

Consider the following example, a simplified version of the data association problem studied by [Pasula *et al.*, 1999]. There are two classes, $Vehicle$ and $Observation$. There is one complex attribute, $generated\_by$, which maps an $Observation$ to the $Vehicle$ that generated it. Each $Observation$ of a vehicle reports a $colour$, which depends (through a probabilistic model of the sensing process) on the $colour$ of the corresponding $Vehicle$. Instances are added to the KB as follows: whenever a vehicle is detected at some

[1]Note that it doesn't matter *which* object the instances denote—objects are essentially placeholders in a relational structure.

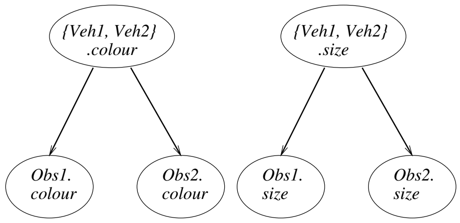

Figure 7: Bayesian network structure for the two-vehicle case, where the vehicles are currently assumed to be one and the same object.

sensor, an $Observation$ is instantiated with its colour set to the measured value, and with a new $Vehicle$ instance attached. In this domain, inference involves reasoning about which $Vehicle$s are, in fact, identical.[2]

Figure 7 shows the Bayesian network structure generated when the knowledge base contains observations Obs_1 and Obs_2 of vehicles Veh_1 and Veh_2, and when the equivalence relation on instances is set to $\{\{Veh_1, Veh_2\}\}$ (i.e., Veh_1 and Veh_2 are the same object).

5.1 Defining a probability distribution

Each possible world now contains an equivalence relation $\omega \in \Omega$ as well as the attribute values, and the knowledge base must now specify a unique probability distribution over this enlarged space of worlds. We can write

$$P(\omega, \langle\text{attribute values}\rangle) = P(\langle\text{attribute values}\rangle|\omega)P(\omega) \quad (2)$$

Given each fixed ω, identity uncertainty is eliminated, and the probability distribution over the attributes can be defined as before. All that remains is to specify the prior $P(\omega)$. When doing so, we can use the fact that instances in disjoint classes cannot co-refer—for example, vehicles cannot be observations. $P(\omega)$ can thus be factored into terms dealing with each of the classes. Even then, however, the state space of each factor is exponential in the number of objects in its class. Fortunately, there are more compact ways of expressing the ω-distribution for each class. The simplest is to give an explicit distribution over the *number* of objects in each class, which can be done by adding a number attribute to classes in the knowledge base. In more complex situations, such as full data association, the ω-distributions are specified implicitly in conjunction with object arrival and detection models [Pasula *et al.*, 1999]. The general topic of modular specification of ω-distributions is likely to require a good deal of further research.

5.2 Inference with identity uncertainty

In the presence of identity uncertainty, exact inference calls for a summation over all possible values of ω. This is clearly infeasible when $|\Omega|$ is large—and $|\Omega|$ is exponential in the number of instances in identity-uncertain classes. MCMC permits us to replace the summation with a sample. The algorithm now works as follows. In addition to "normal transitions," which run as before given a fixed ω, the algorithm performs "identity transitions" in which ω changes. Because of the large state space of ω, the latter type of transition uses M-H steps. Moreover, these steps propose changes not only to ω, but also to some of the conventional attributes.

The reason for this is that an identity transition can propose (among other things) that two currently separate instances be grouped together in the same equivalence class. Clearly, if these two instances currently have different values for the same attribute, the probability of the destination state will be zero. Hence, identity transitions will have a chance of occurring only when the instances "line up"—that is, a sequence of normal transitions causes all of their attribute values to match exactly. When instances have many attributes, line-up may be very rare, resulting in a low acceptance ratio and a slow algorithm. Moreover, there are cases in which waiting for line-up makes the Markov chain non-ergodic—the algorithm can be trapped in a subset of possible ωs with no escape.

Identity transition proposals

Consider a current assignment ω, and a proposed assignment ω'. Let u be the set of unobserved attributes of all the objects in ω affected by the transition, and let u' be the unobserved attributes of the corresponding objects in ω'. Let o be the set of all other attributes within the Markov blankets of the attributes in u or u': this may include observed attributes, as well as the unobserved attributes of other objects not affected by the transition. Note that all attributes in o remain fixed throughout the transition.

Each identity transition proceeds as follows:

- ω' is chosen using a proposal $q_I(\omega'|\omega)$. This proposal can take many forms: one simple possibility is to suggest that a randomly selected instance move from one equivalence class to another. Another method might suggest merging two equivalence classes, or splitting an equivalence class in two.
- Values for all of u' are chosen using a proposal $q_U(u'|\omega', \omega, u, o)$. Some possible choices for q_U will be explained below.
- The proposed change is accepted with probability

  $$min\left(1, \frac{\pi(\omega')q_I(\omega|\omega')\pi(o, u'|\omega')q_U(u|\omega, \omega', u', o)}{\pi(\omega)q_I(\omega'|\omega)\pi(o, u|\omega)q_U(u'|\omega', \omega, u, o)}\right)$$

  which is derived from Equation (3) using Equation (2) and the two-step nature of our proposal.

As an example, let us consider the world state portrayed in Figure 7. In this situation,

- $\omega = \{\{Veh_1, Veh_2\}\}$
- $u = \{\{Veh_1, Veh_2\}.colour, \{Veh_1, Veh_2\}.size\}$

A split can now be proposed as follows:

- $\omega' = \{\{Veh_1\}, \{Veh_2\}\}$
- $u' = \{Veh_1.colour, Veh_2.colour, Veh_1.size, Veh_2.size\}$

To ensure that the chain is ergodic, new values must then be proposed by q_U for at least the newly created attributes.

[2] An alternative interpretation for this knowledge base, more familiar to statisticians, is that there are an unknown number of balls in a bag; balls are pulled out one at a time, their colour is measured, and they are replaced.

Choosing the attribute values

One possible q_U proposal picks a value uniformly at random from the values available at each attribute. This approach is simple to implement and yields an acceptance ratio that is easy to calculate. The $q_U(u|\ldots)$ terms reduce to $1/\sum_{u_i}|u_i|$. This is the algorithm that we test experimentally below.

Unfortunately, this q_U proposal may suggest very unlikely value combinations for the attributes, resulting in low acceptance ratios. We might be better off with a more intelligent proposal mechanism, such as likelihood weighting which generates samples from $q_U(u|o,\omega) = \prod_{x_i \in u} P(x_i|Pa(x_i),\omega)$ together with weights $w(u|o,\omega) = \prod_{x_i \in o} P(x_i|Pa(x_i),\omega)$. Since

$$\pi(u|\omega,o) = \frac{w(u|o,\omega)q_U(u|o,\omega)}{\sum_U w(u=U|o,\omega)q_U(u=U|o,\omega)}$$

the fraction in the acceptance ratio can now cancel to give

$$\frac{\pi(\omega'|o)q_I(\omega|\omega',o)w(u'|o,\omega')\sum_U w(U|o,\omega)q_U(|o,\omega)}{\pi(\omega|o)q_I(\omega'|\omega,o)w(u|o,\omega)\sum_{U'} w(U'|o,\omega')q_U(U'|o,\omega')}$$

This ratio is no longer so simple, as performing the summations may be quite time-consuming. Fortunately, whenever the set of uncertain attributes can be split into several d-separated sets, the summations can be rewritten as products of summations over these sets. If the sets are all small, this algorithm may be feasible.

Yet another approach is to run a "local" MCMC algorithm inside q_U to generate attribute values that reflect the current values of o and hence are likely to be accepted. (When this method is used, care must be taken to ensure that the chain is run until convergence: stopping it prematurely may result in biased answers.)

5.3 Experimental results

To check our identity uncertainty algorithm, we first performed experiments analogous to those in Section 4.3. We began with a problem that was small enough for exact calculation: it consisted of five observations with known attribute values, and queried the posterior probability of the vehicle generating the first observation. Figure 8 shows that our algorithm's estimates converge to the correct value over time.

We then tested the scalability of the algorithm by, once more, applying it to randomly generated networks of increasing size, and measuring time to convergence using a standard diagnostic. The results shown in Figure 9 were obtained by constructing larger and larger sets of vehicle–observation pairs, and requesting a posterior distribution over the number of vehicles. As before, the algorithm appears to scale well.

Finally, we performed an additional experiment to demonstrate that our algorithm works as expected. It should be true that, if observations are generated from a specific set of vehicles, increasing the number of observations in the knowledge base will enable us to model characteristics of that set of vehicles with increasing accuracy. Figure 10 shows that adding more and more observations does enable us to infer the number of vehicles responsible for their generation.

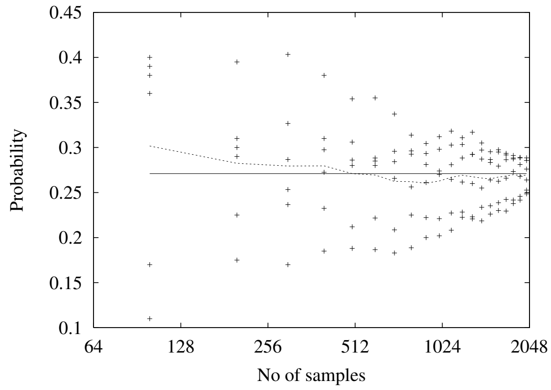

Figure 8: MCMC convergence on a simple example with five observed vehicles. The exact probability is 0.271 (horizontal line). Each point represents the estimate from one of 6 Markov chains after a given number of samples (log scale). The dotted line shows the average of the 6 individual estimates.

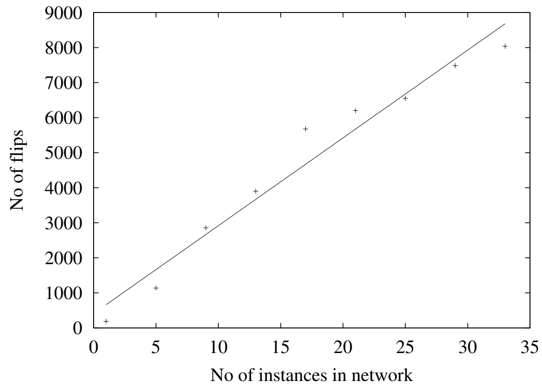

Figure 9: MCMC convergence as a function of network size. The y-axis measures the number of state transitions required to reach a preset convergence threshold for a standard diagnostic. The straight line is a regression fit to the data, which is averaged over 30 trials.

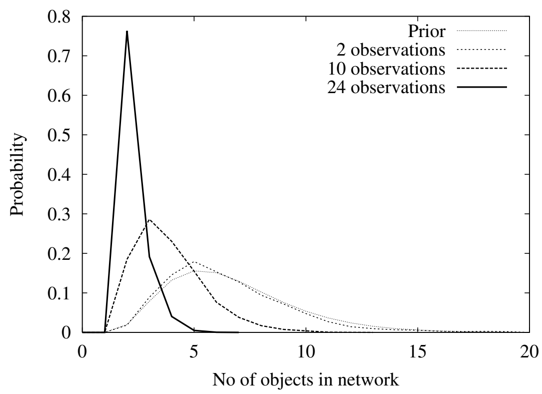

Figure 10: Distributions over the number of objects in the network. The prior distribution is shown; note that it insists that there are at least two vehicles. Observations are then generated from exactly two vehicles. When given just two observations, the algorithm converges to a distribution close to the prior, as expected. As more observations are added, the result moves closer to the true value of two vehicles.

6 Conclusions and further work

We have proposed an algorithmic approach for inference in first-order probabilistic languages, based on Markov chain Monte Carlo. Our preliminary investigations suggest that the approach is very promising. We believe that it can significantly increase the expressive power of languages that can be considered "practical" for knowledge representation and reasoning under uncertainty. For example, the lazy exploration approach should make it possible to handle infinite recursive models, including temporal models, with somewhat greater generality than so far envisaged [Koller and Pfeffer, 2000].

The Metropolis–Hastings algorithm allows for a wide variety of proposal distributions. In particular, *intelligent proposals* can be used to focus computation. One can easily construct *data-driven* and *query-driven* proposals, analogous to forward and backward chaining in logical systems, that essentially result in "activation" spreading outwards from query and evidence variables. These concepts can be subsumed by a general mechanism for proposing transitions based on the expected value of computation [Russell and Wefald, 1991]. We can also insert arbitrary domain-specific knowledge into the proposal mechanism—for example, proposing candidate advisors based on the student's research interest. Such knowledge-based proposals, which result in faster convergence, can also be *learned* via so-called *adaptive proposals*—that is, allowing the proposal distribution q to change over time based on the algorithm's experience in generating samples and computing acceptances.

Clearly, we have just scratched the surface of this topic. A large effort is needed to apply first-order probabilistic languages to real problems, in order to identify useful language features, common representation structures, and their effect on inference. Having a flexible and general—albeit sometimes slow—MCMC inference engine should help with this task. Finally, we also need to develop complexity results for approximate inference, using tools such as those provided by [Jerrum and Sinclair, 1997].

References

[Gelman, 1996] Andrew Gelman. Inference and monitoring convergence. In W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, editors, *Markov Chain Monte Carlo in Practice*, pages 131–143. Chapman and Hall, London, 1996.

[Gilks *et al.*, 1996] W.R. Gilks, S. Richardson, and D.J. Spiegelhalter, editors. *Markov chain Monte Carlo in practice*. Chapman and Hall, London, 1996.

[Halpern, 1990] J. Y. Halpern. An analysis of first-order logics of probability. *Artificial Intelligence*, 46(3):311–350, 1990.

[Jerrum and Sinclair, 1997] M. Jerrum and A. Sinclair. The Markov chain Monte Carlo method. In D. S. Hochbaum, editor, *Approximation Algorithms for NP-hard Problems*. PWS Publishing, Boston, 1997.

[Koller and Pfeffer, 1998] D. Koller and A. Pfeffer. Probabilistic frame-based systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, Madison, Wisconsin, July 1998. AAAI Press.

[Koller and Pfeffer, 2000] D. Koller and A. Pfeffer. Semantics and inference for recursive probability models. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-00)*, Austin, Texas, July 2000. AAAI Press.

[Pasula *et al.*, 1999] Hanna Pasula, Stuart Russell, Michael Ostland, and Ya'acov Ritov. Tracking many objects with many sensors. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, Stockholm, Sweden, August 1999. Morgan Kaufmann.

[Pearl, 1988] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, California, 1988.

[Pfeffer *et al.*, 1999] A. Pfeffer, D. Koller, B. Milch, and K.T. Takusagawa. SPOOK: A system for probabilistic object-oriented knowledge representation. In *Uncertainty in Artificial Intelligence: Proceedings of the Fifteenth Conference*, Stockholm, August 1999. Morgan Kaufmann.

[Pfeffer, 2000] Avrom J. Pfeffer. *Probabilistic Reasoning for Complex Systems*. PhD thesis, Stanford University, Stanford, California, 2000.

[Russell and Wefald, 1991] Stuart J. Russell and Eric H. Wefald. *Do the Right Thing: Studies in Limited Rationality*. MIT Press, Cambridge, Massachusetts, 1991.

[Russell, 1999] Stuart Russell. Expressive probability models in science. In *Proc. of the 2nd Int'l Conf. on Discovery Science*, Tokyo, Japan, December 1999. Springer Verlag.

[Sato and Kameya, 1997] T. Sato and Y. Kameya. PRISM: A symbolic-statistical modeling language. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97)*, pages 1330–1335, Nagoya, Japan, August 1997. Morgan Kaufmann.

[Wellman *et al.*, 1992] M. P. Wellman, J. S. Breese, and R. P. Goldman. From knowledge bases to decision models. *Knowledge Engineering Review*, 7(1):35–53, March 1992.

[Zhang and Poole, 1996] N.L. Zhang and D. Poole. Exploiting causal independence in Bayesian network inference. *Journal of Artificial Intelligence Research*, 5:301–328, 1996.