BLOG: Probabilistic Models with Unknown Objects

Brian Milch, Bhaskara Marthi, Stuart Russell,
David Sontag, Daniel L. Ong and Andrey Kolobov

UC Berkeley, Computer Science Div., Berkeley, CA 94720-1776, USA
{milch,bhaskara,russell,dsontag,dlong,karaya1}@cs.berkeley.edu

Abstract. This paper introduces BLOG, a formal language for defining probability models with unknown objects and identity uncertainty. A BLOG model describes a generative process in which some steps add objects to the world, and others determine attributes and relations on these objects. Subject to certain acyclicity constraints, a BLOG model specifies a unique probability distribution over first-order model structures that can contain varying and unbounded numbers of objects. Furthermore, inference algorithms exist for a large class of BLOG models.

Keywords. Knowledge representation, probability, first-order logic, identity uncertainty

1 Introduction

Human beings and AI systems must make inferences about the entities and events that underlie their observations. For instance, a system that builds a bibliographic database based on works-cited lists must reason about the researchers and publications referred to by the observed citations. A radar tracking system must reason about the aircraft that underlie the observed radar blips. No prespecified list of objects is given; the agent must infer the existence of objects that were not known initially to exist. The agent also faces the problem of *identity uncertainty* (also called *record linkage* or *data association*): determining when two observations, such as citation strings or radar blips, correspond to the same object.

Probability models for such tasks are not new: Bayesian models for data association have been used since the 1960s. The models are written in mathematical English and converted by hand into special-purpose code. Recently, *formal representation languages* such as graphical models have led to general inference algorithms, more sophisticated models, and automated model selection (structure learning). However, there is not yet a formal language that can describe probability models with unknown objects in a compact and intuitive way. This paper introduces BLOG (Bayesian LOGic), a language that meets this requirement; for a more complete discussion, see [1]. We begin with a motivating example:

Dagstuhl Seminar Proceedings 05051
Probabilistic, Logical and Relational Learning - Towards a Synthesis
http://drops.dagstuhl.de/opus/volltexte/2006/416

```
1  type Aircraft;  type Blip;

2  random R6Vector State(Aircraft, NaturalNum);
3  random R3Vector ApparentPos(Blip);

4  nonrandom NaturalNum Pred(NaturalNum) = Predecessor;

5  generating Aircraft Source(Blip);
6  generating NaturalNum Time(Blip);

7  #Aircraft ~ NumAircraftDistrib();

8  State(a, t)
9    if t = 0 then ~ InitState()
10   else ~ StateTransition(State(a, Pred(t)));

11 #Blip: (Source, Time) -> (a, t)
12   ~ DetectionDistrib(State(a, t));

13 #Blip: (Time) -> (t)
14   ~ NumFalseAlarmsDistrib();

15 ApparentPos(r)
16   if (Source(r) = null) then ~ FalseAlarmDistrib()
17   else ~ ObsDistrib(State(Source(r), Time(r)));
```

Fig. 1. BLOG model for Ex. 1.

Example 1. An unknown number of aircraft are flying in some volume of airspace. We observe the area with radar: aircraft appear as identical blips on a radar screen. Each blip gives the approximate position of the aircraft that generated it. However, some blips may be false detections, and some aircraft may not be detected at a given time step. What aircraft exist, and what are their trajectories? Are there any unobserved aircraft?

A BLOG model for this scenario is shown in Fig. 1. Intuitively, this BLOG model describes the following generative process: first sample the number of aircraft in the area. Then for each time step t (starting at $t = 0$), choose the state (position and velocity) of each aircraft given its state at time $t-1$. Also, for each aircraft a and time step t, possibly generate a radar blip r with $\mathsf{Source}(r) = a$ and $\mathsf{Time}(r) = t$. Whether a blip is generated or not depends on the state of the aircraft. Also, at each step, generate some false alarm blips r' with $\mathsf{Source}(r') = \mathsf{null}$. Finally, sample the position for each blip given the state of its source aircraft (or using a default distribution for false alarms).

Different outcomes of this process include different sets of aircraft and radar blips. Moreover, the generative process cannot simply be split into one stage that chooses what objects exist and a second stage that chooses their attributes. The existence of radar blips is determined at many separate steps in the generative process (one for each aircraft and each time step), and depends on the positions of the aircraft.

2 Syntax and Semantics

A BLOG model defines a probability distribution over model structures of a certain typed first-order logical language. The first few statements in a BLOG model define the particular logical language being used. For instance, Fig. 1 declares the types Aircraft and Blip, and defines function symbols such as $\mathsf{State}(a, t)$, which returns the state of aircraft a at time t. The aircraft state is an object of the built-in type R6Vector.

A *model structure* of a typed first-order language specifies the set of objects that exist of each type, and the value of each function symbol on each tuple of arguments. A BLOG model defines a distribution over a particular set of model structures, called the *possible worlds*. The possible worlds all contain certain *guaranteed objects* (such as natural numbers and $\mathbb{R}^6$ vectors), and all assign the same interpretations to *nonrandom functions* such as Pred (line 4). Given this nonrandom starting point, a generative process constructs a complete possible world step by step. The steps are of two kinds: some add new objects to the structure, and others set the value of a function on some arguments. These two kinds of steps are described by *number statements* and *dependency statements*.

Line 11 in Fig. 1 is a number statement. It says that for each aircraft a and time step t, there exist some number of radar blips r such that $\mathsf{Source}(r) = a$ and $\mathsf{Time}(r) = t$. The values of Source and Time are set when blips are added to the world; thus these functions are declared as *generating* functions on lines 5–6. The distribution over how many blips are added is given by the *elementary conditional probability distribution* (CPD) `DetectionDistrib`, which takes $\mathsf{State}(a, t)$ as an argument. On line 13, there is a number statement that generates "false alarm" blips r' with $\mathsf{Time}(r') = t$ and $\mathsf{Source}(r') = \mathsf{null}$. Line 7 is a very simple number statement: it defines a distribution over the total number of aircraft.

Generative steps that set the values of functions are described by dependency statements, such as the one for $\mathsf{State}(a, t)$ starting on line 8 of Fig. 1. A dependency statement consists of a sequence of *clauses*, each of which consists of a condition, an elementary CPD, and a sequence of CPD arguments. For a given assignment of objects to the variables a, t, the clause that applies is the first one whose condition is satisfied in the model structure constructed so far. The distribution over function values is determined by evaluating the CPD arguments and passing them to the specified CPD.

We have presented BLOG semantics intuitively in terms of a generative process. The full version of this paper [1] also includes declarative semantics for BLOG, defined in terms of the joint distribution of a set of *basic random variables* whose values uniquely specify a possible world.

Theorem 1. *If a* BLOG *model satisfies certain acyclicity conditions described in the full version of the paper, then it defines a unique probability distribution over possible worlds.*

3 Inference

A BLOG model need not impose any upper bound on the number of objects that exist. Thus, it appears that there are BLOG models where inference is not even decidable. However, for a large class of BLOG models, we can apply a sampling-based approximate inference algorithm that takes finite time per sampling step and converges to the correct distribution. We illustrate this algorithm on a very simple example.

Example 2. An urn contains an unknown number of balls—say, a number chosen from a Poisson distribution with mean 6. Balls are equally likely to be blue or green. We draw 10 balls from the urn, observing the color of each and replacing it. Observed colors are wrong with probability 0.2. Given that 5 drawn balls appeared green and 5 appeared blue, what is the posterior distribution for the number of balls in the urn?

An abbreviated BLOG model for this example is given in Fig. 2. To do inference in this model, we compile it into a *contingent Bayesian network* (CBN) [2] with a variable for the number of balls, an infinite sequence of TrueColor variables (we cannot limit this sequence to any finite length *a priori* because the number of balls is unbounded), and BallDrawn and ObsColor variables for each of the ten draws. We then apply the likelihood weighting algorithm described in [2]. In this case, the algorithm samples the number-of-balls variable first, then the BallDrawn variables, and then the TrueColor variables for those balls that appear as values for the BallDrawn variables. The algorithm detects that all the other TrueColor variables are irrelevant in this context, and do not need to be sampled. Thus, a sample is generated in finite time although the set of variables is infinite. The results of five runs of this algorithm with 5 million samples each are shown in Fig. 3; note that the five runs agree almost perfectly with each other and with an (almost) exhaustive calculation. Milch *et al.* [2] give conditions under which this algorithm is guaranteed to run in finite time per sampling step on a CBN; in the full version of this paper [1], we discuss how those conditions can be lifted to BLOG models.

```
1 guaranteed Color Blue, Green;
2 guaranteed Draw Draw1, Draw2, ..., Draw10;

3 #Ball ~ Poisson[6]();

4 TrueColor(b) ~ TabularCPD[[0.5, 0.5]]();

5 BallDrawn(d) ~ UniformChoice[]({Ball b});

6 ObsColor(d)
7     if (BallDrawn(d) != null) then
8         ~ TabularCPD[[0.8, 0.2], [0.2, 0.8]]
9             (TrueColor(BallDrawn(d)));
```

Fig. 2. BLOG model for the urn-and-balls scenario of Ex. 2.

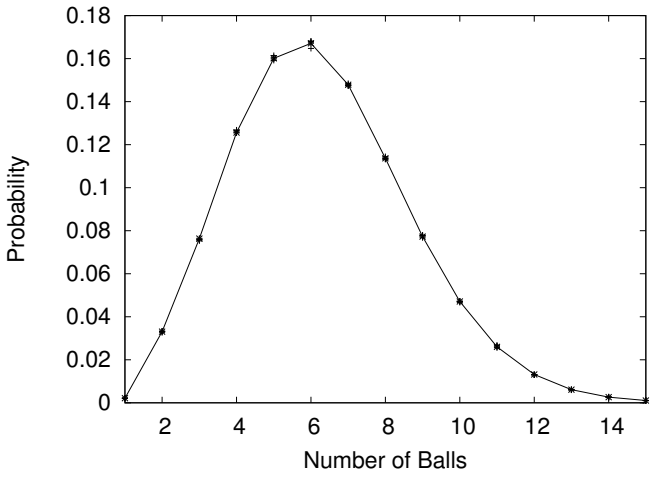

Fig. 3. Posterior probabilities for various numbers of balls (Ex. 2) from five sampling runs (crosses) and an exact calculation (solid line).

Theorem 2. *There is a convergent sampling algorithm that performs approximate inference in finite time per sampling step on any* BLOG *model that satisfies certain technical conditions described in the full paper.*

4 Conclusion

BLOG is a representation language for probability models with unknown objects. It contributes to the solution of a very general problem in AI: intelligent systems must represent and reason about objects, but those objects may not be known *a priori* and may not be uniquely identified by sensors. A BLOG model defines a generative process that creates first-order model structures by adding objects and setting function values. The steps that add objects are defined by *number statements*, which generalize existing approaches [3,4,5,6] by allowing the set of objects generated to depend on the existence and attributes of other objects. Much remains to be done, especially on inference: we expect to employ MCMC with user-defined proposal distributions, and develop algorithms that exploit interchangeability among objects.

References

1. Milch, B., Marthi, B., Russell, S., Sontag, D., Ong, D.L., Kolobov, A.: BLOG: Probabilistic models with unknown objects. In: Proc. 19th International Joint Conf. on AI. (2005) To appear.
2. Milch, B., Marthi, B., Sontag, D., Russell, S., Ong, D.L., Kolobov, A.: Approximate inference for infinite contingent Bayesian networks. In: 10th International Workshop on AI and Statistics. (2005)

3. Koller, D., Pfeffer, A.: Probabilistic frame-based systems. In: Proc. 15th National Conf. on AI. (1998) 580–587
4. Getoor, L., Friedman, N., Koller, D., Taskar, B.: Learning probabilistic models of relational structure. In: Proc. 18th International Conf. on Machine Learning. (2001) 170–177
5. Pasula, H., Marthi, B., Milch, B., Russell, S., Shpitser, I.: Identity uncertainty and citation matching. In: Advances in Neural Information Processing Systems 15. MIT Press, Cambridge, MA (2003)
6. Laskey, K.B.: MEBN: A logic for open-world probabilistic reasoning. Technical report, George Mason Univ. (2004)