Study Notes for AI (CS 440/520)
Lecture 11: Bayesian Networks and Exact Inference

Corresponding Book Chapters: 14.1-14.2-14.3
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 Bayesian Networks

If we have available the full joint probability distribution of a problem, then we can infer the probability of any event that involves the variables in the joint distribution. This is typically, however, a very inefficient way to compute the probabilities of variables, as it requires a lot of knowledge about the world.

Independence and conditional independence properties can assist in reducing the number of probabilities that we have to know in order to answer probabilistic queries. This lecture will cover Bayesian networks, a graphical way to represent dependencies between variables and hopefully allow us to detect independencies.

1.1 Formalization

A Bayesian network is a graph where:

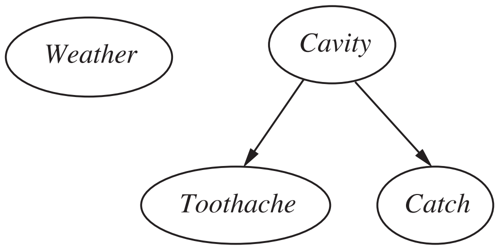
- the nodes correspond to the random variables of a problem,
- directed links connect pairs of nodes so that:

$$X \rightarrow Y \quad \text{(X is a parent of Y)} \qquad \text{if and only if X has direct influence on Y,}$$

- each X_i has a conditional probability distribution. The conditional distribution expresses the effects of the parents to the node:
$$P(X_i|Parents(X_i)),$$
- the graph has no directed cycles, hence it is a directed, acyclic graph or a `DAG`.

Figure 1 provides an example of a Bayesian network for a dentist related problem. We have three random variables: (i) cavity: which expresses the probability of truly having a cavity, (ii) toothache: the probability of suffering from toothache and (iii) catch: the probability that the dentist's tool "catches" on to a tooth, that is it detects a cavity. The Bayesian network shows that each of the last two variables, toothache and catch, depend only on the first one, the cavity variable. An independent random variable, weather, is not connected to the random variables of the dentist problem. The advantage of a Bayesian network is that it explicitly represents the independence and conditional independence properties of the problem.

The figure also provides the joint probability distribution for the dentist problem. Notice than in order to define the joint distribution we need 8 variables. On the other hand, the Bayesian network stores only 5 probabilities:



	touthache		¬touthache	
	catch	¬catch	catch	catch
cavity	0.108	0.012	0.072	0.008
¬cavity	0.016	0.064	0.144	0.576

Figure 1: An example Bayesian network and its full joint probability distribution.

$P(Cavity), P(Toothache|Cavity), P(Toothache, |\neg Cavity), P(Catch|Cavity), P(Catch|\neg Cavity)$. For larger problems, the difference between the number of probabilities stored in the joint distribution and the Bayesian network is typically larger.

1.2 Equivalence with full joint probability distribution

We assert at this point that a Bayesian network is actually able to answer any probabilistic query that can be answered by the joint distribution. The following computations explain why by applying repetitively the product rule ($P(a, b) = P(b|a)P(a)$):

$$\begin{aligned} P(X_1, \ldots, X_n) &= P(X_n|X_{n-1}, \ldots, X_1) \cdot P(X_{n-1}, \ldots, X_1) \\ &= P(X_n|X_{n-1}, \ldots, X_1) \cdot P(X_{n-1}|X_{n-2}, \ldots, X_1) \cdot \ldots \cdot P(X_2|X_1) \cdot P(X_1) \\ &= \prod_{i=1}^{n} P(X_i|X_{i-1}, \ldots, X_1) \end{aligned}$$

If the Bayesian network correctly represents the independence properties of the problem then the only nodes that influence each random variable X_i are the parent random variables $Parents(X_i)$ in the Bayesian network. Consequently, the following is true:

$$P(X_i|X_{i-1}, \ldots, X_1) = P(X_i|Parents(X_i))$$

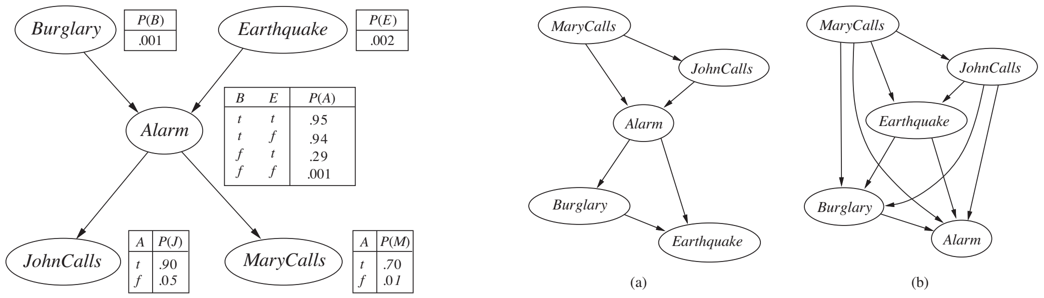
[Notice, that for the above statement to be true there is an additional requirement: $Parents(X_i) \subset \{X_{i-1}, \ldots, X_1\}$. This requirement can be easily achieved with an appropriate partial ordering of the DAG structure of the Bayesian network.]

Consequently, we have that the joint distribution can be expressed as a product of the conditional probabilities stored on the Bayesian network:

$$P(X_1, \ldots, X_n) = \prod_{i=1}^{n} P(X_i|Parents(X_i))$$

Since the joint distribution can be used to answer every probabilistic query involving its random variables, then so does the information available on the Bayesian network of those variables.

The difference is that the information on the Bayesian network is much more compact and manageable. Consider, for example, a problem with $n = 30$ nodes, each with five parents ($k = 5$). Then the Bayesian network needs $n \cdot 2^k$ probabilities to answer all queries (that is 960 numbers for our example), instead of 2^n probabilities (=1 billion numbers) that we have to specify for the joint probability distribution.



P(B)
.001

P(E)
.002

B	E	P(A)
t	t	.95
t	f	.94
f	t	.29
f	f	.001

A	P(J)
t	.90
f	.05

A	P(M)
t	.70
f	.01

Figure 2: A more complicated Bayesian network.

1.3 The order matters

Now consider an example which is just a little more complex. You have a new burglar alarm installed at home. It is fairly reliable at detecting a burglary, but also responds on occasion to minor earthquakes. You also have two neighbors, John and Mary, who have promised to call you at work when they hear the alarm. John always calls when he hears the alarm, but sometimes confuses the telephone ringing with the alarm and calls then, too. Mary, on the other hand, likes rather loud music and sometimes misses the alarm altogether. Given the evidence of who has or not called, we would like to estimate the probability of a burglary.

Figure 2 (left) shows one possible way of representing the above problem with a Bayesian network. This Bayesian network requires 10 probabilities in order to be able to answer any query. Notice, that we could have selected a different structure for the network. For example, one of the two structures on the right side of figure 2. The network to the far right side requires 31 probabilities to be specified - exactly the same as the full joint distribution. Any of the three networks represent exactly the same joint distribution. Nevertheless, the last two simply fail to represent all the conditional independence relationships and hence end up specifying a lot of unnecessary numbers instead.

2 Exact Inference in Bayesian Networks

We will use the following notation for this section:

- X: denotes a query random variable
- e: observed event over evidence variables: $E = E_1, \ldots, E_m$
- Y: hidden variables

Consequently, the complete set of variables involved in a problem is: $X \cup E \cup Y$, and we want to compute a probability of the form:

$$P(X|e)$$

For example, what is the probability that a burglary occurred if both John and Mary called?
$P(burglary|JohnCalls = true, MaryCalls = true)$.

2.1 Inference by enumeration

One possible way to approach exact inference in Bayesian networks is to take advantage of the fact that the Bayesian network is equivalent to full joint probability distribution. Thus, the approach first maps any query to the joint distribution as represented by the Bayesian network and then enumerates all the atomic events.

We will show how this approach works by using the burglary example. We will abbreviate the random variables, so in the following discussion: $b = burglary, j = JohnCalls, m = MaryCalls, e = Earthquake, a = alarm$.

$$\begin{aligned}
P(b|j,m) &= \alpha \cdot P(b,j,m) && \text{(product rule)}\\
&= \alpha \textstyle\sum_e \sum_a P(b,e,a,j,m) && \text{(conditioning)}\\
&= \alpha \textstyle\sum_e \sum_a P(b)P(e)P(a|b,e)P(j|a)P(m|a) && \text{(Bayesian network)}\\
&= \alpha P(b) \textstyle\sum_e P(e) \sum_a P(a|b,e)P(j|a)P(m|a) && \text{(moving terms out of the summations)}
\end{aligned}$$

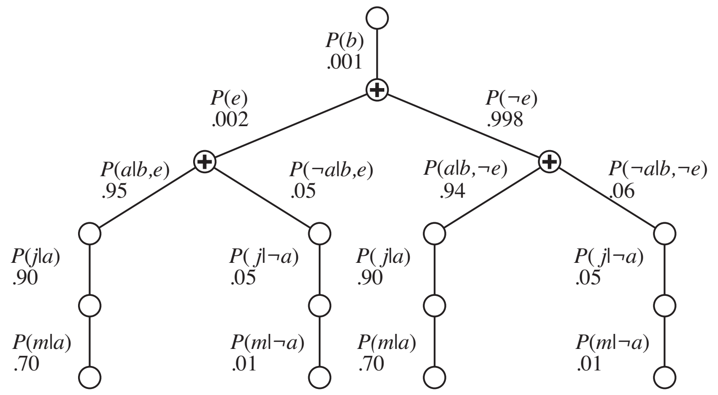


Figure 3: The enumeration tree for computing the probability $P(b|j,m)$.

For example, consider that we have to answer the conditional probability we referenced above:

The above steps first turn the query $P(b|j,m)$ into an expression that depends on the full joint probability distribution $[\alpha \sum_e \sum_a P(b,e,a,j,m)]$. Then we apply the rule that the joint distribution is equivalent to the product of all the conditional probabilities in the Bayesian network to acquire the final result. Notice that he final result contains summations. This means that we have to consider the various different values for the variables e and a and compute each time the products $P(a|b,e)P(j|a)P(m|a)$ for these values.

This computation can be achieved efficiently with a search based approach. Figure 3 shows the search tree that we can define to answer such problems. Each probability appears as a node on the tree. The summations appear as branching nodes, where the different branches consider different values for the random variable that the summation is defined over. Once we define the search tree, it is possible to compute the solution by running a depth-first search technique (the depth of the tree is limited).

It might be beneficial, however, to use a bottom-up approach instead of a top-down search procedure. Notice in figure 3 that parts of the four resulting branches have the same probabilities (e.g., the products $P(j|a)P(m|a)$ and $P(j|\neg a)P(m|\neg a)$ appear twice each). A bottom-up approach can potentially detect this fact and avoid recomputing products of probabilities that have been already computed.

The space complexity of the exact inference approach we describe is linear in the number of random variables (while it would have been exponential if we were depending on the full joint distribution). The time complexity if $O(2^n)$, which is better than the $O(n2^n)$ complexity if we have used the full joint distribution, but still very grim. There are alternative ways to achieve exact inference in Bayesian networks (variable elimination) that tend to be more efficient but still have exponential dependency in the worst case. Thus, in general, exact inference in Bayesian networks is considered intractable. In the next lecture we will describe approximate inference techniques for Bayesian networks.

2.2 Variable Elimination

The enumeration algorithm can be improved substantially by eliminating repeated calculations, as those shown in Figure 3. The idea is simple: do the calculations once and save the results for later. Variable elimination evaluates probabilistic expressions by following a bottom-up approach along the enumeration tree. Intermediate results are stored and summations over each variable are done only for those portions of the expression that depend on the variable.

Consider again the burglary example:

$$P(B|j,m) = \alpha \underbrace{P(B)}_{f_1(B)} \sum_e \underbrace{P(e)}_{f_2(E)} \sum_a \underbrace{P(a|B,e)}_{f_3(A,B,E)} \underbrace{P(j|a)}_{f_4(A)} \underbrace{P(m|a)}_{f_5(A)}$$

Each part of the expression is annotated with the name of the corresponding factor. Each factor is a matrix indexed by the values of its argument variables. For example, the factors $f_4(A)$ and $f_5(A)$ depend just on A because J and M are fixed as evidence variables. They are therefore two-element vectors:

$$f_4(A) = \begin{pmatrix} P(j|a) \\ P(j|\neg a) \end{pmatrix} = \begin{pmatrix} 0.9 \\ 0.05 \end{pmatrix} \qquad f_5(A) = \begin{pmatrix} P(m|a) \\ P(m|\neg a) \end{pmatrix} = \begin{pmatrix} 0.70 \\ 0.01 \end{pmatrix}.$$

$f_3(A, B, E)$ will be a $2 \times 2 \times 2$ matrix and so on. When you multiple factors, you have to be careful, since this is a point-wise operation. The point-wise product of two factors f_1 and f_2 yields a new factor f whose variables are the union of the variables in f_1 and f_2 and whose elements are given by the product of the corresponding elements in the two factors. Suppose the two factors have variables $Y_1, Y_2, \ldots, Y_k$ in common, then we have:

$$f(X_1, \ldots, X_j, Y_1, \ldots, Y_k, Z_1, \ldots, Z_l) = f_1(X_1, \ldots, X_j, Y_1, \ldots, Y_k) f_2(Y_1, \ldots, Y_k, Z_1, \ldots, Z_l)$$

For example if $f_1(A = true, B = true) = 0.3$ and $f_2(B = true, C = true) = 0.2$, then $f_3(A = true, B = true, C = true) = 0.3 \times 0.2 = 0.06$.

Summing out a variable from a product of factors is done by adding up the sub-matrices formed by fixing the variable to each of its values in turn. Notice than any factor that does not depend on the variable to be summed out can be moved outside the summation. One way to further speed up variable elimination is to notice the following: "Every variable that is not an ancestor of a query variable or evidence variable is irrelevant to the query". The algorithm removes all these variables before evaluating the query.

Complexity The time and space requirements of variable elimination are dominated by the size of the largest factor constructed during the operation of the algorithm. This in turn is determined by the order of elimination of variables and by the structure of the network. It turns out to be intractable to determine the optimal ordering, but several good heuristics are available.

It also turns out that singly connected networks or poly-trees have a particularly nice property: "The time and space complexity of exact inference in poly-trees is linear in the size of the network". The size is defined as the number of entries in the tables representing the Conditional Probability Distributions on the network. In the general case, however, for multiply connected networks, exact inference has exponential complexity. It can actually be shown that the problem is strictly harder than NP-complete problems. Thus, in general, exact inference in Bayesian networks is considered intractable. In the next lecture we will describe approximate inference techniques for Bayesian networks.