Study Notes for AI (CS 440/520)
Lecture 12: Approximate Inference in Bayesian Networks

Corresponding Book Chapters: 14.4-14.5
Note: These notes provide only a short summary and some highlights of the material covered in the corresponding lecture based on notes collected from students. Make sure you check the corresponding chapters. Please report any mistakes in or any other issues with the notes to the instructor.

1 Approximate Inference in Bayesian Networks

Previously, we have talked about using enumeration trees and variable elimination to solve problems modeled by a Bayesian network. Exact inference in Bayesian networks is in the general case intractable. Thus, we consider approximate inference techniques for Bayesian networks, which employ sampling in order to deal with the computational problems that arise in Bayesian networks. There are two families of sampling-based algorithms for approximate inference in Bayesian networks, also known as Monte Carlo algorithms:

1. Direct Sampling
2. Markov Chain Simulation

1.1 Direct Sampling

The basic idea is that when you have a probability it is possible to sample from it. So given a Bayesian network and its conditional probabilities, it is possible to sample an "atomic event". An atomic event is an assignment of values to the random variables.

The basic operation of direct sampling is that we sample each variable in topological order according to the conditional probability over its parents.

Consider, for example, the Bayesian network in Figure 1. We will use it to demonstrate how direct sampling works:

- We first sample the value of the variable "Cloudy". $P(Cloudy) =< 0.5, 0.5 >$. Assume that sampling returns: "Cloudy"= true.
- We then sample the value of the variable "Sprinkler". Since "Cloudy" is true, we have to use the corresponding conditional probability: $P(Sprinkler|Cloudy = True) =< 0.1, 0.9 >$. Assume that the sample returns: "Sprinkler"=false.
- We then sample the value of the variable "Rain". Since "Cloudy" is true, we have to use the corresponding conditional probability: $P(Rain|Cloudy = True) =< 0.8, 0.2 >$. Suppose that the sampling process return: "Rain"=true.
- We then sample the value of the variable "Wet Grass". Since "Sprinkler" is false and "Rain" is true, we have to use the corresponding conditional probability: $P(WetGrass|Sprinkler = False, Rain = true) =< 0.9, 0.1 >$. Assume that the sampling returns true.

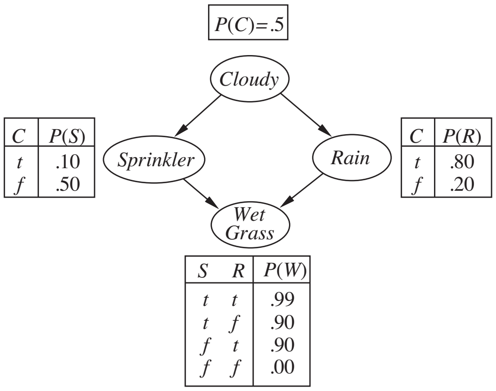

Figure 1: The Bayesian network for the rain-grass example.

Based on the above operation the atomic event that we managed to sample is:

$$< Cloudy, Sprinkler, Rain, WetGrass >=< true, false, true, true >$$

Suppose we generate M total samples and let $N(X_1, \ldots, X_n)$ be the number of samples that have produced the atomic event: $X_1, \ldots, X_n$. The important property of direct sampling is that:

$$\lim_{M \to \infty} \frac{N(X_1, \ldots, X_n)}{M} = P(X_1, \ldots, X_n)$$

This means that as the number of samples we produce goes to infinity, the population approaches the true joint probability distribution. We say that direct sampling is able to the limit to produce a consistent estimate.

For the rain-grass example, the actual probability of the atomic event we sampled is:

$$P(< true, false, true, true >) = 0.5 \times 0.9 \times 0.8 \times 0.9 = 0.324$$

In the limit 32.4% of the samples are of this event. So we expect that after we sample 100,000 samples, the number of those that have produced this atomic event to be somewhere close to 32,400.

1.1.1 Computing Conditional Probabilities

What we typically want to compute with a Bayesian network is a conditional probability of the form: $P(X|e)$ (e.g., what is the probability of a burglary if both John and Mary call). We have two algorithms that work on top of direct sampling to answer such queries:

1. Rejection Sampling
2. Likelihood Weighting

Rejection Sampling The idea in rejection sampling is that initially we will produce the samples with the same approach as we described above with direct sampling. Then we will reject those samples that do not match the evidence. Let $\hat{P}(X|e)$ be the estimated distribution that the algorithm returns, then:

$$\hat{P}(X|e) = \frac{N(X, e)}{N(e)} \approx P(X|e)$$

For example, consider that we want to compute the probability $P(Rain|Sprinkler = true)$ using 100 samples. Assume also that in 73 of these samples, the variable "Sprinkler" appears with value false and in the remaining 27 it appers with value true. What the algorithm does, is that it rejects the 73 samples where “Sprinkler” appears with value false and considers only the rest. Lets assume that out of the remaining 27, 8 have a value of “Rain” equal to true and the 19 have a value false. In this case the value that rejection sampling is going to return is:

$$P(Rain|Sprinkler = true) =< \frac{8}{27}, \frac{19}{27} >=< 0.296, 0.704 >$$

The true answer to this question if we use exact inference is $< 0.3, 0.7 >$. The sampling process is able to only approximate the correct value. The error of the approach has a standard deviation of $\frac{1}{n}$, where n is the number of samples used.

The problem with rejection sampling is that it rejects too many samples. There was a lot of time invested in the generation of these samples that is not utilized by the method. The consecutive method, likelihood weighting, attempts to avoid this pitfall.

Likelihood Weighting The idea in likelihood weighting is to fix the evidence variables E and sample only the remaining variables X and Y. We have to be careful, however, as we are biasing the sampling process to follow a different distribution that the true joint distribution. This time, each sample also stores a weight. The weight expresses the probability the sample could have been produced if we had not fixed the evidence variables. Initially the value of the weight is 1 and each time we assign the value of an evidence variable, we use the probability of this assignment to multiply with the existing weight of the sample.

Consider, for example, that we want to sample atomic events with likelihood weighting so as to compute the conditional probability: $P(Rain|Sprinkler = true, WetGrass = true)$. We show here the generation of a specific sample:

- Initially the weight of the sample is set to 1: $w = 1$.
- Then we sample the value of the random variable “Cloudy”. $P(Cloudy) =< 0.5, 0.5 >$. Assume that sampling returns: “Cloudy”= true.
- “Sprinkler” is an evidence variable with value true. For the evidence variables we have to update the weight:

$$w \leftarrow w \times P(Sprinkler = ture|Cloudy = true) = 0.1$$

- Then we sample the value of “Rain” according to the appropriate distribution: $P(Rain|Cloudy = True) =< 0.8, 0.2 >$. Assume that it returns true.
- “WetGrass” is an evidence variable with value true and we update the weight:

$$w \leftarrow w \times P(WetGrass = true|Sprinkler = true, Rain = true) = 0.099$$

This process will return the sample $< true, true, true, true >$ with weight 0.099. The question that arises is whether this process generates a consistent estimate. In order to be consistent it has to conform to the limit to true distribution.

Notice, however, that the sampling process generates atomic events that follow the distribution:

$$S(z, e) = \prod_{i=1}^{l} P(z_i|Parents(z_i))$$

where $Z = X \cup Y$, is the set of all the variables but the evidence variables. On the other hand, the weights represent:

$$w(z, e) = \prod_{i=1}^{m} P(e_i|Parents(E_i))$$

Consequently, the combination of the weights together with the samples themselves express the following distribution:

$$S(z, e) \cdot w(z, e) = \prod_{i=1}^{l} P(Z_i|Parents(Z_i)) \cdot \prod_{i=1}^{m} P(e_i|Parents(E_i)) = P(z, e)$$

$$\begin{aligned}\hat{P}(x|e) &= \alpha \textstyle\sum_y N(x,y,e)\cdot w(x,y,e)\\ &\approx \alpha \textstyle\sum_y S(x,y,e)\cdot w(x,y,e) \qquad \text{as } N \text{ approaches infinity}\\ &= \alpha' \textstyle\sum_y P(x,y,e)\\ &= \alpha' P(x,e)\\ &= P(x|e)\end{aligned}$$

which is the joint probability distribution.

The estimated posterior probability is actually:

and the estimate of the likelihood weighting is also consistent.

This approach becomes problematic as the number of evidence variables increases and we are sampling from an increasingly smaller subset of the full joint probability distribution. In this case, samples will have very low weights, and the sampled population will be biased by the small fraction of samples that accord to the evidence variables.

1.2 Markov Chain Simulation

The idea behind Markov Chain Monte Carlo (MCMC) approaches is different from that of the direct sampling family. In MCMC, we do not create each sample from scratch. Instead we change the previous sample. To achieve that, we sample a value for one of the non-evidence variables X_i. But since all the neighbors in the Bayesian network of this variable X_i already have assignments, we must take their values into accoount before sampling X_i. Figure 2 shows the variables that we actually have to take into account in this process. This set of variables for X_i is called the "Markov blanket" of X_i. It corresponds to the parents of X_i, the children and to other parents of X_i's children.

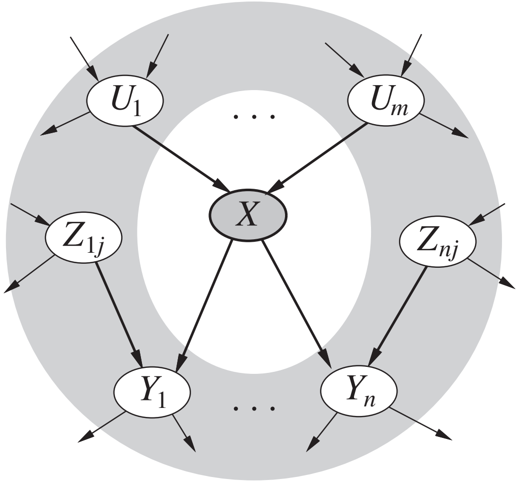

Figure 2: The Markov Blanket of variable X.

Let's use the MCMC approach to compute the probability: $P(Rain|Sprinkler = true, WetGrass = true)$. Initially we sample randomly the non-evidence variables "Rain" and "Cloudy". Let's assume that the sampling process returns true for both variables. Consequently, the initial state of the MCMC process is the atomic event: $< true, true, false, true >$. Then we can have the following iterations of the algorithm:

1. At each iteration we can sample the value of non-evidence variables, consequently we will sample either "Rain" or "Cloudy". Let's say we pick "Cloudy" and then we sample it, given its Markov blanket ("Sprinkler"

and "Rain"). Assume that the sampling process returns: false. Then we have created a new atomic event $< false, true, false, true >$.

2. Let's say that during the second iteration we pick "Rain". Then we sample this variable according to its Markov blanket (all the remaining variables). Assume that the result of the sampling process is: "Rain"=true. Then the new state that we have produced is: $< false, true, true, true >$.

In order to compute now the probability $P(Rain|Sprinkler = true, WetGrass = true)$, we just count the number of atomic events that have "Rain"=true. The important property of the MCMC approach is the following:

The sampling process settles into a "dynamic equilibrium",
where the fraction of time spent in each state is proportional to its posterior probability.

All of the above approaches deal with static Bayesian networks and problems were over time the state does not change. The following lectures will focus on state estimation problems, were the state evolves with time. These problems are referred to as temporal state estimation problems.