

6.825 Techniques in Artificial Intelligence

Inference in Bayesian Networks

Lecture 16 • 1

Now that we know what the semantics of Bayes nets are; what it means when we have one, we need to understand how to use it. Typically, we'll be in a situation in which we have some evidence, that is, some of the variables are instantiated, and we want to infer something about the probability distribution of some other variables.

6.825 Techniques in Artificial Intelligence

Inference in Bayesian Networks

- Exact inference

Lecture 16 • 2

In exact inference, we analytically compute the conditional probability distribution over the variables of interest.

6.825 Techniques in Artificial Intelligence

Inference in Bayesian Networks

- Exact inference
- Approximate inference

Lecture 16 • 3

But sometimes, that's too hard to do, in which case we can use approximation techniques based on statistical sampling.

Query Types

Given a Bayesian network, what questions might we want to ask?

Given a Bayesian network, what kinds of questions might we want to ask?

Query Types

Given a Bayesian network, what questions might we want to ask?

- Conditional probability query: $P(x \mid e)$

The most usual is a conditional probability query. Given instantiations for some of the variables (we'll use e here to stand for the values of all the instantiated variables; it doesn't have to be just one), what is the probability that node X has a particular value x ?

Query Types

Given a Bayesian network, what questions might we want to ask?

- Conditional probability query: $P(x | e)$
- Maximum a posteriori probability:
What value of x maximizes $P(x|e)$?

Lecture 16 • 6

Another interesting question you might ask is, what is the most likely explanation for some evidence? We can think of that as the value of node X (or of some group of nodes) that maximizes the probability that you would have seen the evidence you did. This is called the maximum a posteriori probability or MAP query.

Query Types

Given a Bayesian network, what questions might we want to ask?

- Conditional probability query: $P(x | e)$
- Maximum a posteriori probability:
What value of x maximizes $P(x|e)$?

General question: What's the whole probability distribution over variable X given evidence e , $P(X | e)$?

Lecture 16 • 7

In our discrete probability situation, the only way to answer a MAP query is to compute the probability of x given e for all possible values of x and see which one is greatest.

Query Types

Given a Bayesian network, what questions might we want to ask?

- Conditional probability query: $P(x | e)$
- Maximum a posteriori probability:
What value of x maximizes $P(x|e)$?

General question: What's the whole probability distribution over variable X given evidence e , $P(X | e)$?

Lecture 16 • 8

So, in general, we'd like to be able to compute a whole probability distribution over some variable or variables X , given instantiations of a set of variables e .

Using the joint distribution

To answer any query involving a conjunction of variables, sum over the variables not involved in the query.

Given the joint distribution over the variables, we can easily answer any question about the value of a single variable by summing (or marginalizing) over the other variables.

Using the joint distribution

To answer any query involving a conjunction of variables, sum over the variables not involved in the query.

$$\begin{aligned}\Pr(d) &= \sum_{ABC} \Pr(a,b,c,d) \\ &= \sum_{a \in \text{dom}(A)} \sum_{b \in \text{dom}(B)} \sum_{c \in \text{dom}(C)} \Pr(A = a \wedge B = b \wedge C = c)\end{aligned}$$

So, in a domain with four variables, A, B, C, and D, the probability that variable D has value d is the sum over all possible combinations of values of the other three variables of the joint probability of all four values. This is exactly the same as the procedure we went through in the last lecture, where to compute the probability of cavity, we added up the probability of cavity and toothache and the probability of cavity and not toothache.

Using the joint distribution

To answer any query involving a conjunction of variables, sum over the variables not involved in the query.

$$\begin{aligned}\Pr(d) &= \sum_{ABC} \Pr(a,b,c,d) \\ &= \sum_{a \in \text{dom}(A)} \sum_{b \in \text{dom}(B)} \sum_{c \in \text{dom}(C)} \Pr(A = a \wedge B = b \wedge C = c)\end{aligned}$$

In general, we'll use the first notation, with a single summation indexed by a list of variable names, and a joint probability expression that mentions values of those variables. But here we can see the completely written-out definition, just so we all know what the shorthand is supposed to mean.

Using the joint distribution

To answer any query involving a conjunction of variables, sum over the variables not involved in the query.

$$\begin{aligned}\Pr(d) &= \sum_{ABC} \Pr(a,b,c,d) \\ &= \sum_{a \in \text{dom}(A)} \sum_{b \in \text{dom}(B)} \sum_{c \in \text{dom}(C)} \Pr(A = a \wedge B = b \wedge C = c)\end{aligned}$$

$$\Pr(d | b) = \frac{\Pr(b,d)}{\Pr(b)} = \frac{\sum_{AC} \Pr(a,b,c,d)}{\sum_{ACD} \Pr(a,b,c,d)}$$

To compute a conditional probability, we reduce it to a ratio of conjunctive queries using the definition of conditional probability, and then answer each of those queries by marginalizing out the variables not mentioned.

Using the joint distribution

To answer any query involving a conjunction of variables, sum over the variables not involved in the query.

$$\begin{aligned}\Pr(d) &= \sum_{ABC} \Pr(a,b,c,d) \\ &= \sum_{a \in \text{dom}(A)} \sum_{b \in \text{dom}(B)} \sum_{c \in \text{dom}(C)} \Pr(A = a \wedge B = b \wedge C = c)\end{aligned}$$

$$\Pr(d | b) = \frac{\Pr(b,d)}{\Pr(b)} = \frac{\sum_{AC} \Pr(a,b,c,d)}{\sum_{ACD} \Pr(a,b,c,d)}$$

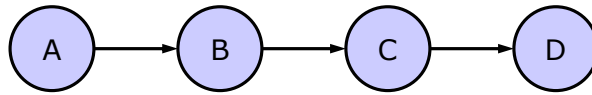
In the numerator, here, you can see that we're only summing over variables A and C, because b and d are instantiated in the query.

Simple Case

Lecture 16 • 14

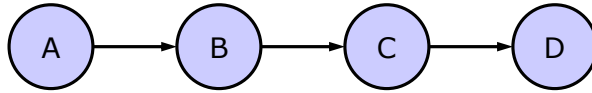
We're going to learn a general purpose algorithm for answering these joint queries fairly efficiently. We'll start by looking at a very simple case to build up our intuitions, then we'll write down the algorithm, then we'll apply it to a more complex case.

Simple Case



Okay. Here's our very simple case. It's a bayes net with four nodes, arranged in a chain.

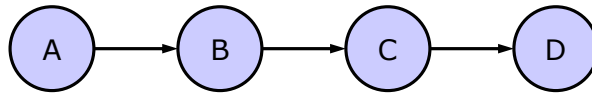
Simple Case



$$\Pr(d) = \sum_{ABC} \Pr(a, b, c, d)$$

So, we know from before that the probability that variable D has some value little d is the sum over A, B, and C of the joint distribution, with d fixed.

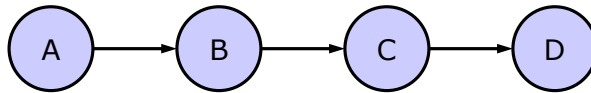
Simple Case



$$\begin{aligned}\Pr(d) &= \sum_{ABC} \Pr(a, b, c, d) \\ &= \sum_{ABC} \Pr(d | c) \Pr(c | b) \Pr(b | a) \Pr(a)\end{aligned}$$

Now, using the chain rule of Bayesian networks, we can write down the joint probability as a product over the nodes of the probability of each node's value given the values of its parents. So, in this case, we get $P(d|c)$ times $P(c|b)$ times $P(b|a)$ times $P(a)$.

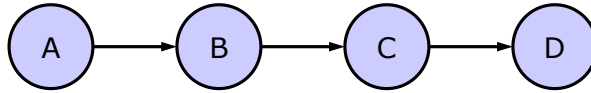
Simple Case



$$\begin{aligned}\Pr(d) &= \sum_{ABC} \Pr(a, b, c, d) \\ &= \sum_{ABC} \Pr(d | c) \Pr(c | b) \Pr(b | a) \Pr(a)\end{aligned}$$

This expression gives us a method for answering the query, given the conditional probabilities that are stored in the net. And this method can be applied directly to any other bayes net. But there's a problem with it: it requires enumerating all possible combinations of assignments to A, B, and C, and then, for each one, multiplying the factors for each node. That's an enormous amount of work and we'd like to avoid it if at all possible.

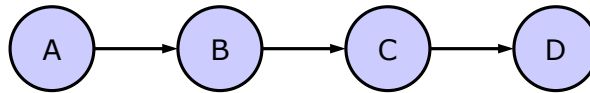
Simple Case



$$\begin{aligned}\Pr(d) &= \sum_{ABC} \Pr(a, b, c, d) \\ &= \sum_{ABC} \Pr(d | c) \Pr(c | b) \Pr(b | a) \Pr(a) \\ &= \sum_C \sum_B \sum_A \Pr(d | c) \Pr(c | b) \Pr(b | a) \Pr(a)\end{aligned}$$

So, we'll try rewriting the expression into something that might be more efficient to evaluate. First, we can make our summation into three separate summations, one over each variable.

Simple Case

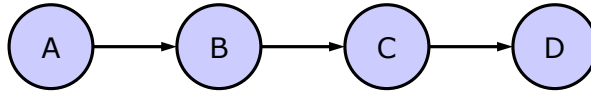


$$\begin{aligned}\Pr(d) &= \sum_{ABC} \Pr(a,b,c,d) \\ &= \sum_{ABC} \Pr(d | c) \Pr(c | b) \Pr(b | a) \Pr(a) \\ &= \sum_C \sum_B \sum_A \Pr(d | c) \Pr(c | b) \Pr(b | a) \Pr(a) \\ &= \sum_C \Pr(d | c) \sum_B \Pr(c | b) \sum_A \Pr(b | a) \Pr(a)\end{aligned}$$

Lecture 16 • 20

Then, by distributivity of addition over multiplication, we can push the summations in, so that the sum over A includes all the terms that mention A, but no others, and so on. It's pretty clear that this expression is the same as the previous one in value, but it can be evaluated more efficiently. We're still, eventually, enumerating all assignments to the three variables, but we're doing somewhat fewer multiplications than before. So this is still not completely satisfactory.

Simple Case

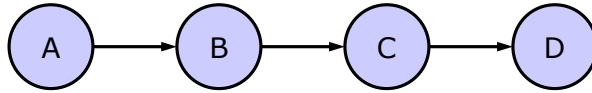


$$\Pr(d) = \sum_C \Pr(d | c) \sum_B \Pr(c | b) \underbrace{\sum_A \Pr(b | a) \Pr(a)}$$

$$\begin{bmatrix} \Pr(b_1 | a_1) \Pr(a_1) & \Pr(b_1 | a_2) \Pr(a_2) \\ \Pr(b_2 | a_1) \Pr(a_1) & \Pr(b_2 | a_2) \Pr(a_2) \end{bmatrix}$$

If you look, for a minute, at the terms inside the summation over A, you'll see that we're doing these multiplications over for each value of C, which isn't necessary, because they're independent of C. Our idea, here, is to do the multiplications once and store them for later use. So, first, for each value of A and B, we can compute the product, generating a two dimensional matrix.

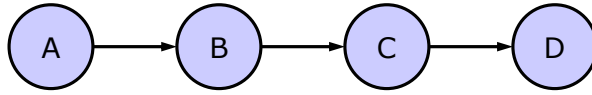
Simple Case



$$\Pr(d) = \sum_C \Pr(d | c) \sum_B \Pr(c | b) \underbrace{\sum_A \Pr(b | a) \Pr(a)}_{\left[\begin{array}{l} \sum \Pr(b_1 | a) \Pr(a) \\ \sum_A \Pr(b_2 | a) \Pr(a) \end{array} \right]}$$

Then, we can sum over the rows of the matrix, yielding one value of the sum for each possible value of b .

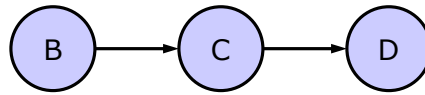
Simple Case



$$\Pr(d) = \sum_C \Pr(d | c) \sum_B \Pr(c | b) \underbrace{\sum_A \Pr(b | a) \Pr(a)}_{f_1(b)}$$

We'll call this set of values, which depends on b, f_1 of b.

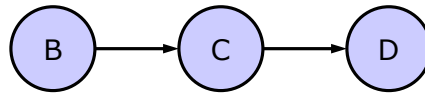
Simple Case



$$\Pr(d) = \sum_C \Pr(d | c) \underbrace{\sum_B \Pr(c | b) f_1(b)}_{f_2(c)}$$

Now, we can substitute f_1 of b in for the sum over A in our previous expression. And, effectively, we can remove node A from our diagram. Now, we express the contribution of b , which takes the contribution of a into account, as f_1 of b .

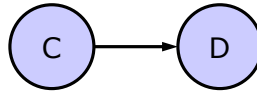
Simple Case



$$\Pr(d) = \sum_c \Pr(d | c) \underbrace{\sum_B \Pr(c | b) f_1(b)}_{f_2(c)}$$

We can continue the process in basically the same way. We can look at the summation over b and see that the only other variable it involves is c . We can summarize those products as a set of factors, one for each value of c . We'll call those factors f_2 of c .

Simple Case



$$\Pr(d) = \sum_c \Pr(d | c) f_2(c)$$

We substitute f_2 of c into the formula, remove node b from the diagram, and now we're down to a simple expression in which d is known and we have to sum over values of c .

Variable Elimination Algorithm

Given a Bayesian network, and an *elimination order* for the non-query variables

Lecture 16 • 27

That was a simple special case. Now we can look at the algorithm in the general case. Let's assume that we're given a Bayesian network and an ordering on the variables that aren't fixed in the query. We'll come back later to the question of the influence of the order, and how we might find a good one.

Variable Elimination Algorithm

Given a Bayesian network, and an *elimination order* for the non-query variables, compute

$$\sum_{X_1} \sum_{X_2} \dots \sum_{X_m} \prod_j \Pr(x_j \mid Pa(x_j))$$

We can express the probability of the query variables as a sum over each value of each of the non-query variables of a product over each node in the network, of the probability that that variable has the given value given the values of its parents.

Variable Elimination Algorithm

Given a Bayesian network, and an *elimination order* for the non-query variables, compute

$$\sum_{X_1} \sum_{X_2} \dots \sum_{X_m} \prod_j \Pr(x_j \mid Pa(x_j))$$

For $i = m$ downto 1

So, we'll eliminate the variables from the inside out. Starting with variable X_m and finishing with variable X_1 .

Variable Elimination Algorithm

Given a Bayesian network, and an *elimination order* for the non-query variables, compute

$$\sum_{X_1} \sum_{X_2} \dots \sum_{X_m} \prod_j \Pr(x_j \mid Pa(x_j))$$

For $i = m$ downto 1

- remove all the factors that mention X_i

To eliminate variable X_i , we start by gathering up all of the factors that mention X_i , and removing them from our set of factors. Let's say there are k such factors.

Variable Elimination Algorithm

Given a Bayesian network, and an *elimination order* for the non-query variables, compute

$$\sum_{X_1} \sum_{X_2} \dots \sum_{X_m} \prod_j \Pr(x_j \mid Pa(x_j))$$

For $i = m$ downto 1

- remove all the factors that mention X_i
- multiply those factors, getting a value for each combination of mentioned variables

Now, we make a $k+1$ dimensional table, indexed by X_i as well as each of the other variables that is mentioned in our set of factors.

Variable Elimination Algorithm

Given a Bayesian network, and an *elimination order* for the non-query variables, compute

$$\sum_{X_1} \sum_{X_2} \dots \sum_{X_m} \prod_j \Pr(x_j \mid Pa(x_j))$$

For $i = m$ downto 1

- remove all the factors that mention X_i
- multiply those factors, getting a value for each combination of mentioned variables
- sum over X_i

We then sum the table over the X_i dimension, resulting in a k -dimensional table.

Variable Elimination Algorithm

Given a Bayesian network, and an *elimination order* for the non-query variables, compute

$$\sum_{X_1} \sum_{X_2} \dots \sum_{X_m} \prod_j \Pr(x_j \mid Pa(x_j))$$

For $i = m$ downto 1

- remove all the factors that mention X_i
- multiply those factors, getting a value for each combination of mentioned variables
- sum over X_i
- put this new factor into the factor set

This table is our new factor, and we put a term for it back into our set of factors.

Variable Elimination Algorithm

Given a Bayesian network, and an *elimination order* for the non-query variables, compute

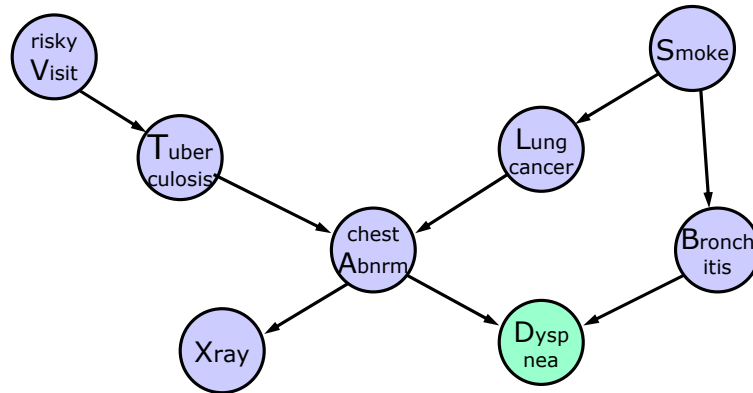
$$\sum_{X_1} \sum_{X_2} \dots \sum_{X_m} \prod_j \Pr(x_j \mid Pa(x_j))$$

For $i = m$ downto 1

- remove all the factors that mention X_i
- multiply those factors, getting a value for each combination of mentioned variables
- sum over X_i
- put this new factor into the factor set

Once we've eliminated all the summations, we have the desired value.

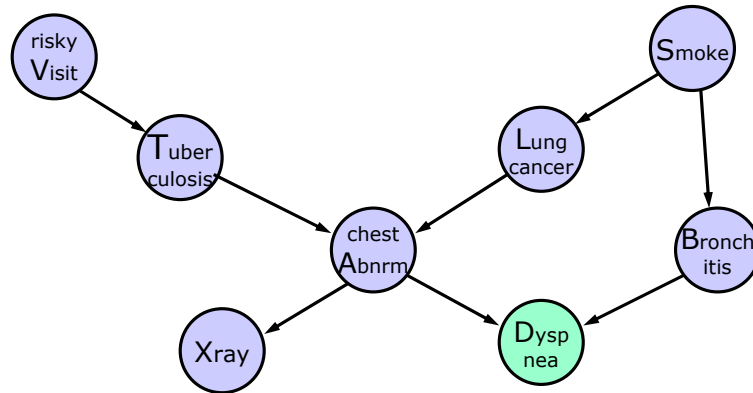
One more example



Lecture 16 • 35

Here's a more complicated example, to illustrate the variable elimination algorithm in a more general case. We have this big network that encodes a domain for diagnosing lung disease. (Dyspnea, as I understand it, is shortness of breath).

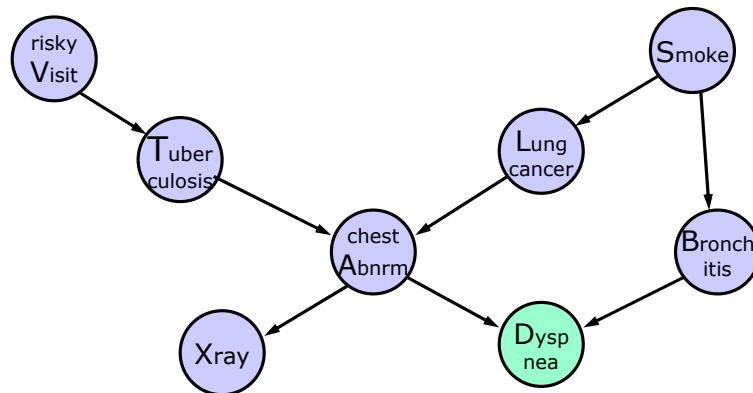
One more example



$$\Pr(d) = \sum_{A,B,L,T,S,X,V} \Pr(d | a,b) \Pr(a | t,l) \Pr(b | s) \Pr(l | s) \Pr(s) \Pr(x | a) \Pr(t | v) \Pr(v)$$

We'll do variable elimination on this graph using elimination order A, B, L, T, S, X, V.

One more example



$$\Pr(d) = \sum_{A,B,L,T,S,X} \Pr(d | a,b) \Pr(a | t,l) \Pr(b | s) \Pr(l | s) \Pr(s) \Pr(x | a) \underbrace{\sum_V \Pr(t | v) \Pr(v)}_{f_1(t)}$$

Lecture 16 • 37

So, we start by eliminating V . We gather the two terms that mention V and see that they also involve variable T . So, we compute the product for each value of T , and summarize those in the factor f_1 of T .

Kcanavan
MigrationConfirmed set by Kcanavan

