

---

# Extending Bayesian Networks to the Open-Universe Case

BRIAN MILCH AND STUART RUSSELL

## 1 Introduction

One of Judea Pearl’s now-classic examples of a Bayesian network involves a home alarm system that may be set off by a burglary or an earthquake, and two neighbors who may call the homeowner if they hear the alarm. Like most scenarios modeled with BNs, this example involves a known set of objects (one house, one alarm, and two neighbors) with known relations between them (the alarm is triggered by events that affect this house; the neighbors can hear this alarm). These objects and relations determine the relevant random variables and their dependencies, which are then represented by nodes and edges in the BN.

In many real-world scenarios, however, the relevant objects and relations are initially unknown. For instance, suppose we have a set of ASCII strings containing irregularly formatted and possibly erroneous academic citations extracted from online documents, and we wish to make a list of the distinct publications that are referred to, with correct author names and titles. In this case, the publications, authors, venues, and so on are not known in advance, nor is the mapping between publications and citations. The same challenge of making inferences about unknown objects is called *coreference resolution* in natural language processing, *data association* in multitarget tracking, and *record linkage* in database systems. The issue is actually much more widespread than this short list suggests; it arises in any data interpretation problem in which objects or events come without unique identifiers.

In this chapter, we show how the Bayesian network (BN) formalism that Judea Pearl pioneered has been extended to handle such scenarios. The key contribution on which we build is the use of acyclic directed graphs of local conditional distributions to generate well-defined, global probability distributions. We begin with a review of *relational probability models* (RPMs), which specify how to construct a BN for a given set of objects and relations. We then describe *open-universe probability models*, or OUPMs, which represent uncertainty about what objects exist. OUPMs may not boil down to finite, acyclic BNs; we present results from Milch [2006] showing how to extend the factorization and conditional independence semantics of BNs to models that are only *context-specifically* finite and acyclic. Finally, we discuss how Markov chain Monte Carlo (MCMC) methods can be used to perform approximate inference on OUPMs and briefly describe some applications.

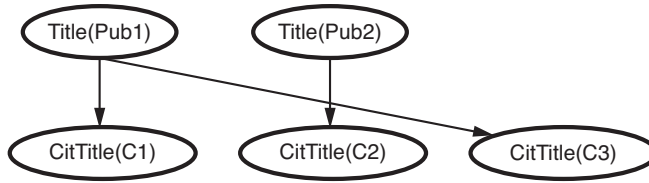


Figure 1. A BN for a bibliography scenario where we know that citations Cit1 and Cit3 refer to Pub1, while citation Cit2 refers to Pub2.

$$\text{Title}(p) \sim \text{TitlePrior}()$$

$$\text{CitTitle}(c) \sim \text{TitleEditCPD}(\text{Title}(\text{PubCited}(c)))$$

Figure 2. Dependency statements for a bibliography scenario, where  $p$  ranges over publications and  $c$  ranges over citations. This model assumes that the PubCited function and the sets of publications and citations are known.

## 2 Relational probability models

Suppose we are interested in inferring the true titles of publications given some observed citations, and we know the set of publications and the mapping from citations to publications. Assuming we have a prior distribution for true title strings (perhaps a word  $n$ -gram model) and a conditional probability distribution (CPD) for citation titles given true titles, we can construct a BN for this scenario, as shown in Figure 1.

### 2.1 The RPM formalism

A relational probability model represents such a BN compactly using *dependency statements* (see Figure 2), which specify the CPDs and parent sets for whole classes of variables at once. In this chapter, we will not specify any particular syntax for dependency statements, although we use a syntax based loosely on BLOG [Milch et al. 2005]. The important point is that dependencies are specified via relations among objects. For example, the dependency statement for CitTitle in Figure 2 specifies that each CitTitle( $c$ ) variable depends (according to the conditional distribution TitleEditCPD that describes how titles may be erroneously transcribed) on Title(PubCited( $c$ ))—that is, on the true title of the publication that  $c$  cites. The PubCited relation is nonrandom, and thus forms part of the known *relational skeleton* of the RPM. In this case, the skeleton also includes the sets of citations and publications.

Formally, it is convenient to think of an RPM  $M$  as defining a probability distribution over a set of model structures of a typed first-order logical language. These structures are called the *possible worlds* of  $M$  and denoted  $\Omega_M$ . The function sym-

bols of the logical language (including constant and predicate symbols) are divided into a set of *nonrandom* function symbols whose interpretations are specified by the relational skeleton, and a set of *random* function symbols whose interpretations vary between possible worlds. An RPM includes one dependency statement for each random function symbol.

Each RPM  $M$  defines a set of *basic random variables*  $\mathbf{V}_M$ , one for each application of a random function to a tuple of arguments. We will write  $X(\omega)$  for the value of a random variable  $X$  in world  $\omega$ . If  $X$  represents the value of the random function  $f$  on some arguments, then the dependency statement for  $f$  defines a parent set and CPD for  $X$ . The parent set for  $X$ , denoted  $\text{Pa}(X)$ , is the set of basic variables that are needed to evaluate the expressions in the dependency statement in any possible world. For instance, if we know that  $\text{PubCited}(\text{Cit1}) = \text{Pub1}$ , then the dependency statement in Figure 2 yields the single parent  $\text{Title}(\text{Pub1})$  for the variable  $\text{CitTitle}(\text{Cit1})$ . The CPD for a basic variable  $X$  is a function  $\varphi_X(x, \mathbf{pa})$  that defines a conditional probability distribution over values  $x$  of  $X$  given each instantiation  $\mathbf{pa}$  of  $\text{Pa}(X)$ . We obtain this CPD by evaluating the expressions in the dependency statement (such as  $\text{Title}(\text{PubCited}(\text{Cit1}))$ ) and passing them to an *elementary distribution function* such as  $\text{TitleEditCPD}$ .

Thus, an RPM defines a BN over its basic random variables. If this BN is acyclic, it defines a joint distribution for the basic RVs. Since there is a one-to-one correspondence between full instantiations of  $\mathbf{V}_M$  and worlds in  $\Omega_M$ , this BN also gives us a probability measure over  $\Omega_M$ . We define this to be the probability measure represented by the RPM.

## 2.2 Relational uncertainty

This formalism also allows us to model cases of *relational uncertainty*, such as a scenario where the mapping from citations to publications is unknown. We can handle this by making  $\text{PubCited}$  a random function and giving it a dependency statement such as:

$$\text{PubCited}(c) \sim \text{Uniform}(\{\text{Pub } p\}) .$$

This statement says that each citation refers to a publication chosen uniformly at random from the set of all publications  $p$ . The dependency statement for  $\text{CitTitle}$  in Figure 2 now represents a *context-specific* dependency: for a given citation  $C_i$ , the  $\text{Title}(p)$  variable that  $\text{CitTitle}(C_i)$  depends on varies from world to world.

In the BN defined by this model, shown in Figure 3, the parents of each  $\text{CitTitle}(c)$  variable include all variables that might be needed to evaluate the dependency statement for  $\text{CitTitle}(c)$  in any possible world. This includes  $\text{PubCited}(c)$  and all the  $\text{Title}(p)$  variables. The CPD in the BN is a multiplexer that conditions on the appropriate  $\text{Title}(p)$  variable for each value of  $\text{PubCited}(c)$ . If the BN constructed this way is still finite and acyclic, the usual BN semantics hold.

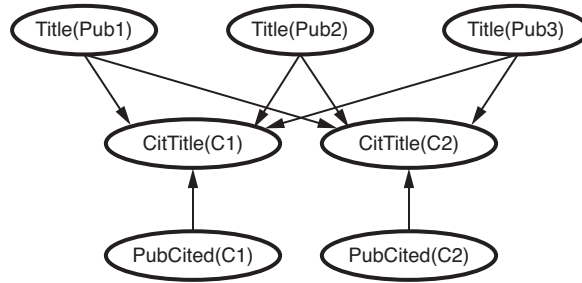


Figure 3. A BN arising from an RPM with relational uncertainty.

### 2.3 Names, objects, and identity uncertainty

We said earlier that the function symbols of an RPM include the constants and predicate symbols. For predicates, this simply means that a predicate can be thought of as a Boolean function that returns *true* or *false* for each tuple of arguments. The constants, on the other hand, are 0-ary functions that refer to objects. In most RPM languages, all constants are nonrandom and assumed to refer to distinct objects—the *unique names* assumption for constants. With this assumption, there is no need to distinguish between constant symbols and the objects they refer to, which is why we are able to name the basic random variables  $\text{Title}(\text{Pub1})$ ,  $\text{Title}(\text{Pub2})$  and so on, even though, strictly speaking, the arguments should be objects in the domain rather than constant symbols.

If the RPM language allows constants to be random functions, then the equivalent BN will include a node for each such constant. For example, suppose that Milch asks Russell to “fix the typo in the Pfeffer citation.” Russell’s mental software may already have formed nonrandom constant symbols  $C1$ ,  $C2$ , and so on for all the citations at the end of the chapter, and these are in one-to-one correspondence with all the objects in this particular universe. It may then form a new constant symbol  $\text{ThePfefferCitation}$ , which co-refers with one of these. Because there is more than one citation to a work by Pfeffer, there is *identity uncertainty* concerning which citation object the new symbol refers to. Identity uncertainty is a degenerate form of relational uncertainty, but often has a quite distinct flavor.

## 3 Open-universe probability models

For all RPMs, even those with relational and identity uncertainty, the *objects* are known and are the same across all possible worlds. If the set of objects is unknown, however—e.g., if we don’t know the set of publications that exist and might be cited—then RPMs as we have described them do not suffice. Whereas an RPM can be seen as defining a generative process that chooses a value for each random function on each tuple of arguments, an *open-universe probability model* (OUPM) includes generative steps that add objects to the world. These steps set the values of *number variables*.

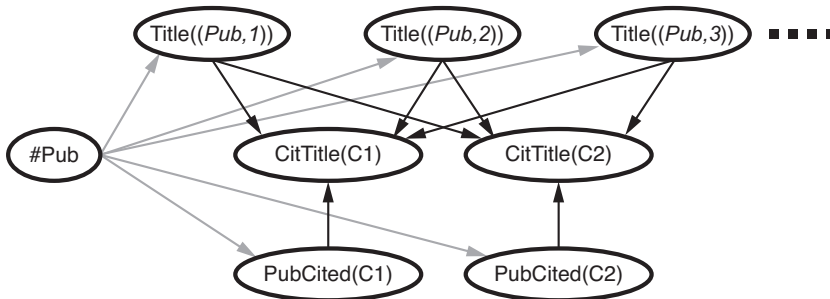


Figure 4. A BN that defines a probability distribution over worlds with unbounded numbers of publications.

### 3.1 Number variables

For the bibliography example, we introduce just one number variable, defining the total number of publications. There is no reason to place any *a priori* upper bound on the number of publications; we might be interested in asking how many publications there are for which we have found no citations (this question becomes more well-defined and pressing if we ask, say, how many aircraft are in an area but have not generated a blip on our radar screens). Thus, this number variable may have a distribution that assigns positive probability to all natural numbers.

We can specify the conditional probability distribution for a number variable using a dependency statement. In our bibliography example, we might use a very simple statement:

$$\#Pub \sim \text{NumPubsPrior}().$$

Number variables can also depend on other variables; we will consider an example of this below.

In the RPM where we had a fixed set of publications, the relational skeleton specified a constant symbol such as `Pub1` for each publication. In an OUPM where the set of publications is unknown, it does not make sense for the language to include such constant symbols. The possible worlds contain publication objects—which will assume are pairs  $\langle Pub, 1 \rangle$ ,  $\langle Pub, 2 \rangle$ , etc.—but now they are not necessarily in one-to-one correspondence with any constant symbols.

The set of basic variables now includes the number variable `#Pub` itself, and variables for the application of each random function to all arguments that exist in any possible world. Figure 4 shows the BN over these variables. Note that we have an infinite sequence of `Title` variables: if we had a finite number, our BN would not define probabilities for worlds with more than that number of publications. We stipulate that if a basic variable has an object  $o$  as an argument, then in worlds where  $o$  does not exist, the variable takes on the special value `null`. Thus, `#Pub` is a parent of each `Title( $p$ )` variable, determining whether that variable takes the value `null` or not. The set of publications available for selection in the dependency

```

#Researcher ~ NumResearchersPrior()
Position( $r$ ) ~ [0.7 : GradStudent, 0.2 : PostDoc, 0.1 : Prof]
#Pub(FirstAuthor =  $r$ ) ~ NumPubsCPD(Position( $r$ ))

```

Figure 5. Dependency statements that augment our bibliography model to represent a set of researchers, the position of each researcher, and the set of first-authored publications by each researcher.

statement for `PubCited( $c$ )` also depends on the number variable.

Objects of a given type may be generated by more than one event in the generative process. For instance, if we include objects of type `Researcher` in our model and add a function `FirstAuthor( $p$ )` that maps publications to researchers, we may wish to say that each researcher independently generates a crop of papers on which he or she is the first author. The number of papers generated may depend on the researcher’s position (graduate student, professor, etc.). We now get a family of number variables `#Pub(FirstAuthor =  $r$ )`, where  $r$  ranges over researchers. The number of researchers may itself be governed by a number variable. Figure 5 shows the dependency statements for these aspects of the scenario.

In this model, `FirstAuthor( $p$ )` is an *origin function*: in the generative model underlying the OUPM, it is set when  $p$  is created, not in a separate generative step. The values of origin functions on an object tell us which number variable governs that object’s existence; for example, if `FirstAuthor( $p$ )` is  $\langle \text{Researcher}, 5 \rangle$ , then `#Pub(FirstAuthor =  $\langle \text{Researcher}, 5 \rangle$ )` governs the existence of  $p$ . Origin functions can also be used in dependency statements, just like any other function: for instance, we might change the dependency statement for `PubCited( $c$ )` so that more significant publications are more likely to be cited, with the significance of a publication  $p$  being influenced by `Position(FirstAuthor( $p$ ))`.

In the scenario we have considered so far, each possible world contains finitely many `Researcher` and `Pub` objects. OUPMs can also accommodate infinite numbers of objects. For instance, we could define a model for academia where each researcher  $r$  generates a random number of new researchers  $r'$  such that `Advisor( $r'$ ) =  $r$` . Some possible worlds in this model may contain infinitely many researchers.

### 3.2 Possible worlds and basic random variables

In defining the semantics of RPMs, we said that a model  $M$  defines a BN over its basic random variables  $\mathbf{V}_M$ , and then we exploited the one-to-one correspondence between full instantiations of those variables and possible worlds. In an OUPM, however, there may be instantiations of the basic random variables that do not correspond to any possible world. An example in our bibliography scenario is an

instantiation where  $\#Pub = 100$ , but  $Title(p)$  takes on a non-null value for 200 publications.

To facilitate using the basic variables to define a probability measure over the possible worlds, we would like to have a one-to-one mapping between  $\Omega_M$  and a set of *achievable* instantiations of  $\mathbf{V}_M$ . This is straightforward in cases like our first OUPM, where there is only one number variable for each type of object. Then our semantics specifies that the *non-guaranteed objects* of each type—that is, the objects that exist in some possible worlds and not others, like the publications in our example—are pairs  $\langle Pub, 1 \rangle, \langle Pub, 2 \rangle, \dots$ . In each world, the set of non-guaranteed objects of each type that exist is required to be a prefix of this numbered sequence. Thus, if we know that  $\#Pub = 4$  in a world  $\omega$ , we know that the publications in  $\omega$  are  $\langle Pub, 1 \rangle$  through  $\langle Pub, 4 \rangle$ , not some other set of non-guaranteed objects.

Things are more complicated when we have multiple number variables for a type, as in our example with researchers generating publications. Given values for all the number variables of the form  $\#Pub(FirstAuthor = r)$ , we do not want there to be any uncertainty about *which* non-guaranteed objects have each *FirstAuthor* value. We can achieve this by letting the non-guaranteed objects be nested tuples that encode their generation history. For the publications with  $\langle Researcher, 5 \rangle$  as their first author, we use tuples

$$\begin{aligned} &\langle Pub, \langle FirstAuthor, \langle Researcher, 5 \rangle \rangle, 1 \rangle \\ &\langle Pub, \langle FirstAuthor, \langle Researcher, 5 \rangle \rangle, 2 \rangle \end{aligned}$$

and so on. As before, in each possible world, the set of tuples in each sequence must form a prefix of the sequence. This construction yields the following lemma.

LEMMA 1. *In any OUPM  $M$ , each complete instantiation of  $\mathbf{V}_M$  is consistent with at most one possible world in  $\Omega_M$ .*

Section 4.3 of Milch [2006] gives a more rigorous formulation and proof of this result. Given this lemma, the probability measure defined by an OUPM  $M$  on  $\Omega_M$  is well-defined if the OUPM specifies a joint probability distribution for  $\mathbf{V}_M$  that is concentrated on the set of achievable instantiations. Since the OUPM’s CPDs implicitly force a variable to take the value null when any of its arguments do not exist, any distribution consistent with the CPDs will indeed put probability one on achievable instantiations.

Informally, the probability distribution for the basic random variables can be defined by a generative process that builds up an instantiation step-by-step, sampling a value for each variable according to its dependency statement. In the next section, we show how this intuitive semantics can be formalized using an extended version of Bayesian networks.

## 4 Extending BN semantics

There are two equivalent ways of defining the probability distribution represented by a BN  $\mathbf{B}$ . The first is based on conditional independence statements; specifically,

```

#Pub ~ NumPubsPrior()

Title(p) ~ TitlePrior()

Date(c) ~ DatePrior()

SourceCopied(c) ~ [0.9 : null,
                  0.1 : Uniform({Citation c2 :
                                (PubCited(c2) = PubCited(c))
                                ∧ (Date(c2) < Date(c))})]

CitTitle(c) ~ if SourceCopied(c) = null
               then TitleEditCPD(Title(PubCited(c)))
               else TitleEditCPD(CitTitle(SourceCopied(c)))

```

Figure 6. Dependency statements for a model where each citation was written on some date, and a citation may copy the title from an earlier citation of the same publication rather than copying the publication title directly.

the directed local Markov property: each variable is conditionally independent of its non-descendants given its parents. The second is based on a product expression for the joint distribution; if  $\sigma$  is any instantiation of the full set of variables  $\mathbf{V}_{\mathbf{B}}$  in the BN, then

$$P(\sigma) = \prod_{X \in \mathbf{V}_{\mathbf{B}}} \varphi_X(\sigma[X], \sigma[\text{Pa}(X)]) .$$

The remarkable property of BNs is that if the graph is finite and acyclic, then there is guaranteed to be exactly one joint distribution that satisfies these conditions.

#### 4.1 Infinite sets of variables

Note that in the BN in Figure 4, the  $\text{CitTitle}(c)$  variables have infinitely many parents. The fact that the BN has infinitely many nodes means that we can no longer use the standard product-expression semantics for the BN, because the product of the CPDs for all variables is an infinite product, and will typically be zero for all values of the variables. We would like to specify probabilities for certain partial, finite instantiations of the variables that are sufficient to define the joint distribution. As noted by Kersting and DeRaedt [2001], if it is possible to number the nodes of the BN in topological order, then it suffices to specify the product expression for each finite prefix of this numbering. However, if a variable has infinitely many parents, then the BN has no topological numbering—if we try numbering the nodes in topological order, we will spend forever on  $X$ 's parents and never reach  $X$ .



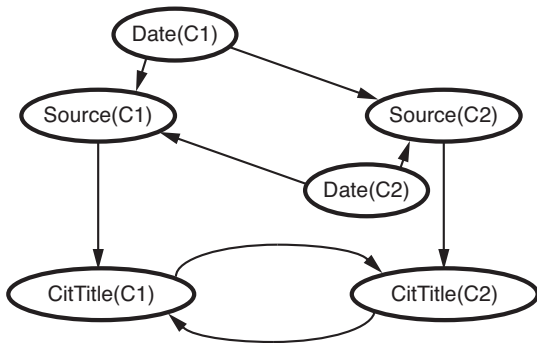


Figure 7. Part of the BN defined by the OUPM in Figure 6, for two citations.

#### 4.2 Cyclic sets of potential dependencies

In OUPMs and even RPMs with relational uncertainty, it is fairly easy to write dependency statements that define a cyclic BN. For instance, suppose that some citations are composed by copying another citation, and we do not know who copied whom. We can specify a model where each citation was written at some unknown date, and with probability 0.1, a citation copies an earlier citation to the same publication if one exists. Figure 6 shows the dependency statements for this model. (Note that **Date** here is the date the citation was written, i.e., the date of the citing paper, not the date of the paper being cited.)

The BN defined by this OUPM is cyclic, as shown in Figure 7. In general, a cyclic BN may fail to define a distribution; there may be no joint distribution with the specified CPDs. However, in this case, it is intuitively clear that that cannot happen. Since a citation can only copy another citation with a strictly earlier date, the dependencies that are active in any positive-probability world must be acyclic. There are actually elements of the possible world set  $\Omega_M$  where the dependencies are cyclic: these are worlds where, for some citation  $c$ ,  $\text{SourceCopied}(c)$  does not have an earlier date than  $c$ . But the CPD for  $\text{SourceCopied}$  forces these worlds to have probability zero.

The difficult aspect of semantics for this class of cyclic BNs is the directed local Markov property. It is no longer sufficient to assert that  $X$  is independent of its non-descendants in the full BN given its parents, because its set of non-descendants in the full BN may be too small. In this model, all the  $\text{CitTitle}$  nodes are descendants of each other, so the standard directed local Markov property would yield no assertions of conditional independence between them.

### 4.3 Partition-based semantics for OUPMs

We can solve these difficulties by exploiting the context-specific nature of dependencies in an OUPM, as revealed by dependency statements.<sup>1</sup> For each basic random variable  $X$ , an OUPM defines a partition  $\Lambda_X$  of  $\Omega_M$ . Two worlds are in the same block of this partition if evaluating the dependency statement for  $X$  in these two worlds yields the same conditional distribution for  $X$ . For instance, in our OUPM for the bibliography domain, the partition blocks for  $\text{CitTitle}(\text{Cit1})$  are sets of worlds that agree on the value of  $\text{Title}(\text{PubCited}(\text{Cit1}))$ . For each block  $\lambda \in \Lambda_X$ , the OUPM defines a probability distribution  $\varphi_X(x, \lambda)$  over values of  $X$ .

One defining property of the probability measure  $P_M$  specified by an OUPM  $M$  is that for each basic random variable  $X \in \mathbf{V}_M$  and each partition block  $\lambda \in \Lambda_X$ ,

$$P_M(X = x \mid \lambda) = \varphi_X(x, \lambda) \tag{1}$$

To fully define  $P_M$ , however, we need to make an assertion analogous to a BN’s factorization property or directed local Markov property. We will say that a partial instantiation  $\sigma$  *supports* a random variable  $X$  if there is some block  $\lambda \in \Lambda_X$  such that  $\sigma \subseteq \lambda$ . An instantiation that supports  $X$  in an OUPM is analogous to an instantiation that assigns values to all the parents of  $X$  in a BN. We define an instantiation  $\sigma$  to be *self-supporting* if for each variable  $X \in \text{vars}(\sigma)$ , the restriction of  $\sigma$  to  $\text{vars}(\sigma) \setminus \{X\}$  (denoted  $\sigma_{-X}$ ) supports  $X$ . We can now state a factorization property for OUPMs.

PROPERTY 2 (Factorization property for an OUPM  $M$ ). For each finite, self-supporting instantiation  $\sigma$  on  $\mathbf{V}_M$ ,

$$P_M(\sigma) = \prod_{X \in \text{vars}(\sigma)} \varphi_X(\sigma[X], \lambda_X(\sigma_{-X}))$$

where  $\lambda_X(\sigma_{-X})$  is the partition block in  $\Lambda_X$  that has  $\sigma_{-X}$  as a subset.

We can also define an analogue of the directed local Markov property for OUPMs. Recall that in the BN case, the directed local Markov property asserts that  $X$  is conditionally independent of every subset of its non-descendants given  $\text{Pa}(X)$ . In fact, it turns out to be sufficient to make this assertion for only a special class of non-descendant subsets, namely those that are *ancestral* (closed under the parent relation). Any ancestral set of variables that does not contain  $X$  contains only non-descendants of  $X$ . So in the BN case, we can reformulate the directed local Markov property to assert that given  $\text{Pa}(X)$ ,  $X$  is conditionally independent of any ancestral set of variables that does not contain  $X$ .

In OUPMs, the equivalent of a variable set that is closed under the parent relation is a self-supporting instantiation. We can formulate the directed local Markov property for an OUPM  $M$  as follows:

---

<sup>1</sup>We will assume all random variables are discrete in this treatment, but the ideas can be extended to the continuous case.

PROPERTY 3 (Directed local Markov property for an OUPM  $M$ ). For each basic random variable  $X \in \mathbf{V}_M$ , each block  $\lambda \in \Lambda_X$ , and each self-supporting instantiation  $\sigma$  on  $\mathbf{V}_M$  such that  $X \notin \text{vars}(\sigma)$ ,  $X$  is conditionally independent of  $\sigma$  given  $\lambda$ .

Under what conditions is there a unique probability measure  $P_M$  on  $\Omega_M$  that satisfies Properties 2 and 3? In the BN case, it suffices for the graph to admit a topological numbering. We can define a similar notion that is specific to individual worlds: a *supportive numbering* for a world  $\omega \in \Omega_M$  is a numbering  $X_0, X_1, \dots$  of  $\mathbf{V}_M$  such that for each natural number  $n$ , the instantiation  $(X_0(\omega), \dots, X_{n-1}(\omega))$  supports  $X_n$ .

THEOREM 4. *Let  $M$  be an OUPM such that for every world  $\omega \in \Omega_M$ , either:*

- $\omega$  has a supportive numbering, or
- for some basic random variable  $X \in \mathbf{V}_M$ ,  $\varphi_X(X(\omega), \lambda_X(\omega)) = 0$ .

*Then there is exactly one probability measure on  $\Omega_M$  satisfying the factorization property (Property 2), and it is also the unique probability measure that satisfies both Equation 1 and the directed local Markov property (Property 3).*

This theorem follows from Lemma 1 and results proved in Section 3.4 of Milch [2006]. Note that the theorem does not require supportive numberings for worlds that are *directly disallowed*—that is, those that are forced to have probability zero by the CPD for some variable.

In our basic bibliography scenario with unknown publications, we can construct a supportive numbering for each possible world  $\omega$  by taking first the number variable #Pub, then the PubCited( $c$ ) variables, then the Title( $p$ ) variables for the publications that serve as values of PubCited( $c$ ) variables in  $\omega$ , then the CitTitle( $c$ ) variables, and finally the infinitely many Title( $p$ ) variables for publications that are uncited or do not exist in  $\omega$ . For the scenario where citation titles can be copied from earlier citations, we have to add the Date( $c$ ) variables and then the SourceCopied( $c$ ) variables before the CitTitle( $c$ ) variables. We order the CitTitle( $c$ ) variables in a way that is consistent with Date( $c$ ). This procedure yields a supportive numbering in all worlds except those where  $\exists c \text{ Date}(\text{SourceCopied}(c)) \geq \text{Date}(c)$ , but such worlds are directly disallowed by the CPD for SourceCopied( $c$ ).

#### 4.4 Representing OUPMs as contingent Bayesian networks

The semantics we have given for OUPMs so far does not make reference to any graph. But we can also view an OUPM as defining a *contingent Bayesian network* (CBN) [Milch et al. 2005], which is a BN where each edge is labeled with an event. The event indicates when the edge is active, in a sense we will soon make precise. Figures 8 and 9 show CBNs corresponding to the infinite BN in Figure 4 and the cyclic BN in Figure 7, respectively.

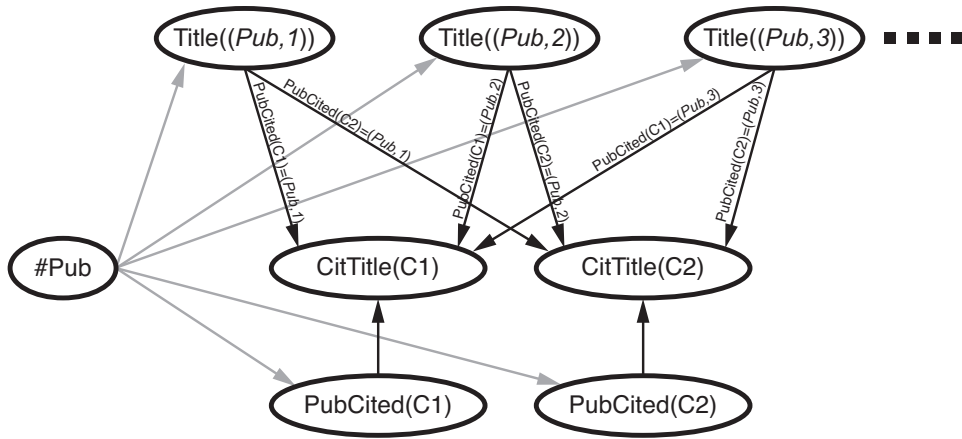


Figure 8. A contingent BN for the bibliography scenario with unknown objects.

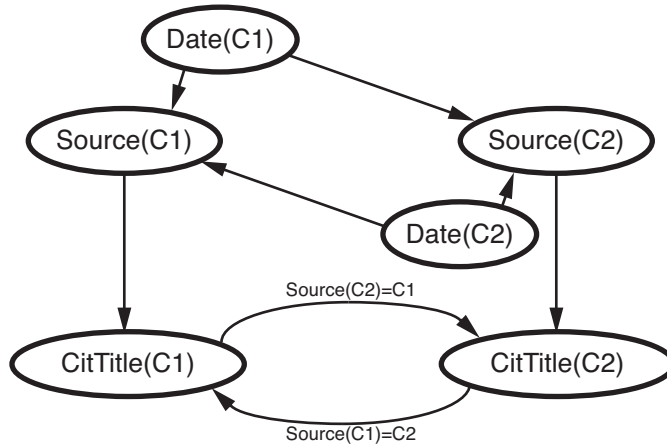


Figure 9. Part of a contingent BN for the OUPM in Figure 6.

A CBN can be viewed as a partition-based model where the partition  $\Lambda_X$  for each random variable  $X$  is defined by a decision tree. The internal nodes in this decision tree are labeled with random variables; the edges are labeled with variable values; and the leaves specify conditional probability distributions for  $X$ . The blocks in  $\Lambda_X$  correspond to the leaves in this tree (we assume the tree has no infinite paths, so the leaves cover all possible worlds). The restriction to decision trees allows us to define a notion of a parent being active in a particular world: if we walk along  $X$ 's tree from the root, following edges consistent with a given world  $\omega$ , then the random variables on the nodes we visit are the active parents of  $X$  in  $\omega$ . The label on an edge  $W \rightarrow X$  in a CBN is the event consisting of those worlds where  $W$  is an active parent of  $X$ . (In diagrams, we omit the trivial label  $A = \Omega_M$ , which indicates that the dependency is always active.)

The abstract notions of a self-supporting instantiation and a supportive numbering have simple graphical analogues in a CBN. We will use  $\mathbf{B}^\sigma$  to denote the BN obtained from a CBN  $\mathbf{B}$  by keeping only those edges whose conditions are entailed by  $\sigma$ . An instantiation  $\sigma$  supports a variable  $X$  if and only if all the parents of  $X$  in  $\mathbf{B}^\sigma$  are in  $\text{vars}(\sigma)$ , and it is self-supporting if and only if  $\text{vars}(\sigma)$  is an ancestral set in  $\mathbf{B}^\sigma$ . A supportive numbering for a world  $\omega$  is a topological numbering of the BN  $\mathbf{B}^\omega$  obtained by keeping only those edges whose conditions are satisfied by  $\omega$ . Thus, the well-definedness condition in Theorem 4 can be stated for CBNs as follows: for each world  $\omega \in \Omega_M$  that is not directly disallowed,  $\mathbf{B}^\omega$  must have a topological numbering.

Not all partitions can be represented exactly as the leaves of a decision tree, so there are sets of context-specific independence properties that can be captured by OUPMs and not CBNs. However, when we perform inference on an OUPM, we typically use a function that evaluates the dependency statement for each variable, looking up the values of other random variables in a given world (or partial instantiation) as needed. For example, a function evaluating the dependency statement for `CitTitle(Cit1)` will always access `PubCited(Cit1)`, and then it will access a particular `Title` variable depending on the value of the `PubCited` variable. This evaluation process implicitly defines a decision tree; the order of splits in the tree depends on the evaluation order used. When we discuss inference for OUPMs, we will assume that we are operating on the CBN implicitly defined by some evaluation function.

## 5 Inference

Given an OUPM, we would like to be able to compute the probability of a query event  $Q$  given an evidence event  $E$ . For example,  $Q$  could be the event that `PubCited(Cit1) = PubCited(Cit2)` and  $E$  could be the event that `CitTitle(Cit1) = "Learning Probabilistic Relational Models"` and `CitTitle(Cit2) = "Learning Probabilistic Relation Models"`. The ideas we present can be extended to other tasks such as computing the posterior distribution of a random variable, or finding the *maximum a posteriori* (MAP) assignment of values to a set of random variables.

### 5.1 MCMC over partial worlds

Sampling-based or Monte Carlo inference algorithms are well-suited for OUPMs because each sample specifies what objects exist and what relations hold among them. We focus on Markov chain Monte Carlo (MCMC), where we simulate a Markov chain over possible worlds consistent with the evidence  $E$ , such that the stationary distribution of the chain is the posterior distribution over worlds given  $E$ . Such a chain can be constructed using the Metropolis–Hastings method, where we use an arbitrary *proposal distribution*  $q(\omega'|\omega_t)$ , but accept or reject each proposal based on the relative probabilities of  $\omega'$  and  $\omega_t$ .

Specifically, at each step  $t$  in our Markov chain, we sample  $\omega'$  from  $q(\omega'|\omega_t)$  and

then compute the *acceptance probability*:

$$\alpha = \min \left( 1, \frac{P_M(\omega')q(\omega_t|\omega')}{P_M(\omega_t)q(\omega'|\omega_t)} \right).$$

With probability  $\alpha$ , we accept the proposal and let  $\omega_{t+1} = \omega'$ ; otherwise we reject the proposal and let  $\omega_{t+1} = \omega_t$ .

The difficulty in OUPMs is that each world may be very large. For instance, if we have a world where  $\#Pub = 1000$ , but only 100 publications are referred to by our observed citations, then the world must also specify the titles of the 900 unobserved publications. Sampling values for these 900 Title variables and computing their probabilities will slow down our algorithm unnecessarily. In scenarios where some possible worlds have infinitely many objects, specifying a possible world completely may be impossible.

Thus, we would like to run MCMC over *partial* descriptions that specify values only for certain random variables. The set of instantiated variables may vary from world to world. Since a partial instantiation  $\sigma$  defines an event (the set of worlds that are consistent with it), a Markov chain over partial instantiations can be viewed as a chain over events. Thus, we use the acceptance probability:

$$\alpha = \min \left( 1, \frac{P_M(\sigma')q(\sigma_t|\sigma')}{P_M(\sigma_t)q(\sigma'|\sigma_t)} \right)$$

where  $P_M(\sigma)$  is the probability of the event  $\sigma$ . As long as the set  $\Sigma$  of partial instantiations that can be returned by  $q$  forms a partition of  $E$ , and each partial instantiation is specific enough to determine whether  $Q$  is true, we can estimate  $P(Q|E)$  using a Markov chain on  $\Sigma$  with stationary distribution proportional to  $P_M(\sigma)$  [Milch and Russell 2006].

In general, computing the probability  $P_M(\sigma)$  involves summing over all the variables not instantiated in  $\sigma$ —which is precisely what we want to avoid by using a Monte Carlo inference algorithm. Fortunately, if each instantiation in  $\Sigma$  is self-supporting, we can compute its probability using the product expression from Property 2. Thus, our partial worlds are self-supporting instantiations that include the query and evidence variables. We also make sure to use *minimal* instantiations satisfying this condition—that is, instantiations that would cease to be self-supporting if we removed any non-query, non-evidence variable. It can be shown that in a CBN, such minimal self-supporting instantiations are mutually exclusive. So if our set of partial worlds  $\Sigma$  covers all of  $E$ , we are guaranteed to have a partition of  $E$ , as required. An example of a partial world in our bibliography scenario is:

$\#Pub = 50$ ,  $CitTitle(Cit1) = \text{“Calculus”}$ ,  $CitTitle(Cit2) = \text{“Intro to Calculus”}$ ,  
 $PubCited(Cit1) = \langle Pub, 17 \rangle$ ,  $PubCited(Cit2) = \langle Pub, 31 \rangle$ ,  
 $Title(\langle Pub, 17 \rangle) = \text{“Calculus”}$ ,  $Title(\langle Pub, 31 \rangle) = \text{“Intro to Calculus”}$

## 5.2 Abstract partial worlds

In the partial instantiation above, we specify the tuple representation of each publication, as in  $\text{PubCited}(\text{Cit1}) = \langle \text{Pub}, 17 \rangle$ . If partial worlds are represented this way, then the code that implements the proposal distribution has to choose numbers for any new objects it adds, keep track of the probability of its choices, and compute the probability of the reverse proposal. Some kinds of moves are impossible unless the proposer renumbers the objects: for instance, the total number of publications cannot be decreased from 1000 to 900 when publication 941 is in use.

To simplify the proposal distribution, we can use partial worlds that abstract away the identities of objects using existential quantifiers:

$\exists$  distinct  $x, y$   
 $\#\text{Pub} = 50, \text{CitTitle}(\text{Cit1}) = \text{“Calculus”}, \text{CitTitle}(\text{Cit2}) = \text{“Intro to Calculus”},$   
 $\text{PubCited}(\text{Cit1}) = x, \text{PubCited}(\text{Cit2}) = y,$   
 $\text{Title}(x) = \text{“Calculus”}, \text{Title}(y) = \text{“Intro to Calculus”}$

The probability of the event corresponding to an abstract partial world depends on the number of ways the logical variables can be mapped to distinct objects. For simplicity, we will assume that there is only one number variable for each type. If an abstract partial world  $\sigma$  uses logical variables for a type  $\tau$ , we require it to instantiate the number variable for that type. We also require that for each logical variable  $x$ , there is a distinct ground term  $t_x$  such that  $\sigma$  implies  $t_x = x$ ; this ensures that each mapping from logical variables to tuple representations yields a distinct possible world. Let  $T$  be the set of types of logical variables in  $\sigma$ , and for each type  $\tau \in T$ , let  $n_\tau$  be the value of  $\#\tau$  in  $\sigma$  and  $\ell_\tau$  be the number of logical variables of type  $\tau$  in  $\sigma$ . Then we have:

$$P(\sigma) = P_c(\sigma) \prod_{\tau \in T} \frac{n_\tau!}{(n_\tau - \ell_\tau)!}$$

where  $P_c(\sigma)$  is the probability of any one of the “concrete” instantiations obtained by substituting distinct tuple representations for the logical variables in  $\sigma$ .

## 5.3 Locality of computation

Given a current instantiation  $\sigma_t$  and a proposed instantiation  $\sigma'$ , computing the acceptance probability involves computing the ratio:

$$\frac{P_M(\sigma')q(\sigma_t|\sigma')}{P_M(\sigma_t)q(\sigma'|\sigma_t)} = \frac{q(\sigma_t|\sigma') \prod_{X \in \text{vars}(\sigma')} \varphi_X(\sigma'[X], \sigma'[\text{Pa}_{\sigma'}(X)])}{q(\sigma'|\sigma_t) \prod_{X \in \text{vars}(\sigma_t)} \varphi_X(\sigma_t[X], \sigma_t[\text{Pa}_{\sigma_t}(X)])}$$

where  $\text{Pa}_\sigma(X)$  is the set of parents of  $X$  whose edge conditions are entailed by  $\sigma$ . This expression is daunting, because even though the instantiations  $\sigma_t$  and  $\sigma'$  are only partial descriptions of possible worlds, they may still assign values to large sets of random variables — and the number of instantiated variables grows at least linearly with the number of observations we have. Since we may want to run

millions of MCMC steps, having each step take time proportional to the number of observations would make inference prohibitively expensive.

Fortunately, with most proposal distributions used in practice, each step changes the values of only a small set of random variables. Furthermore, if the edges that are active in any given possible world are fairly sparse, then  $\sigma[\text{Pa}_{\sigma'}(X)]$  will also be the same as  $\sigma_t[\text{Pa}_{\sigma_t}(X)]$  for many variables  $X$ . Thus, many factors will cancel out in the ratio above.

We need to compute the “new” and “old” probability factors for a variable  $X$  only if either  $\sigma'[X] \neq \sigma_t[X]$ , or there is some active parent  $W \in \text{Pa}_{\sigma_t}(X)$  such that  $\sigma'[W] \neq \sigma_t[W]$ . (We take these inequalities to include the case where  $\sigma'$  assigns a value to the variable and  $\sigma_t$  does not, or vice versa.) Note that it is not possible for  $\text{Pa}_{\sigma'}(X)$  to be different from  $\text{Pa}_{\sigma_t}(X)$  unless one of the “old” active parents in  $\text{Pa}_{\sigma_t}(X)$  has changed: given that  $\sigma_t$  is a self-supporting instantiation, the values of  $X$ ’s instantiated parents in  $\sigma_t$  determine the truth values of the conditions on all the edges into  $X$ , so the set of active edges into  $X$  cannot change unless one of these parent variables changes.

This fact is exploited in the BLOG system [Milch and Russell 2006] to efficiently detect which probability factors need to be computed for a given proposal. The system maintains a graph of the edges that are active in the current instantiation  $\sigma_t$ . The proposer provides a list of the variables that are changed in  $\sigma'$ , and the system follows the active edges in the graph to identify the children of these variables, whose probability factors also need to be computed. Thus, the graphical locality that is central to many other BN inference algorithms also plays a role in MCMC over relational structures.

## 6 Related work

The connection between probability and first-order languages was first studied by Carnap [1950]. Gaifman [1964] and Scott and Krauss [1966] defined a formal semantics whereby probabilities could be associated with first-order sentences and for which models were probability measures on possible worlds. Within AI, this idea was developed for propositional logic by Nilsson [1986] and for first-order logic by Halpern [1990]. The basic idea is that each sentence *constrains* the distribution over possible worlds; one sentence entails another if it expresses a stronger constraint. For example, the sentence  $\forall x P(\text{Hungry}(x)) > 0.2$  rules out distributions in which any object is hungry with probability less than 0.2; thus, it entails the sentence  $\forall x P(\text{Hungry}(x)) > 0.1$ . Bacchus [1990] investigated knowledge representation issues in such languages. It turns out that writing a *consistent* set of sentences in these languages is quite difficult and constructing a unique probability model nearly impossible unless one adopts the representational approach of Bayesian networks by writing suitable sentences about conditional probabilities.

The impetus for the next phase of work came from researchers working with BNs directly. Rather than laboriously constructing large BNs by hand, they built



them by composing and instantiating “templates” with logical variables that described local causal models associated with objects [Breese 1992; Wellman et al. 1992]. The most important such language was BUGS (Bayesian inference Using Gibbs Sampling) [Gilks et al. 1994], which combined Bayesian networks with the indexed-random-variable notation common in statistics. These languages inherited the key property of Bayesian networks: every well-formed knowledge base defines a unique, consistent probability model. Languages with well-defined semantics based on unique names and domain closure drew on the representational capabilities of logic programming [Poole 1993; Sato and Kameya 1997; Kersting and De Raedt 2001] and semantic networks [Koller and Pfeffer 1998; Pfeffer 2000]. Initially, inference in these models was performed on the equivalent Bayesian network. *Lifted* inference techniques borrow from first-order logic the idea of performing an inference once to cover an entire equivalence class of objects [Poole 2003; de Salvo Braz et al. 2007; Kisynski and Poole 2009]. MCMC over relational structures was introduced by Pasula and Russell [2001]. Getoor and Taskar [2007] collect many important papers on first-order probability models and their use in machine learning.

Probabilistic reasoning about identity uncertainty has two distinct origins. In statistics, the problem of *record linkage* arises when data records do not contain standard unique identifiers—for example, in financial, medical, census, and other data [Dunn 1946; Fellegi and Sunter 1969]. In control theory, the problem of *data association* arises in multitarget tracking when each detected signal does not identify the object that generated it [Sittler 1964]. For most of its history, work in symbolic AI assumed erroneously that sensors could supply sentences with unique identifiers for objects. The issue was studied in the context of language understanding by Charniak and Goldman [1993] and in the context of surveillance by Huang and Russell [1998] and Pasula *et al.* [1999]. Pasula *et al.* [2003] developed a complex generative model for authors, papers, and citation strings, involving both relational and identity uncertainty, and demonstrated high accuracy for citation information extraction. The first formally defined language for open-universe probability models was BLOG [Milch et al. 2005], from which the material in the current chapter was developed. Laskey [2008] describes another open-universe modeling language called multi-entity Bayesian networks.

Another important thread goes under the name of *probabilistic programming languages*, which include IBAL [Pfeffer 2007] and CHURCH [Goodman et al. 2008]. These languages represent first-order probability models using a programming language extended with a randomization primitive; any given “run” of a program can be seen as constructing a possible world, and the probability of that world is the probability of all runs that construct it.

The OUPMs we have described here bear some resemblance to probabilistic programs, since each dependency statement can be viewed as a program fragment for sampling a value for a child variable. However, expressions in dependency statements have different semantics from those in a probabilistic functional language

such as IBAL: if an expression such as `Title(Pub5)` is evaluated in several dependency statements in a given possible world, it returns the same value every time, whereas the value of an expression in IBAL is sampled independently each time it appears. The CHURCH language incorporates aspects of both approaches: it includes a *stochastic memoization* construct that lets the programmer designate certain expressions as having values that are sampled once and then reused. McAllester *et al.* [2008] define a probabilistic programming language that makes sources of random bits explicit and has a possible-worlds semantics similar to OUPMs.

This chapter has described generative, directed models. The combination of relational and first-order notations with (undirected) Markov networks is also interesting [Taskar et al. 2002; Richardson and Domingos 2006]. Undirected formalisms are convenient because there is no need to avoid cycles. On the other hand, an essential assumption underlying relational probability models is that one set of CPD parameters is appropriate for a wide range of relational structures. For instance, in our RPMs, the prior for a publication’s title does not depend on how many citations refer to it. But in an undirected model, adding more citations to a publication (and thus more potentials linking `Title(p)` to `CitTitle(c)` variables) will usually change the marginal on `Title(p)`, even when none of the `CitTitle(c)` values are observed. This suggests that all the potentials must be learned jointly on a training set with roughly the same distribution of relational structures as the test set; in the directed case, we are free to learn different CPDs from different data sources.

## 7 Discussion

This chapter has stressed the importance of unifying probability theory with first-order logic—particularly for cases with unknown objects—and has presented one possible approach based on open-universe probability models, or OUPMs. OUPMs draw on the key idea introduced into AI by Judea Pearl: generative probability models based on local conditional distributions. Whereas BNs generate worlds by assigning values to variables one at a time, relational models can assign values to a whole class of variables through a single dependency assertion, while OUPMs add object creation as one of the generative steps.

OUPMs appear to enable the straightforward representation of a wide range of situations. In addition to the citation model mentioned in this chapter (see Milch [2006] for full details), models have been written for multitarget tracking, plan recognition, sibyl attacks (a security threat in which a reputation system is compromised by individuals who create many fake identities), and detection of nuclear explosions using networks of seismic sensors [Russell and Vaidya 2009]. In each case, the model is essentially a transliteration of the obvious English description of the generative process.

Inference, however, is another matter. The generic Metropolis–Hastings inference engine written for BLOG in 2006 is far too slow to support any of the applications described in the preceding paragraph. For the citation problem, Milch [2006] de-

scribes an application-specific proposal distribution for the generic M–H sampler that achieves speeds comparable to a completely hand-coded, application-specific inference engine. This approach is feasible in general but requires a significant coding effort by the user. Current efforts in the BLOG project are aimed instead at improving the generic engine: implementing a generalized Gibbs sampler for structurally varying models; enabling the user to specify blocks of variables that are to be sampled jointly to avoid problems with slow mixing; borrowing compiler techniques from the logic programming field to reduce the constant factors; and building in parametric learning. With these changes, we expect BLOG to be usable over a wide range of applications with only minimal user intervention.

## References

- Bacchus, F. (1990). *Representing and Reasoning with Probabilistic Knowledge*. MIT Press.
- Breese, J. S. (1992). Construction of belief and decision networks. *Computational Intelligence* 8(4), 624–647.
- Carnap, R. (1950). *Logical Foundations of Probability*. Univ. of Chicago Press.
- Charniak, E. and R. P. Goldman (1993). A Bayesian model of plan recognition. *Artificial Intelligence* 64(1), 53–79.
- de Salvo Braz, R., E. Amir, and D. Roth (2007). Lifted first-order probabilistic inference. In L. Getoor and B. Taskar (Eds.), *Introduction to Statistical Relational Learning*. MIT Press.
- Dunn, H. L. (1946). Record linkage. *Am. J. Public Health* 36(12), 1412–1416.
- Fellegi, I. and A. Sunter (1969). A theory for record linkage. *J. Amer. Stat. Assoc.* 64, 1183–1210.
- Gaifman, H. (1964). Concerning measures in first order calculi. *Israel J. Math.* 2, 1–18.
- Getoor, L. and B. Taskar (Eds.) (2007). *Introduction to Statistical Relational Learning*. MIT Press.
- Gilks, W. R., A. Thomas, and D. J. Spiegelhalter (1994). A language and program for complex Bayesian modelling. *The Statistician* 43(1), 169–177.
- Goodman, N. D., V. K. Mansinghka, D. Roy, K. Bonawitz, and J. B. Tenenbaum (2008). Church: A language for generative models. In *Proc. 24th Conf. on Uncertainty in AI*.
- Halpern, J. Y. (1990). An analysis of first-order logics of probability. *Artificial Intelligence* 46, 311–350.
- Huang, T. and S. J. Russell (1998). Object identification: A Bayesian analysis with application to traffic surveillance. *Artificial Intelligence* 103, 1–17.

- Kersting, K. and L. De Raedt (2001). Adaptive Bayesian logic programs. In *Proc. 11th International Conf. on Inductive Logic Programming*, pp. 104–117.
- Kisynski, J. and D. Poole (2009). Lifted aggregation in directed first-order probabilistic models. In *Proc. 21st International Joint Conf. on Artificial Intelligence*, pp. 1922–1929.
- Koller, D. and A. Pfeffer (1998). Probabilistic frame-based systems. In *Proc. 15th AAAI National Conf. on Artificial Intelligence*, pp. 580–587.
- Laskey, K. B. (2008). MEBN: A language for first-order Bayesian knowledge bases. *Artificial Intelligence* 172, 140–178.
- McAllester, D., B. Milch, and N. D. Goodman (2008). Random-world semantics and syntactic independence for expressive languages. Technical Report MIT-CSAIL-TR-2008-025, Massachusetts Institute of Technology.
- Milch, B., B. Marthi, S. Russell, D. Sontag, D. L. Ong, and A. Kolobov (2005). BLOG: Probabilistic models with unknown objects. In *Proc. 19th International Joint Conf. on Artificial Intelligence*, pp. 1352–1359.
- Milch, B., B. Marthi, D. Sontag, S. Russell, D. L. Ong, and A. Kolobov (2005). Approximate inference for infinite contingent Bayesian networks. In *Proc. 10th International Workshop on Artificial Intelligence and Statistics*.
- Milch, B. and S. Russell (2006). General-purpose MCMC inference over relational structures. In *Proc. 22nd Conf. on Uncertainty in Artificial Intelligence*, pp. 349–358.
- Milch, B. C. (2006). *Probabilistic Models with Unknown Objects*. Ph.D. thesis, Univ. of California, Berkeley.
- Nilsson, N. J. (1986). Probabilistic logic. *Artificial Intelligence* 28(1), 71–87.
- Pasula, H., B. Marthi, B. Milch, S. Russell, and I. Shpitser (2003). Identity uncertainty and citation matching. In *Advances in Neural Information Processing Systems 15*. MIT Press.
- Pasula, H. and S. Russell (2001). Approximate inference for first-order probabilistic languages. In *Proc. 17th International Joint Conf. on Artificial Intelligence*, pp. 741–748.
- Pasula, H., S. J. Russell, M. Ostland, and Y. Ritov (1999). Tracking many objects with many sensors. In *Proc. 16th International Joint Conf. on Artificial Intelligence*, pp. 1160–1171.
- Pfeffer, A. (2000). *Probabilistic Reasoning for Complex Systems*. Ph.D. thesis, Stanford Univ.
- Pfeffer, A. (2007). The design and implementation of IBAL: A general-purpose probabilistic language. In L. Getoor and B. Taskar (Eds.), *Introduction to Statistical Relational Learning*. MIT Press.

- Poole, D. (1993). Probabilistic Horn abduction and Bayesian networks. *Artificial Intelligence* 64(1), 81–129.
- Poole, D. (2003). First-order probabilistic inference. In *Proc. 18th International Joint Conf. on Artificial Intelligence*, pp. 985–991.
- Richardson, M. and P. Domingos (2006). Markov logic networks. *Machine Learning* 62, 107–136.
- Russell, S. and S. Vaidya (2009). Machine learning and data mining for Comprehensive Test Ban Treaty monitoring. Technical Report LLNL-TR-416780, Lawrence Livermore National Laboratory.
- Sato, T. and Y. Kameya (1997). PRISM: A symbolic–statistical modeling language. In *Proc. 15th International Joint Conf. on Artificial Intelligence*, pp. 1330–1335.
- Scott, D. and P. Krauss (1966). Assigning probabilities to logical formulas. In J. Hintikka and P. Suppes (Eds.), *Aspects of Inductive Logic*. North-Holland.
- Sittler, R. W. (1964). An optimal data association problem in surveillance theory. *IEEE Trans. Military Electronics MIL-8*, 125–139.
- Taskar, B., P. Abbeel, and D. Koller (2002). Discriminative probabilistic models for relational data. In *Proc. 18th Conf. on Uncertainty in Artificial Intelligence*, pp. 485–492.
- Wellman, M. P., J. S. Breese, and R. P. Goldman (1992). From knowledge bases to decision models. *Knowledge Engineering Review* 7, 35–53.