Approximate Inference in Structured Bayesian Networks

R. Kumar, K.S. Ng

College of Engineering and Computer Science
The Australian National University

3 August 2009

1 Introduction

We investigate Monte Carlo inference techniques for structured Bayesian networks. In particular, we describe an algorithm for choosing the number of samples to make in static networks, and a particle filter algorithm for dynamic networks.

A *structured* Bayesian network is a Bayesian network in which some of the nodes represent variables over composite, structured domains such as lists or sets. Permitting structured domains can reduce the number of nodes required for modeling, and can ease making a model intuitive. A cost can be constructing densities over complex domains and sampling from those densities, although these tasks are not always difficult.

In this paper we focus on three examples of problems solvable with structured Bayesian networks; two use static networks and one uses a dynamic network. We begin with a brief description of inference problems in Bayesian networks, including the exact solution. In §3 we give a short introduction to structured domains and our notation. In §4 we describe a sampling-based inference algorithm for static networks and apply it to two example problems. In §5 we describe a different sampling-based inference algorithm for dynamic networks, relate it to the static case, and apply it to an example problem. In §6 and §7 we list further work and conclude.

2 Exact Inference

Consider a Bayesian network over the variables $X = U \cup E$, where $U = \{x_1, x_2, \ldots, x_n\}$ (listed in topological order) is the set of unknown variables and $E = \{y_1, y_2, \ldots, y_m\}$ is the set of evidence variables. Suppose we assign value v_i to variable y_i, $i = 1, \ldots, m$. The partially instantiated joint probability density function over X can be written down in the following form

$$\left(\prod_{i=1}^{n} f_i(x_i \mid pa(x_i))\right)\left(\prod_{i=1}^{m} g_i(v_i \mid pa(y_i))\right),$$

where f_i and g_i are (conditional) probability functions and $pa(z_i)$ denotes the parent nodes of variable z_i in X. Marginalising out the unknown variables yields

$$\sum_{x_1}\sum_{x_2}\cdots\sum_{x_{n-1}}\sum_{x_n}\left(\prod_{i=1}^{n} f_i(x_i \mid pa(x_i))\right)\left(\prod_{i=1}^{m} g_i(v_i \mid pa(y_i))\right) \tag{1}$$

$$= \sum_{x_1} \sum_{x_2} \cdots \sum_{x_{n-1}} \prod_{i=1}^{n-1} f_i(x_i \mid pa(x_i)) \sum_{x_n} f_n(x_n \mid pa(x_n)) \prod_{i=1}^{m} g_i(v_i \mid pa(y_i))$$

$$\vdots$$

$$= \sum_{x_1} f_1(x_1 \mid pa(x_1)) \sum_{x_2} f_2(x_2 \mid pa(x_2)) \cdots \sum_{x_n} f_n(x_n \mid pa(x_n)) \prod_{i=1}^{m} g_i(v_i \mid pa(y_i))$$

$$= \mathbb{E}_{x_1 \sim f_1(\cdot \mid pa(x_1))} \mathbb{E}_{x_2 \sim f_2(\cdot \mid pa(x_2))} \cdots \mathbb{E}_{x_n \sim f_n(\cdot \mid pa(x_n))} \prod_{i=1}^{m} g_i(v_i \mid pa(y_i)), \tag{2}$$

where the last line can be further simplified by moving factors in the product outside expectations when applicable.

A typical query asks for the conditional probability of some variable's value given the values of the evidence variables, that is

$$\Pr(x_q = v_q \mid y_1 = v_1, \ldots, y_m = v_m) = \frac{\Pr(x_q = v_q, y_1 = v_1, \ldots, y_m = v_m)}{\Pr(y_1 = v_1, \ldots, y_m = v_m)},$$

which is (1) with x_q an evidence variable divided by (1) as given. Clearly we can ask queries of multiple variables by putting them all in the evidence set for the numerator. Furthermore we can query the conditional probability of $h(x_q) = v_q$ for some function h by multiplying the numerator by an indicator function for $h(x_q) = v_q$.

3 Structured Domains

The variables in our Bayesian network models are over structured domains.

4 Static Bayesian Networks

4.1 Approximate Inference

In structured Bayesian networks like those presented in [NLU09], it is often infeasible to compute expressions like (1) exactly for a given query. The equivalent form (2) does however suggest a natural sampling-based approach. We approximate $\mu_f = \mathbb{E}_{x \sim p(\cdot)} f(x)$ by drawing i.i.d samples $s = \{x_1, x_2, \ldots, x_n\}$ from $p(\cdot)$ and compute

$$\hat{\mu}_f = \frac{1}{n} \sum_{x \in s} f(x).$$

From the Central Limit Theorem (see Appendix A), we know that the distribution of $\hat{\mu}_f$ converges in distribution to $\mathcal{N}(\mu_f, \sigma_f^2/n)$ if the variance $\sigma^2 = \mathbb{E}_{x \sim p(\cdot)}(f(x) - \mu_f)^2$ is finite. This means with probability $1 - \alpha$, we have

$$|\hat{\mu}_f - \mu_f| \leq z_\alpha \frac{\sigma_f}{\sqrt{n}},$$

where z_α is the z-score chosen to put $1-\alpha$ of the area under the normal curve between $-z_\alpha \frac{\sigma_f}{\sqrt{n}}$ and $z_\alpha \frac{\sigma_f}{\sqrt{n}}$. Suppose we want $|\hat{\mu}_f - \mu_f| \leq \epsilon$ with probability $1 - \alpha$ for some given ϵ. Solving $z_\alpha \frac{\sigma_f}{\sqrt{n}} \leq \epsilon$ for n yields

$$n \geq \left(\sigma_f \frac{z_\alpha}{\epsilon} \right)^2. \tag{3}$$

Replacing σ_f with the sample variance $\hat{\sigma}_f$ in (3), which is not unreasonable for n sufficiently large, gives us an empirically testable condition for ensuring the desired accuracy of $\hat{\mu}_f$ with high

confidence. [HGJ07] gives an algorithm that starts with some n_{min} number of samples needed to put the sample mean in its asymptotic normal regime and then iteratively increases the sample size until a condition similar to (3) is reached (they scale the error bound by $|\mu_f|$). Using the fact that the sample variance $\hat{\sigma}_f$ is asymptotically distributed according to $\mathcal{N}(\sigma_f^2, (\mu_{4f} - \sigma_f^4)/n)$, where u_{4f} is the fourth central moment, they show that with probability at least $1 - 2\alpha$, the iterative algorithm terminates with

$$n \leq O\left(\frac{\sigma_f^2}{\mu_f^2} + \frac{\sigma_f}{\mu_f}\sqrt{\frac{\mu_{4f}}{\sigma_f^4} - 1}\right).$$

Given that (2) can be written as

$$\mathbb{E}_{(x_1,x_2,\ldots,x_n)\sim f_1(\cdot|pa(x_1))f_2(\cdot|pa(x_2))\cdots f_n(\cdot|pa(x_n))} \prod_{i=1}^{m} g_i(v_m \mid pa(y_i)),$$

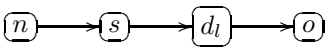
we now have an algorithm that can approximately compute any posterior given any required accuracy and confidence requirement, if the distribution from which we are sampling from has finite variance.

Balls problem

Our first example is a main example in [MMR+05]. An urn contains an unknown number of balls; the number is from a Poisson distribution with mean 6. The balls are equally likely to be blue or green. We draw some balls from the urn, observing the color of each and replacing it. The observed colors are wrong with probability 0.2. Given the list of observations, what is the conditional density on the number of balls in the urn? What is the probability that a ball was drawn more than once? We will consider the first question where the list of observations is either 10 or 15 blue balls in a row.

This problem was modelled as a structured Bayesian network in [NLU09]. Through a series of rewrites of the query expression, exact inference for a particular class of observations was shown to be tractable. We shall use the same Bayesian network but forego the rewrites, trading accuracy for efficiency by using sampling.

There are four variables in the model: the number n of balls in the urn, the actual set s of balls in the urn, the list d of balls drawn, and the list o of observations made. The length of the lists d and o, which we shall call l, is a parameter to the model. The figure below illustrates the graphical model.



We define N, $S(\cdot)$, and $D_l(\cdot)$ as the sets of support for the numbers of balls, the sets of balls, and the balls drawn (assuming as many were drawn as observed), and $P_n(n)$, $P_s(s \mid n)$, $P_{d,l}(d \mid s)$, and $P_{o,l}(o \mid d)$ as the relevant (conditional) probability distributions. The definitions are as implied by the problem statement:

$$\begin{aligned}
&P_n(n) = \text{Poisson}(6, n),\\
&P_s(s \mid n) = 2^{-|s|},\\
&P_{d,l}(d \mid s) = |s|^{-l}, \text{ and}\\
&P_{o,l}(o \mid d) = 0.8^c 0.2^{l-c}
\end{aligned}$$

where, in the last line, c is the number of places where the color of the ball in d agrees with the observation o.

Given an observation $o = [\text{blue}, \text{blue}, \dots]$ of length l and a query value for n, the desired probability in this problem is

$$\begin{aligned}
\Pr(n \mid o) &= \frac{\Pr(n, o)}{\Pr(o)} \\
&= \frac{\sum_{s \in S(n)} \sum_{d \in D(s)} \Pr(n, s, d, o)}{\sum_{n \in N} \sum_{s \in S(n)} \sum_{d \in D_l(s)} \Pr(n, s, d, o)} \\
&= \frac{\mathbb{E}_{(s,d) \sim P_s(\cdot|n) P_{d,l}(\cdot|s)} \frac{\Pr(n,s,d,o)}{P_s(s|n) P_{d,l}(d|s)}}{\mathbb{E}_{(n,s,d) \sim P_n(\cdot) P_s(\cdot|n) P d,l(\cdot|s)} \frac{\Pr(n,s,d,o)}{P_n(n) P_s(s|n) P_{d,l}(d|s)}}.
\end{aligned} \tag{4}$$

The denominator is a normalizing constant that only needs to be calculated once for a particular value of o since it doesn't depend on n. The joint probability distribution is $\Pr(n, s, d, o) = P_n(n) P_s(n \mid s) P_{d,l}(s \mid d) P_{o,l}(o \mid d)$.

We can sample from $P_n(\cdot)$, $P_s(\cdot \mid n)$, and $P_{d,l}(\cdot \mid s)$, assuming we can sample from $\mathrm{U}(0, 1)$ and from any uniform categorical distribution. For $P_n(\cdot)$ we sample a Poisson distribution with mean 6 using, for example, the algorithm in [Knu97]. For $P_s(\cdot \mid n)$ we create a set of n balls each of whose color is sampled from $\{\text{blue}, \text{green}\}$ uniformly. For $P_{d,l}(\cdot \mid s)$ we create a list of l balls sampled from s uniformly.

We used the iterative sampling algorithm to evaluate (4) with $o = \text{blue}^{10} = \{\text{blue}, \text{blue}, \dots, \text{blue}\}$ and $o = \text{blue}^{15}$ for $n = 1, 2, \dots, 15$. [NLU09] gives exact results for these queries that were calculated by exploiting symmetries in the problem and in repeated observations of the same color. With $(\epsilon, \alpha, n_{min}) = (10^{-4}, 10^{-3}, 100)$ for the denominator and $(\epsilon, \alpha, n_{min}) = (10^{-4}, 10^{-1}, 100)$ for the numerator the estimated probabilities compare well to the exact ones as shown in Figure 1. The figure also shows the number of samples required for each query.

Radar problem

Our second example is also from [MMR+05], but is more complicated. A volume of airspace contains an unknown number of aircraft; the number is from a Poisson distribution. The state of each aircraft at each time step is its position and velocity, and depends on the previous time step. We observe the airspace with (two dimensional) radar. Each radar blip at each time step gives the approximate position of an aircraft that generated it. Some blips are not generated by aircraft (false alarms), and sometimes aircraft do not generate a blip. Given the time series of observed blips, what is the conditional density on the state of the airspace (the number of aircraft, their trajectories, and the sources of blips)? We can also do forward inference and ask for the posterior density of a time series of observations. We will evaluate a backwards inference query of the aircrafts' velocities given their starting positions and observations.

We model this problem as a Bayesian network parameterized by the total number T of time steps under consideration. We assume a simple version of the problem in which the acceleration of each aircraft is known to be zero, and the number of aircraft remains constant over time. There are nine variables in our model including the number of aircraft n, the initial positions of the aircraft p_0, the initial velocities v_0, and six time series variables. The time series variables are of aircraft positions $\bar{p} = [p_0, \dots, p_{T-1}]$, of positions of aircraft that generate blips $\bar{g} = [g_0, \dots, g_{T-1}]$, of the number of false alarm sources $\bar{m} = [m_0, \dots, m_{T-1}]$, of positions of false alarm sources $\bar{f} = [f_0, \dots, f_{T-1}]$, of apparent positions $\bar{a} = [a_0, \dots, a_{T-1}]$, and of observed blips $\bar{o} = [o_0, \dots, o_{T-1}]$.

Each p_t is a list of length n of the positions of each aircraft at time t. v_0 is a list of length n of the velocities (change in each position coordinate) of each aircraft. Each g_t is a subset of the elements of p_t. Each m_t is an integer at least zero. Each f_t is a list (possibly with duplicates) of positions of false alarm sources. Each a_t is a list of apparent positions. Each o_t is a set of observed blip positions. The dependencies of the variables are illustrated in the graphical models in Figure 2.

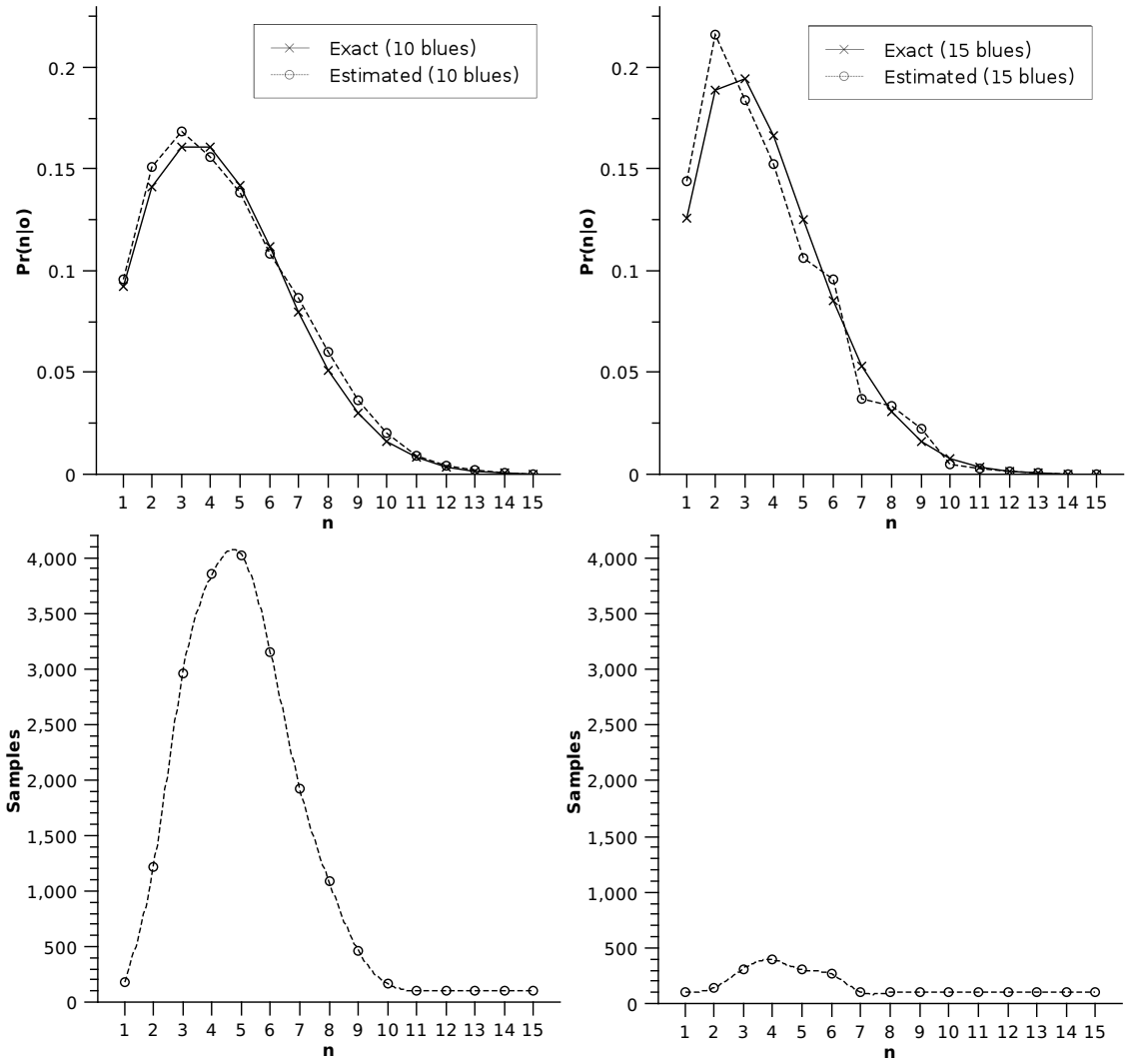


Figure 1: Accuracy of approximate inference in the Balls problem for a query of the number n of balls when the observed sequence o is either 10 (left) or 15 (right) blue balls in a row. The number of samples required for each approximated point, with a minimum of 100 samples, are shown in the lower graphs. The graphs on the left represent a single run, on the right the average of two runs.

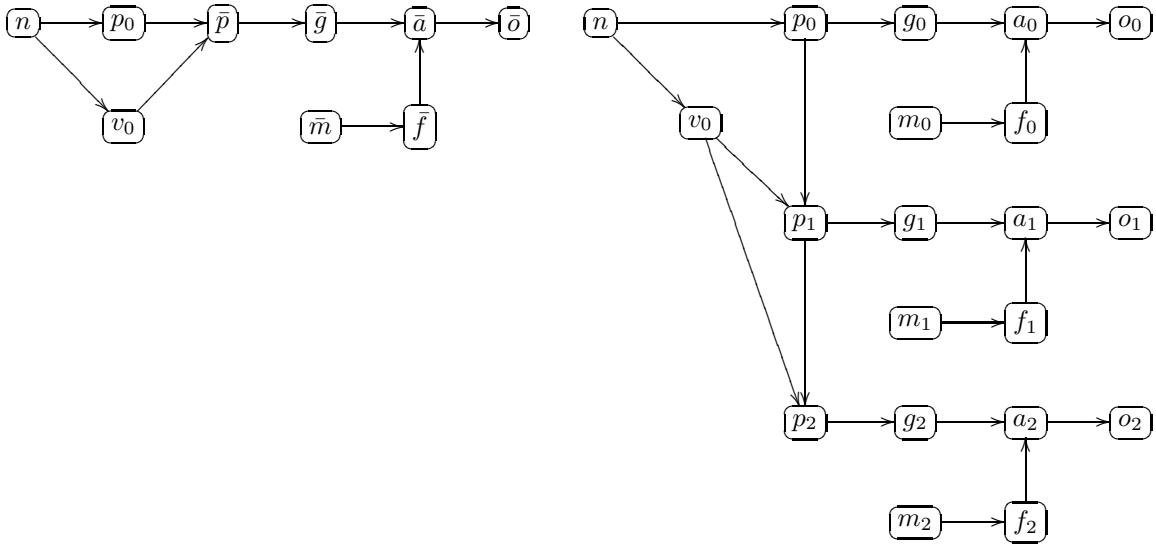


Figure 2: Graphical models illustrating the Radar problem. On the right the time series lists are flattened into the network, with $T = 3$, and the duplication of p_0 (in $\bar{p}$) is removed.

We can define the joint distribution over the variables in this problem with the density function

$$pdf(n, \bar{m}, \bar{f}, p_0, v_0, \bar{p}, \bar{g}, \bar{a}, \bar{o}) =$$
$$P_n(n)P_{\bar{m}}(\bar{m})P_{\bar{f}}(\bar{f} \mid \bar{m})P_{p_0}(p_0 \mid n)P_{v_0}(v_0 \mid n)P_{\bar{p}}(\bar{p} \mid p_0, v_0)P_{\bar{g}}(\bar{g} \mid \bar{p})P_{\bar{a}}(\bar{a} \mid \bar{f}, \bar{g})P_{\bar{o}}(\bar{o} \mid \bar{a})$$

where the factors are the conditional densities of the variables given their parents in the Bayesian network. The time series densities are defined as products of the densities at each time step.

$$P_{\bar{p}}(\bar{p} \mid p_0, v_0) = \prod_{t=0}^{T-1} P_p(p_t \mid p_{t-1}, v_0) \qquad P_{\bar{g}}(\bar{g} \mid \bar{p}) = \prod_{t=0}^{T-1} P_g(g_t \mid p_t)$$
$$P_{\bar{m}}(\bar{m}) = \prod_{t=0}^{T-1} P_m(m_t) \qquad P_{\bar{a}}(\bar{a} \mid \bar{f}, \bar{g}) = \prod_{t=0}^{T-1} P_a(a_t \mid f_t, g_t)$$
$$P_{\bar{f}}(\bar{f} \mid \bar{m}) = \prod_{t=0}^{T-1} P_f(f_t \mid m_t) \qquad P_{\bar{o}}(\bar{o} \mid \bar{a}) = \prod_{t=0}^{T-1} P_o(o_t \mid a_t)$$

The rest of the conditional densities are defined below. If the side conditions aren't satisfied, the densities are defined to be zero. μ_n and μ_f are the mean number of aircraft and of false alarm sources, respectively. $S = \{(x, y) \in \mathbb{Z}^2 \mid s_{\min} \leq x, y < s_{\max}\}$ and $V = \{(dx, dy) \in \mathbb{Z}^2 \mid v_{\min} \leq dx, dy < v_{\max}\}$ represent possible initial positions and initial velocities, respectively.[1] We use the same set S for possible false alarm source positions and observed blip positions, however we allow real (floating point) position coordinates after time step 0. σ_p^2 is the variance in each aircraft's position. λ_g is the probability that an aircraft generates a blip[2]. σ_a^2 is the variance in apparent positions. We index the set s in the definition of P_a in component-wise ascending order. We treat

[1] to be more realistic, these could be subsets of $\mathbb{R}^2$

[2] to be more realistic, this could depend on the aircraft's position

p_t as a set of its elements in defining P_g, and f_t and a_t similarly in defining P_a and P_o.

$$
\begin{aligned}
P_n(n) &= \text{Poisson}(\mu_n, n) \\
P_m(m) &= \text{Poisson}(\mu_f, m) \\
P_f(f_t \mid m_t) &= m_t! \times (s_{\max} - s_{\min})^{-2m_t}, \text{len}(f_t) = m_t, \forall i. f_t[i] \in S \\
P_{v_0}(v_0 \mid n) &= (v_{\max} - v_{\min})^{-2n}, \text{len}(v_0) = n \\
P_{p_0}(p_0 \mid n) &= (s_{\max} - s_{\min})^{-2n}, \text{len}(p_0) = n \\
P_p(p_t \mid p_{t-1}, v_0) &= \prod_{i=0}^{\text{len}(p_t)-1} \prod_{x=0}^{1} \mathcal{N}(p_{t-1}[i][x] + v_0[i][x], \sigma_p^2, p_t[i][x]), \text{len}(p_t) = \text{len}(p_{t-1}) = \text{len}(v_0) \\
P_g(g_t \mid p_t) &= \text{Binomial}(|T|, \lambda_g, |g_t|), T = \{(x, y) \in p_t \subset \mathbb{R}^2 \mid s_{\min} \leq x, y < s_{\max}\}, g_t \subseteq T \\
P_a(a_t \mid f_t, g_t) &= \prod_{s[i] \in f_t \cup g_t} \prod_{x=0}^{1} \mathcal{N}(s[i][x], \sigma_a^2, a_t[i]), \text{len}(a_t) = |f_t| + |g_t| \\
P_o(o_t \mid a_t) &= 1, o_t = a_t \cap S
\end{aligned}
$$

We assume we can sample from Poisson, Gaussian, and uniform (and thereby binomial and categorical) distributions. Then sampling from the above distributions is straightforward; we describe two examples. To sample from $P_{\bar{g}}(\cdot \mid \bar{p})$, for each p_t we include $p_t[i]$ if, firstly, it is in S and, secondly, according to a binary categorical sample with λ_g weight on inclusion. To sample from $P_a(\cdot \mid f_t, g_t)$ we construct the ordered set of source positions $s = f_t \cup g_t$, then for each source position $s[i]$, for each component $s[i][x]$, we sample a Gaussian with variance σ_a^2 around $s[i][x]$.

We tried the backward inference problem of estimating the posterior density on v_0 given p_0 and $\bar{o}$. The value of n is determined by the length of p_0, so only the other five variables need to be marginalized out. The desired probability is

$$
\begin{aligned}
&\Pr(v_0 \mid p_0, \bar{o}) \\
&= \frac{\Pr(v_0, p_0, \bar{o})}{\Pr(p_0, \bar{o})} \\
&= \frac{\sum_{\bar{m}} \int d\bar{f} \int d\bar{p} \int d\bar{g} \int d\bar{a} \cdot pdf(\dots)}{\sum_{\bar{m}} \int d\bar{f} \int dv_0 \int d\bar{p} \int d\bar{g} \int d\bar{a} \cdot pdf(\dots)} \\
&= \frac{\mathbb{E}_{(\bar{m}, \bar{f}, \bar{p}, \bar{g}, \bar{a}) \sim P_{\bar{m}}(\cdot) P_{\bar{f} \mid \bar{m}}(\cdot) P_{\bar{p}}(\cdot \mid p_0, v_0) P_{\bar{g}}(\cdot \mid \bar{p}) P_{\bar{a}}(\cdot \mid \bar{f}, \bar{g})} P_{p_0}(p_0 \mid n) P_{v_0}(v_0 \mid n) P_{\bar{o}}(\bar{o} \mid \bar{a})}{\mathbb{E}_{(\bar{m}, \bar{f}, v_0, \bar{p}, \bar{g}, \bar{a}) \sim P_{\bar{m}}(\cdot) P_{\bar{f} \mid \bar{m}}(\cdot) P_{v_0}(\cdot \mid n) P_{\bar{p}}(\cdot \mid p_0, v_0) P_{\bar{g}}(\cdot \mid \bar{p}) P_{\bar{a}}(\cdot \mid \bar{f}, \bar{g})} P_{p_0}(p_0 \mid n) P_{\bar{o}}(\bar{o} \mid \bar{a})}, n = \text{len}(p_0)
\end{aligned}
\tag{5}
$$

and as in the Balls example the denominator need only be calculated once for use with different values of v_0.

We used the following model parameters.

$T = 2$	$\mu_n = 3$	$\mu_f = 0.1$
$(s_{\min}, s_{\max}) = (-5, 6)$	$\sigma_p^2 = 0.1$	$\sigma_a^2 = 0.1$
$(v_{\min}, v_{\max}) = (-1, 2)$	$\lambda_g = 0.9$	

As sampling parameters we used $(\epsilon, \alpha, n_{min}) = (10^{-2}, 10^{-2}, 10^4)$ for the numerator and the denominator. As evidence variables we used

$$
\begin{aligned}
&p_0 = [(-1, 0), (0, 1), (1, -1)], \\
&o_0 = \{(-1, 0), (0, 1), (1, -1)\}, \text{ and} \\
&o_1 = \{(0, 0), (0, 1), (1, 0)\}.
\end{aligned}
$$

With a little thought we can come up with some likely values of v_0 given these two observations.

The following values would match a deterministic model without noise or false alarms.

$$v_0^1 = [(1,0),(0,0),(0,1)]$$
$$v_0^2 = [(1,1),(0,-1),(0,1)]$$
$$v_0^3 = [(1,1),(1,-1),(-1,1)]$$

The results of querying these values by sampling, averaged over three runs, are

$$\Pr(v_0^1 \mid p_0, \bar{o}) \approx 0.14,$$
$$\Pr(v_0^2 \mid p_0, \bar{o}) \approx 0.15, \text{ and}$$
$$\Pr(v_0^3 \mid p_0, \bar{o}) \approx 0.16,$$

thus these values for v_0 account for almost half of the probability. It's not hard to see that these are the only initial velocities that would match the observations in a deterministic model, so the rest of the probability must be covered by false alarms, variance in p_1 and in $\bar{a}$, and when some of the aircraft do not generate blips. For example, consider the following variations on v_0^1.

$$\Pr([(1,0)(0,1),(0,1)]) \approx 0.02$$
$$\Pr([(1,1)(0,0),(0,1)]) \approx 0.05$$
$$\Pr([(1,1)(0,1),(0,1)]) \approx 0.01$$

The number of samples required for each estimated expectation in the above queries was $n_{min} = 10^4$; in other words, in each case n_{min} either was large enough to satisfy the accuracy and confidence requirements or was not large enough to put the sample mean in its asymptotic normal regime. Possibly in this example, and almost certainly in a more complicated example, $n_{min} = 10^4$ would not be large enough, because the sample space is large but the probability may be concentrated. An improvement to our method here may be to use importance sampling to concentrate sampling in the areas of high probability. An instance of importance sampling is described in the next section.

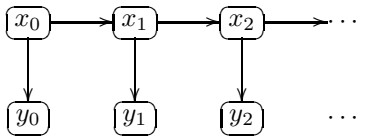
5 Dynamic Bayesian Networks

Dynamic Bayesian networks are used for representing sequences of variables parameterized by, for example, time. A dynamic network can be cut off at a certain point (time) and the subnetwork will be static and amenable to the treatment in §4. The Radar problem could be modelled using a dynamic network, but we presented it as static because we only looked at queries after a single number of time steps T.

When we wish to query the network at multiple time steps, then rather than applying static inference techniques to each subnetwork, we can use a more efficient inference algorithm known as Sequential Monte Carlo [JD08, AMGC02], or, in particular, particle filtering. Particle filtering is applicable to hidden Markov models, that is, models in which the unknown variables at time $T+1$ depend only on the unknown variables at time T, and the evidence variables at time T depend only on the unknown variables at time T. We shall only consider the case where there is a single unknown (state) variable and a single evidence (observation) variable at each time step.

5.1 Approximate Inference

We consider the hidden Markov model with state variables $x_i \in X_u$ and observation variables $y_i \in X_e$ for $t = 0, 1, 2, \ldots$, and where $f_0(x_0)$, $f(x_i \mid x_{i-1})$, and $g(y_i \mid x_i)$ are the (conditional) density functions. The dependencies are illustrated in the figure below.



Suppose we assign value v_i to variable y_i, $i = 0, \ldots$. For several (usually consecutive) values of T we wish to make a query of the subnetwork specified by $i < T$. As in §2 the partially instantiated joint density function can be written

$$\left(f_0(x_0) \prod_{i=1}^{T-1} f(x_i \mid x_{i-1}) \right) \left(\prod_{i=0}^{T-1} g(v_i \mid x_i) \right) \tag{6}$$

In this model, the same functions f and g are used at each time step. We define the function s_{T-1} by marginalizing out all but the latest unknown variable:

$$s_{T-1}(x_{T-1}) = \sum_{x_0} \sum_{x_1} \cdots \sum_{x_{T-3}} \sum_{x_{T-2}} \left(f_0(x_0) \prod_{i=1}^{T-1} f(x_i \mid x_{i-1}) \right) \left(\prod_{i=0}^{T-1} g(v_i \mid x_i) \right).$$

The particle filter algorithm maintains an approximation of s_T over time, and avoids recalculation by using the relation

$$\begin{aligned} s_0(x_0) &= f_0(x_0) g(v_0 \mid x_0), \\ s_T(x_T) &= \sum_{x_{T-1}} s_{T-1}(x_{T-1}) f(x_T \mid x_{T-1}) g(v_T \mid x_T) \\ &= \mathbb{E}_{x \sim s_{T-1}(\cdot)} f(x_T \mid x) g(v_T \mid x_T). \end{aligned}$$

We want to approximate $s_T(x_T)$ by sampling but usually we cannot sample from $s_{T-1}(\cdot)$ because a sampling procedure is difficult or impossible to devise. We can instead use importance sampling. We define a sampling density q_{T-1} that is non-zero on the support of s_{T-1}, then for $T > 0$

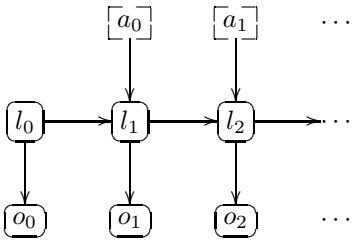
$$\begin{aligned} s_T(x_T) &= \sum_{x_{T-1}} \frac{q_{T-1}(x_{T-1})}{q_{T-1}(x_{T-1})} s_{T-1}(x_{T-1}) f(x_T \mid x_{T-1}) g(v_T \mid x_T) \\ &= \mathbb{E}_{x \sim q_{T-1}(\cdot)} \frac{s_{T-1}(x) f(x_T \mid x) g(v_T \mid x_T)}{q_{T-1}(x)} \end{aligned}$$

which we can approximate by sampling, assuming we apply this technique recursively to approximate $s_{T-1}(x)$.

Localization problem

Our third example is localization of an agent with range sensors in an environment consisting of thin straight walls. Given the sensor readings at each time step, and its model of the sensors, the environment, and its own movement, the agent must estimate its position. The Bayesian network is dynamic with each variable parameterized by time.

For localization, the map of the world is fixed and known. The variables at each time step are the (believed) location of the agent l_t, and the data from the sensors o_t. The joint density function of the variables is conditioned on the movement actions taken by the agent a_t. We represent the variables as follows. l_t is a pair of coordinates (x, y). o_t the is a list of pairs of the form (a, r) where a is the angle relative to the world in which the sensor is pointed and r is the distance recorded by the sensor which can be ∞ indicating out of range. a_t is a pair (a, d) where a is the intended direction of movement as an angle relative to the world and d is the intended distance of movement. The diagram below illustrates the dependencies. The model is conditioned on the actions; they are shown in dashed boxes to indicate that they are not variables themselves although they do affect the conditional probability densities of the variables.



The conditional density functions f and g in the particle filter template are defined in this problem by the "transition model", $f(l_t \mid a_{t-1}, l_{t-1})$, and "sensor model", $g(o_t \mid l_t)$. For localization, we can use a transition model that says the new location is distributed normally around the intended location. We can sample this distribution directly, so q_t in this case is the same as s_t. The sensor model says that there is a laser reading distributed normally around the locations of walls in the lasers' ranges.

This problem can be extended to simultaneous localization and mapping, making the model more complicated: the location variable is replaced by a compound variable containing both the (believed) location and map of the world. The transition model must update a map of the world, so the sampling density q_t, which is (partially) over possible maps, benefits from careful construction.

6 Further work

Automation

The (conditional) density functions in the Balls and Radar models were mostly products or other combinations of elementary densities (uniform, Poisson, etc.). A high-level specification of models like these may be enough to generate code implementing the density functions themselves, or the procedures to sample from them, or both.

Nested sampling

In our approach to approximate inference in static models, each sample is of all the unknown variables at once, and the joint density is calculated for this tuple of values along with the evidence. Specifically the sampling method suggests

$$\mathbb{E}_{(x_1,x_2,\ldots,x_n)\sim f_1(\cdot|pa(x_1))f_2(\cdot|pa(x_2))\cdots f_n(\cdot|pa(x_n))} \prod_{i=1}^{m} g_i(v_m \mid pa(y_i)),$$

rather than the equivalent

$$\mathbb{E}_{x_1\sim f_1(\cdot|pa(x_1))}\mathbb{E}_{x_2\sim f_2(\cdot|pa(x_2))}\cdots\mathbb{E}_{x_n\sim f_n(\cdot|pa(x_n))} \prod_{i=1}^{m} g_i(v_i \mid pa(y_i)).$$

From informal experiments on the second approach, where sampling one variable may involve recursive approximation by sampling, it seems that the first, unnested, approach performs better (requires fewer total samples). This is worth investigating further, especially since the recursive approach would be preferred for an exact calculation to avoid duplicating work.

In [HGJ07] the full recursive sampling algorithm is presented, because in general unnesting is not possible. Furthermore they combine the nested approach with with multi-tree sampling as a variance reduction technique, providing automatic concentration on difficult expectations.

Rao-Blackwellisation

Certain expectations can be calculated exactly efficiently. Whether these exact computations can be easily combined with sampling for the intractable computations within the same query remains to be seen.

7 Conclusion

This paper describes inference problems in structured Bayesian networks and standard sampling-based techniques for approximate inference. We have showed that approximate inference by sampling is appropriate for structured models that are either static or dynamic. The advantage of using structured networks is that these models are relatively small and comprehensible.

References

[AMGC02] Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50:174–188, 2002.

[HGJ07] Michael Holmes, Alexander Gray, and Charles Lee Isbell Jr. Ultrafast monte carlo for statistical summations. In *NIPS*, 2007.

[JD08] A. M. Johansen and A. Doucet. A tutorial on particle filtering and smoothing: Fifteen years later. In Crisan and Roszovskii, editors, *Handbook of Nonlinear Filtering*. Oxford University Press, 2008. To appear.

[Knu97] Donald E. Knuth. *The Art of Computer Programming, Volume 2, Seminumerical Algorithms*. Addison-Wesley, 3rd edition, 1997.

[MMR+05] Brian Milch, Bhaskara Marthi, Stuart Russell, David Sontag, Daniel L. Ong, and Andrey Kolobov. BLOG: Probabilistic models with unknown objects. In L.P. Kaelbling and A. Saffiotti, editors, *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pages 1352–1359, 2005.

[NLU09] K. S. Ng, J. W. Lloyd, and W. T. B. Uther. Probabilistic modelling, inference and learning using logical theories. *Annals of Mathematics and Artificial Intelligence*, 2009. In press.

A Central Limit Theorem

Here we prove a version of the Central Limit Theorem that we use in this paper.

Theorem A.1. *Let $X_1, X_2, \ldots, X_n$ be n random variables that are independently and identically distributed according to a density $p(\cdot)$ on Ω and let $f : \Omega \to \mathbb{R}$ be a real-valued function on Ω. If $\mu_f = \mathbb{E}_{x\sim p(\cdot)} f(x)$ and $\sigma_f^2 = \mathbb{E}_{x\sim p(\cdot)}(f(x) - \mu_f)^2$ are both finite, then the sample average*

$$\bar{X}_f = \frac{1}{n}(f(X_1) + f(X_2) + \cdots + f(X_n))$$

has a distribution that tends to a Gaussian distribution with mean μ_f and variance σ_f^2/n as $n \to \infty$.

Proof. We show that the distribution $q(y)$ of $Y_f = \frac{\bar{X}_f - \mu_f}{\sigma_f/\sqrt{n}}$ converges to the standard normal distribution as $n \to \infty$ from which the result for $\bar{X}_f$ follows. Let $Z(x) = \frac{f(x)-\mu_f}{\sigma_f}$. Since

$$\begin{aligned}\sum_{i=1}^{n} Z(X_i) &= \frac{1}{\sigma_f}\sum_{i=1}^{n}(f(X_i) - \mu_f) \\ &= \frac{n}{\sigma_f}(\bar{X}_f - \mu_f) \\ &= \sqrt{n}Y_f\end{aligned}$$

we have $Y_f = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} Z(X_i)$. Also

$$\begin{aligned}
\mathbb{E}_{x\sim p(\cdot)} Z(x) &= \sum_{x\in\Omega} \frac{f(x)-\mu_f}{\sigma_f} p(x) \\
&= \frac{1}{\sigma_f}\left(\sum_{x\in\Omega} f(x)p(x) - \mu_f \sum_{x\in\Omega} p(x)\right) \\
&= \frac{1}{\sigma_f}(\mu_f - \mu_f) \\
&= 0
\end{aligned}$$

and

$$\begin{aligned}
\mathbb{E}_{x\sim p(\cdot)} Z(x)^2 &= \sum_{x\in\Omega} \left(\frac{f(x)-\mu_f}{\sigma_f}\right)^2 p(x) \\
&= \frac{1}{\sigma_f^2} \sum_{x\in\Omega} (f(x_i)-\mu_f)^2 p(x) \\
&= \frac{1}{\sigma_f^2} \mathbb{E}_{x\sim p(\cdot)} (f(x)-\mu_f)^2 \\
&= \frac{\sigma_f^2}{\sigma_f^2} \\
&= 1
\end{aligned}$$

that is, the first two moments of $Z(X_i)$ are 0 and 1 respectively. The moment generating function for Y_f is

$$\begin{aligned}
m_{Y_f}(t) &= \mathbb{E}_{y\sim q(\cdot)} \exp(ty) \\
&= \mathbb{E}_{x_1,\ldots,x_n\sim p^n(\cdot)} \exp\left(\frac{t}{\sqrt{n}} \sum_{i=1}^{n} Z(x_i)\right) \\
&= \mathbb{E}_{x_1,\ldots,x_n\sim p^n(\cdot)} \exp\left(\frac{tZ(x_1)}{\sqrt{n}}\right) \ldots \exp\left(\frac{tZ(x_n)}{\sqrt{n}}\right) \\
&= \left(\mathbb{E}_{x_1\sim p(\cdot)} \exp\left(\frac{tZ(x_1)}{\sqrt{n}}\right)\right) \times \cdots \times \left(\mathbb{E}_{x_n\sim p(\cdot)} \exp\left(\frac{tZ(x_n)}{\sqrt{n}}\right)\right) \\
&= \left(\mathbb{E}_{x\sim p(\cdot)} \exp\left(\frac{tZ(x)}{\sqrt{n}}\right)\right)^n \\
&= \left(1 + \frac{t\mathbb{E}_{x\sim p(\cdot)} Z(x)}{\sqrt{n}} + \frac{t^2 \mathbb{E}_{x\sim p(\cdot)} Z(x)^2}{2!n} + \frac{t^3 \mathbb{E}_{x\sim p(\cdot)} Z(x)^3}{3! n^{3/2}} + \ldots\right)^n \\
&= \left(1 + 0 + \frac{t^2}{2n} + \frac{t^3 \mathbb{E}_{x\sim p(\cdot)} Z(x)^3}{3! n^{3/2}} + \ldots\right)^n \\
&= \left(1 + \frac{1}{n}\left(\frac{t^2}{2} + \frac{t^3 \mathbb{E}_{x\sim p(\cdot)} Z(x)^3}{3!\sqrt{n}} + \ldots\right)\right)^n \\
\implies \lim_{n\to\infty} m_{Y_f}(t) &= \lim_{n\to\infty} \left(1 + \frac{u}{n}\right)^n \quad \text{where } u = \frac{t^2}{2} + \frac{t^3 \mathbb{E}_{x\sim p(\cdot)} Z(x)^3}{3!\sqrt{n}} + \ldots \\
&= \exp\left(\lim_{n\to\infty} u\right) \\
&= \exp\left(\frac{t^2}{2}\right)
\end{aligned}$$

since all but the first term of u have n in the denominator. But $m(t) = \exp(\frac{t^2}{2})$ is the moment generating function for the standard normal distribution, therefore Y_f has a standard normal distribution by the Lévy continuity theorem. □