

CS221 Lecture notes #11

Bayesian networks

1 Probability Review

This material on Bayesian networks (Bayes nets) will rely heavily on several concepts from probability theory, and here we give a very brief review of these concepts. For more complete coverage, see Chapter 13 of the class textbook.

Although the Bayes net framework can accommodate continuous random variables, we will limit ourselves to discrete random variables X which can take on a finite set of values x^1, \dots, x^d . In general, we will use uppercase letters to denote random variables and lowercase letters to denote the values those variables may take on. The probability that X takes the value x will be denoted $P(X = x)$, or when there is no risk of ambiguity, $P(x)$. The **joint distribution** over n random variables X_1, \dots, X_n encodes the probability of a particular assignment to all of the variables, i.e. $P(X_1 = x_1, \dots, X_n = x_n)$, or simply $P(x_1, \dots, x_n)$.

The **conditional probability** that a random variable X takes on the value x given some other random variable Y takes on the value y is written $P(x | y)$, and is defined as:

$$P(x | y) = \frac{P(x, y)}{P(y)}.$$

More generally, for a set of random variables X_1, \dots, X_m and Y_1, \dots, Y_n , we can write:

$$P(x_1, \dots, x_m | y_1, \dots, y_n) = \frac{P(x_1, \dots, x_m, y_1, \dots, y_n)}{P(y_1, \dots, y_n)}.$$

We will use bold letters to denote sets of random variables and the values they might take. If $\mathbf{X} = \{X_1, \dots, X_m\}$ and $\mathbf{Y} = \{Y_1, \dots, Y_n\}$, we can rewrite

this definition as:

$$P(\mathbf{x} \mid \mathbf{y}) = \frac{P(\mathbf{x}, \mathbf{y})}{P(\mathbf{y})}.$$

We refer to the quantity $P(\mathbf{Y} = \mathbf{y})$ as the **marginal probability** of \mathbf{Y} when we want to emphasize that we are ignoring \mathbf{X} . If we are given the joint distribution over two sets of random variables \mathbf{X} and \mathbf{Y} , and \mathbf{x} and \mathbf{y} denote joint assignments to \mathbf{X} and \mathbf{Y} , we can retrieve the marginal probability of \mathbf{Y} by **marginalizing** over all of the possible assignments to \mathbf{X} , i.e.,

$$P(\mathbf{y}) = \sum_{\mathbf{x}} P(\mathbf{x}, \mathbf{y}).$$

By plugging this equation into our definition of conditional probability, we get **Bayes' Rule**:

$$P(\mathbf{x} \mid \mathbf{y}) = \frac{P(\mathbf{x}, \mathbf{y})}{\sum_{\mathbf{x}'} P(\mathbf{x}', \mathbf{y})}.$$

Two sets of random variables \mathbf{X} and \mathbf{Y} are **independent** if

$$P(\mathbf{x}, \mathbf{y}) = P(\mathbf{x})P(\mathbf{y}).$$

By dividing both sides through by $P(\mathbf{y})$, we see that this definition is equivalent to

$$P(\mathbf{x} \mid \mathbf{y}) = P(\mathbf{x}).$$

More generally, we say two sets of random variables \mathbf{X} and \mathbf{Y} are **conditionally independent** given a third set of random variables \mathbf{Z} if

$$P(\mathbf{x}, \mathbf{y} \mid \mathbf{z}) = P(\mathbf{x} \mid \mathbf{z})P(\mathbf{y} \mid \mathbf{z}),$$

or equivalently,

$$P(\mathbf{x} \mid \mathbf{y}, \mathbf{z}) = P(\mathbf{x} \mid \mathbf{z}).$$

Finally, it is worth noting the **chain rule** for joint probabilities. If \mathbf{X} and \mathbf{Y} are two sets of random variables, we can simply multiply both sides by $P(\mathbf{y})$ in the definition of conditional probability to find that

$$P(\mathbf{x}, \mathbf{y}) = P(\mathbf{x} \mid \mathbf{y})P(\mathbf{y}).$$

If we apply this formula repeatedly, we find that for any sets of random variables $\mathbf{X}_1, \dots, \mathbf{X}_n$,

$$P(\mathbf{x}_1, \dots, \mathbf{x}_n) = \prod_{i=1}^n P(\mathbf{x}_i \mid \mathbf{x}_1, \dots, \mathbf{x}_{i-1}).$$

2 Motivation

When we studied constraint satisfaction problems and propositional satisfiability, we assumed we had a set of “hard” constraints on the world. The CSP formalism only distinguished between those assignments to variables which were possible and those which were impossible. However, we’re often interested in cases where many assignments are possible, but only a subset of them are likely. For instance, imagine you are a doctor, and a patient comes to you with a cough, a sneeze, and red eyes. One possibility is that the patient has the flu; this would account for all three symptoms. Another possibility is that the patient simultaneously has the flu (which accounts for the sneezing), lung cancer (which accounts for the cough), and a corneal ulcer (which accounts for the red eyes). Strictly speaking, both of these possibilities are consistent with our observations of the patient’s symptoms. The latter situation is clearly unlikely, however, and we want to be able to treat the two differently. In order to do this, we need a way to encode uncertainty.

We will do this by giving a probability distribution over all possible states of the world. In our medical diagnosis problem, suppose we represent the state of the world with three binary random variables: flu (F), allergy (A), and sinus (S). We say F takes the value f if the patient has the flu, and it takes the value \bar{f} otherwise. We can represent the **joint distribution** over these three variables with a table of 8 numbers, each one representing the probability of a certain assignment to the three variables. For now, we assume no restrictions on the joint distribution other than that the probabilities sum to 1.

F	A	S	
f	a	s	0.027
f	a	\bar{s}	0.003
f	\bar{a}	s	0.162
f	\bar{a}	\bar{s}	0.108
\bar{f}	a	s	0.014
\bar{f}	a	\bar{s}	0.056
\bar{f}	\bar{a}	s	0.0063
\bar{f}	\bar{a}	\bar{s}	0.6237

With a joint distribution such as the one given above, we can answer any **query** about the domain. More specifically, we can answer any question about the probability of a particular assignment to one set of variables, possibly conditioned on the values of other variables. For example, suppose

we're interested in the probability that the patient has the flu ($F = f$) given that he has sinus trouble ($S = s$). To do this, we apply Bayes' Rule:

$$P(f | s) = \frac{P(f, s)}{P(f, s) + P(\bar{f}, s)}.$$

Note that we use $P(f | s)$ as a shorthand for $P(F = f | S = s)$, the probability that F takes the value f given that S takes the value s . To find $P(f, s)$, we add up the entries from the table which are consistent with that assignment, i.e.

$$P(f, s) = 0.027 + 0.162 = 0.184.$$

Similarly,

$$P(\bar{f}, s) = 0.014 + 0.0063 = 0.0203.$$

By plugging these numbers into Bayes' Rule, we get:

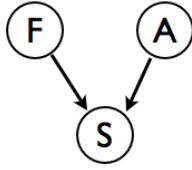
$$P(f | s) = \frac{0.184}{0.184 + 0.0203} = 0.903.$$

Therefore, we can conclude that the probability that the patient has the flu is about 90%.

Explicitly specifying the joint assignment in a table works for tiny examples such as the one above, but it doesn't scale because the size of the representation grows exponentially in the number of variables. Not only does this make actually computing the answers to queries very difficult, but a full joint distribution is unintuitive and hard for humans to specify exactly. Bayesian networks provide a compact and more computationally tractable way to represent joint distributions.

3 Bayesian network definition

A **Bayesian network (Bayes net)** is a directed acyclic graph, where nodes correspond to random variables and edges correspond to direct influence of one variable on another. Each node is associated with a **conditional probability table (CPT)** which gives the probability that the corresponding variable takes on a particular value given the values of its parents. For instance, we might use the following network to represent our medical diagnosis example:



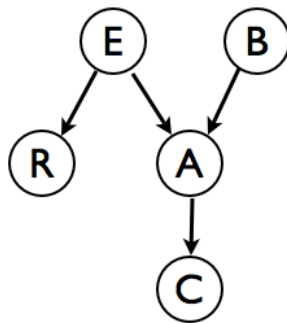
Intuitively, this encodes the information that sinus problems depend directly on having a flu or allergies. The CPTs might be:

$$P(F): \begin{array}{|c|c|} \hline f & \bar{f} \\ \hline 0.3 & 0.7 \\ \hline \end{array} \quad P(A): \begin{array}{|c|c|} \hline a & \bar{a} \\ \hline 0.1 & 0.9 \\ \hline \end{array}$$

$$P(S | F, A): \begin{array}{|c|c|c|c|} \hline & & s & \bar{s} \\ \hline f & a & 0.9 & 0.1 \\ f & \bar{a} & 0.6 & 0.4 \\ \bar{f} & a & 0.2 & 0.8 \\ \bar{f} & \bar{a} & 0.01 & 0.99 \\ \hline \end{array}$$

For example, this indicates that the probability of having allergies is 10% and the probability of having sinus trouble if one has allergies but not the flu is 20%.

Now let's turn to a more complicated example. Suppose an alarm is installed in your home, and the alarm (A) can be set off by an earthquake (E) or a burglar (B). If the alarm goes off, it might cause your neighbor to call (C). Finally, if there is an earthquake, it might be reported on the radio (R). All in all, we have five binary random variables: A , E , B , C , and R . We will represent this domain with the following Bayes net:



The burglary variable can be either true or false, but it isn't "caused" by any other variable in the network. The same goes for the earthquake. The alarm depends on whether or not there is an earthquake, and whether or not there is a burglary. The neighbor's call depends on the alarm, and the radio

depends on the earthquake. The parents of a node, intuitively, are the only things which directly influence that node.

We'll annotate this graph with the following probability distributions:

- $P(B)$ — the prior probability of a burglary.
- $P(E)$ — the prior probability of an earthquake.
- $P(A | B, E)$ — the probability of the alarm going off under any of the relevant circumstances: (b^1, e^1) , (b^1, e^0) , (b^0, e^1) , (b^0, e^0) .
- $P(C | A)$ — the probability of the neighbor's call for each value of A .
- $P(R | E)$ — the probability of the radio reporting an earthquake given each value of E .

In particular, here are our CPTs:

$$P(B): \begin{array}{|c|c|} \hline b^0 & b^1 \\ \hline 0.99 & 0.01 \\ \hline \end{array}$$

$$P(E): \begin{array}{|c|c|} \hline e^0 & e^1 \\ \hline 0.995 & 0.005 \\ \hline \end{array}$$

$$P(A | B, E): \begin{array}{|c|c|c|c|} \hline & & a^0 & a^1 \\ \hline b^0 & e^0 & 0.999 & 0.001 \\ b^0 & e^1 & 0.7 & 0.3 \\ b^1 & e^0 & 0.2 & 0.8 \\ b^1 & e^1 & 0.05 & 0.95 \\ \hline \end{array}$$

$$P(R | E): \begin{array}{|c|c|c|} \hline & r^0 & r^1 \\ \hline e^0 & 0.99999 & 0.00001 \\ e^1 & 0.65 & 0.35 \\ \hline \end{array}$$

$$P(C | A): \begin{array}{|c|c|c|} \hline & c^0 & c^1 \\ \hline a^0 & 0.95 & 0.05 \\ a^1 & 0.3 & 0.7 \\ \hline \end{array}$$

Now, suppose we are given the state $(b^1, e^0, a^1, c^1, r^0)$. What is the probability of this exact state? We define it as follows:

$$\begin{aligned} P(b^1, e^0, a^1, c^1, r^0) &= P(b^1)P(e^0)P(a^1 | b^1, e^0)P(c^1 | a^1)P(r^0 | e^0) \\ &= 0.1 \times 0.995 \times 0.8 \times 0.7 \times 0.99999 \\ &= 0.05572. \end{aligned}$$

We just multiplied together the corresponding entries of all of our conditional probability tables.

We can state this definition for general Bayesian networks as follows. Each node X_i associated with a CPT $P(X_i | \text{Parents}(X_i))$ which specifies a distribution over X_i for each combination of values for X_i 's parents. A

Bayesian network represents a joint probability distribution over its variables X_1, \dots, X_n via the **chain rule** for Bayes nets:

$$P(x_1, \dots, x_n) = \prod_{i=1}^n P(x_i \mid \text{Parents}(X_i)). \quad (1)$$

Bayesian networks give us a compact way of specifying the joint distribution over a set of variables. In our alarm network, we have five variables, each of which can take on one of two values. Therefore, explicitly representing the joint distribution over these variables in a table would require specifying 32 entries. The only constraint is that they all sum to 1, and so we need to specify 31 parameters. On the other hand, by representing the domain as a Bayes net, we only need to specify 10 parameters: 1 for $P(B)$, 1 for $P(E)$, 4 for $P(A \mid B, E)$, 2 for $P(C \mid A)$, and 2 for $P(R \mid E)$.

4 Reasoning patterns

We now discuss some particular kinds of queries we can pose using Bayes nets. Suppose we have two subsets of variables: \mathbf{Q} , the **query**, and \mathbf{E} , the **evidence**. We are interested in computing the probability of the query given the evidence. (For instance, we might be interested in computing the probability that there was a burglary (b^1) given that our neighbor did not call (c^0). Then $\mathbf{Q} = \{B\}$ and $\mathbf{E} = \{C\}$.) We have defined the joint probability of all of the variables in terms of the CPT entries for each of the nodes in the Bayes net. Therefore, in principle, we can answer this query by explicitly writing out the full joint distribution in a table, and then marginalizing out over all of the irrelevant variables. (We'll discuss better methods later in these notes.) In the example above, we can compute:

$$\begin{aligned} P(b^1 \mid c^0) &= \frac{P(b^1, c^0)}{P(c^0)} \\ &= \frac{\sum_{a,e,r} P(b^1, c^0, a, e, r)}{\sum_{a,e,r,b} P(c^0, a, e, r, b)} \end{aligned}$$

We'll now present some informal terms for various kinds of reasoning in Bayes nets:

- **Causal reasoning.** What is the chance that we get a phone call from our neighbor given that there was a burglary? In this case, the query is “downstream” of the evidence.

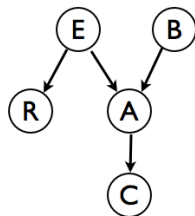


Figure 1: The alarm network. There are five nodes: E (earthquake), B (burglary), A (alarm), C (neighbor's call), and R (radio report of an earthquake).

- **Diagnostic or evidential reasoning.** Given that the alarm went off, what is the chance that there was a burglary? Here, we are trying to infer the probability of upstream events conditioned on downstream events.
- **Intercausal reasoning or explaining away.** Suppose your neighbor calls and informs you that your alarm went off. You are worried that there was a burglary. Then, you hear on the radio that there was an earthquake, and you're relieved because you figure the earthquake probably set off the alarm. We say the earthquake explained away the alarm. This is a very sophisticated form of reasoning, but we will see later that it fits nicely into the Bayes net framework.

5 Bayesian network semantics

We have seen that Bayes nets give a compact way of representing the joint distribution over a set of random variables. Specifying the full distribution over the five binary variables in our alarm network (shown again in Figure 1) by listing all of the individual probabilities in a table required specifying 31 parameters. Specifying the CPT entries for a Bayes net required only 10 parameters. Clearly, since it has fewer parameters to set, the latter approach can only specify a subset of the possible joint distributions over those five variables. We will now formalize precisely which subset of the joint distributions are consistent with a given Bayes net structure.

We will see that a Bayes net is basically encoding **conditional independence** assumptions about the variables in the network. Recall that we have committed to the following joint distribution over variables in the alarm

network:

$$P(B, E, A, C, R) = P(B)P(E)P(A | B, E)P(C | A)P(R | E),$$

or for the more general case, we have the **chain rule for Bayes nets**:

$$P(X_1, \dots, X_n) = \prod_{i=1}^n P(X_i | \text{Parents}(X_i)). \quad (2)$$

Any distribution consistent with the Bayes net must factorize in this way. In principle, we can find all of the conditional independencies in the network simply by performing algebraic manipulations on this expression. For instance, as an exercise, try to prove that B and E are independent in the alarm network. It is clearly tedious to try to uncover all of the conditional independencies through brute force algebraic manipulation. We will now develop higher-level sufficient and necessary conditions for conditional independence which allow us to read off conditional independencies directly from the graph structure.

First, let us recall the definition of conditional independence: Let \mathbf{X} , \mathbf{Y} , and \mathbf{Z} be (not necessarily disjoint) sets of random variables. (Recall that we use plaintext to denote random variables and their values, and boldface to denote sets of random variables and their possible joint assignments.) We define:

\mathbf{X} is **conditionally independent** of \mathbf{Y} given \mathbf{Z} if $P(\mathbf{x} | \mathbf{y}, \mathbf{z}) = P(\mathbf{x} | \mathbf{z})$
for all assignments \mathbf{x} , \mathbf{y} , and \mathbf{z} to \mathbf{X} , \mathbf{Y} , and \mathbf{Z} .

Equivalently, \mathbf{X} and \mathbf{Y} are conditionally independent given \mathbf{Z} if $P(\mathbf{x}, \mathbf{y} | \mathbf{z}) = P(\mathbf{x} | \mathbf{z})P(\mathbf{y} | \mathbf{z})$. As a shorthand, we often write this as $P(\mathbf{X}, \mathbf{Y} | \mathbf{Z}) = P(\mathbf{X} | \mathbf{Z})P(\mathbf{Y} | \mathbf{Z})$. We will also write $I(\mathbf{X}, \mathbf{Y} | \mathbf{Z})$ to denote that \mathbf{X} is conditionally independent of \mathbf{Y} given \mathbf{Z} .

Before we proceed to precisely define necessary and sufficient conditions for conditional independence, let us consider some simple cases. Figure 2 shows some examples where influence does or does not flow from one node X to another node Y . Make sure you understand why influence does or does not flow in each of these cases.

6 d-separation

Now we are going to formalize these intuitions. Suppose we have a three-variable path $X - Z - Y$ in our network. We say this path is **active** (influence “flows” from X to Z through Y) if one of the following holds:

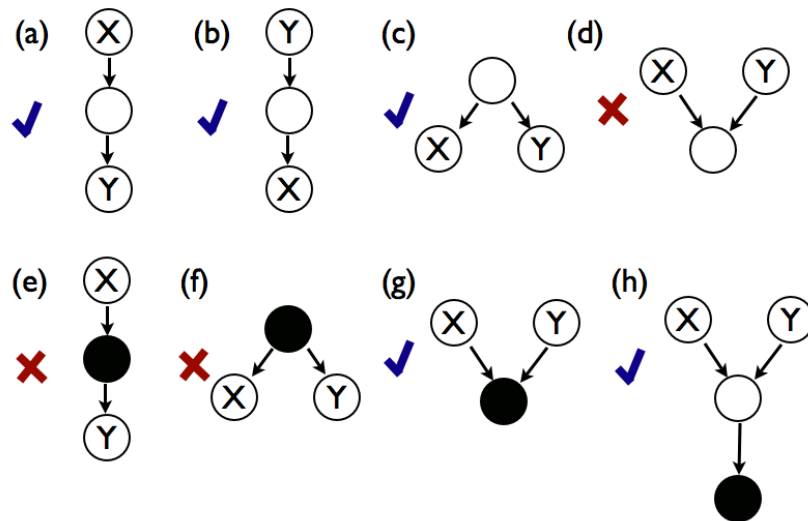


Figure 2: Some examples of conditional independence. In each of these networks, white nodes are unobserved and black nodes are observed. When influence flows from X to Y (i.e. X is not conditionally independent of Y given the observed nodes), the graph is marked with a check. Otherwise, it is marked with an X. (a) Knowing there was a burglary makes it more likely that our neighbor calls. (b) Knowing our neighbor called makes it more likely there was a burglary. (c) Knowing the alarm went off makes it more likely that an earthquake will be reported on the radio. (d) Knowing there was a burglary doesn't tell us anything about whether there was an earthquake. (e) Knowing there was an earthquake doesn't tell us anything about whether our neighbor calls, if we already know the alarm went off. (f) Knowing an earthquake was announced on the radio doesn't tell us anything about whether the alarm went off, if we already know there was an earthquake. (g) If we know the alarm went off, then knowing there was an earthquake "explains away" the alarm, making it less likely there was a burglary. (h) If we know our neighbor called, then knowing there was an earthquake explains away the call, making it less likely there was a burglary.

1. Z is not observed, and the graph structure is one of the following:

$$X \rightarrow Z \rightarrow Y, \quad X \leftarrow Z \rightarrow Y, \quad \text{or} \quad X \leftarrow Z \leftarrow Y.$$

2. The graph structure is a **V-structure** (i.e. $X \rightarrow Z \leftarrow Y$) and Z , or some descendant of Z , is observed.

Now let's generalize this definition to longer paths. A path X_1, X_2, \dots, X_k is **active** given a set of observed nodes \mathbf{Z} if:

1. For any V-structure $X_i \rightarrow X_{i+1} \leftarrow X_{i+2}$, either the vertex X_{i+1} is observed, or some descendant of X_{i+1} is observed.
2. No other node on the path is observed.

Two nodes X and Y are **d-separated** in a graph G given \mathbf{Z} if there is no active path between them in G . Note that we are defining d-separation in terms of the graph structure, rather than the underlying distribution. To connect the graph structure and the distribution, we will need the following theorem, which we state without proof:

Theorem 6.1: *Let P be a distribution that factors according to the Bayes net structure given by the graph G . Suppose two nodes X and Y are d-separated in G given a set of nodes \mathbf{Z} . Then X and Y are conditionally independent given \mathbf{Z} .*

Because of this theorem, we say d-separation is a **sound** mechanism for inferring conditional independence. I.e., if two nodes X and Y are d-separated, then they are necessarily conditionally independent given \mathbf{Z} . It turns out that d-separation is also an “almost complete” mechanism for inferring conditional independence, in a sense that we now explain. If there is an active path between two nodes X and Y given \mathbf{Z} , does this mean X and Y are necessarily conditionally dependent given \mathbf{Z} ? No, because we could assign probabilities in the CPTs in such a way that X and Y *happen* to be conditionally independent given \mathbf{Z} . However, the following is true:

Theorem 6.2: *If two nodes X and Y are not d-separated in G given \mathbf{Z} , there will be some choice of CPT entries such that X and Y are not independent given \mathbf{Z} .*

Hence, in a limited sense, d-separation is also **complete**. With d-separation, we can prove many statements about independencies directly from the graph structure, rather than by directly manipulating the terms in the definition (2).

7 Bayes ball

We have just specified necessary and sufficient conditions for two random variables X and Y being conditionally independent given some set of observed variables \mathbf{Z} . These conditions are useful in mathematical proofs about Bayes nets. However, we are sometimes interested in computing, for a particular graph, which variables are conditionally independent of which other variables.

We now present the **Bayes ball** algorithm, an efficient method for identifying all of the variables in the network which influence X .¹ We discuss the algorithm as if we were carrying it out by hand, but it is possible to formalize Bayes ball as an efficient dynamic programming algorithm. We imagine we have a ball which begins at node X , and which can travel anywhere that influence flows. If there is any way for the ball to travel from X to Y , then X and Y are conditionally dependent given \mathbf{Z} .² Otherwise, they are conditionally independent given \mathbf{Z} .

Now let's specify how the ball is allowed to bounce. We use the rules shown in Figure 3. As before, we use white circles to denote unobserved nodes and black circles to denote observed nodes. In cases where the ball may pass through, we draw red arrows going both directions; in cases where it cannot pass through, we draw arrows which turn around.

These rules have clear similarities with d-separation. For instance, the ball can pass through a path $X \rightarrow Z \rightarrow Y$ if and only if Z is unobserved. However, there is one subtle difference. Notice that the Bayes Ball rules for V-structures say nothing about whether a descendant of the vertex is observed; they only require knowing whether the particular nodes on the path are observed. Consider the problem of showing that R is conditionally dependent on B given C , as demonstrated in Figure 4. Using the properties of d-separation, we find the active path marked in Figure 4(a). However, the Bayes ball path will be the one given in Figure 4(b). Note that the rule which allows the ball to “bounce” back up when it hits C is the one for $X \rightarrow Z \leftarrow Y$, where Z is observed, and X and Y are identical. (In Bayes ball, unlike d-separation, we allow repeated nodes in the paths.) It is this locality of the rules which allows Bayes ball to efficiently discover the conditional dependencies in the graph.

¹This algorithm was not covered in the class lectures, and the material in this section is optional, and will not directly appear in CS221 homeworks/midterm/etc.

²More formally, there is some assignment to the CPT entries such that X and Y are conditionally dependent given \mathbf{Z} .

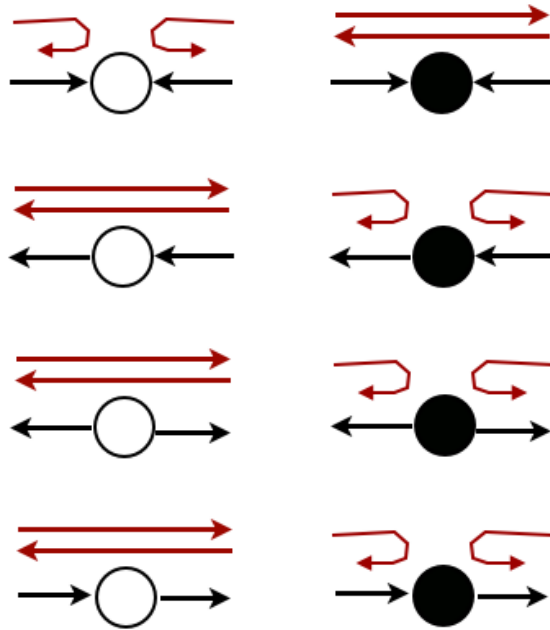


Figure 3: Rules for Bayes ball. We use white circles to denote unobserved nodes and black circles to denote observed nodes. In cases where the ball may pass through, we draw red arrows going both directions; in cases where it cannot pass through, we draw arrows which turn around.

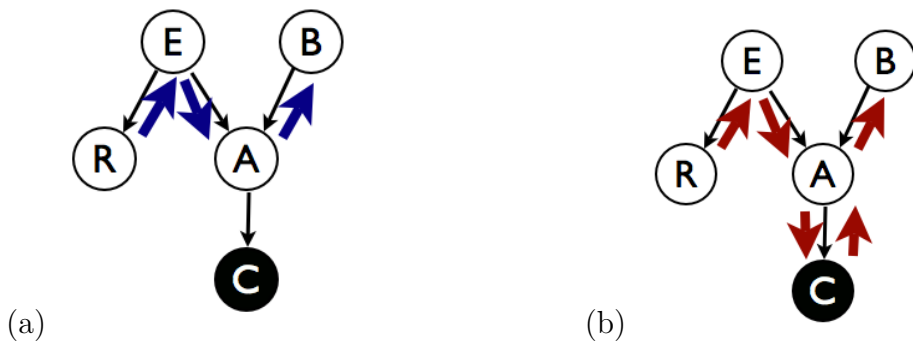


Figure 4: An example of the difference between d-separation and Bayes ball. (a) The active path from R to B given C . (b) The path the Bayes ball takes from R to B .

8 Bayes net inference

Given a Bayes net structure of a domain, we often want to perform **inference** on the Bayes net. Specifically, we might want to determine the conditional probability $P(q \mid \mathbf{e})$ that some **query** variable Q takes on a value q given that another set of variables \mathbf{E} , the **evidence** variables, has a joint assignment \mathbf{e} . We saw previously that we can, in principle, compute this conditional probability by explicitly writing out the full joint distribution over all of the variables in a big table, computing the entries using the chain rule for Bayes nets

$$P(x_1, \dots, x_n) = \prod_{i=1}^n P(x_i \mid \text{Parents}(X_i)),$$

and then applying the definition of conditional probability,

$$P(q \mid \mathbf{e}) = \frac{P(q, \mathbf{e})}{P(\mathbf{e})}.$$

However, the size of such a table would be exponential in the number of variables in the network, and therefore this algorithm is prohibitively expensive for all but the smallest networks. We will show how to use the structure of Bayes nets to perform inference more efficiently.

Suppose we have a two-variable Bayes net $A \rightarrow B$, where A and B are both binary random variables, and we want to compute the marginal probability of B , $P(B)$. We can apply our formula for marginal probability:

$$P(b^1) = P(a^0)P(b^1 \mid a^0) + P(a^1)P(b^1 \mid a^1). \quad (3)$$

Each of the terms is just one of the entries in the CPTs for the Bayes net. We can compute $P(b^0)$ similarly. Recall that we introduced a shorthand notation where uppercase variables in an equation signify that the equation must hold true for any value of the random variable. Using this notation, we can rewrite (3) as:

$$P(B) = P(a^0)P(B \mid a^0) + P(a^1)P(B \mid a^1).$$

More generally, when A is not binary, we have:

$$P(B) = \sum_a P(a)P(B \mid a^0). \quad (4)$$

If A and B can each take on k values, this sum can be computed in $O(k^2)$ time (i.e. $O(k)$ for each value in the domain of B).

Now let's consider a longer chain: $A \rightarrow B \rightarrow C$. We want to compute $P(C)$. If we knew the marginal distribution $P(B)$, we could compute $P(C)$ in the same way as above:

$$P(C) = \sum_b P(b)P(C | b).$$

But how do we get $P(B)$? We find it in the same way:

$$P(B) = \sum_a P(a)P(B | a).$$

Therefore, finding $P(C)$ only requires applying (4) twice. More generally, given a chain X_1, X_2, \dots, X_n , we apply (4) $n - 1$ times. Therefore, the total running time will be $n - 1$ times the running time for computing (4), or $O(nk^2)$. By contrast, if we had tried to compute this sum the naive way (by explicitly writing out the full joint distribution), we would have had to produce a table with $O(k^n)$ entries. We have gone from exponential time to linear time in the number of variables.

Let's formalize this process. Suppose we have the chain Bayes net $A \rightarrow B \rightarrow C \rightarrow D$, and we want to compute $P(D)$. By definition,

$$\begin{aligned} P(d) &= \sum_{a,b,c} P(a)P(b | a)P(c | b)P(d | c) \\ &= \sum_c \sum_b \sum_a P(a)P(b | a)P(c | b)P(d | c) \\ &= \sum_c P(d | c) \sum_b P(c | b) \sum_a P(a)P(b | a). \end{aligned} \quad (5)$$

Consider only the term $\sum_a P(a)P(b | a)$ in (5). The value of this term depends on the value of B . Therefore, we can rewrite the term as a **factor** $f_1(B)$, or a table of k numbers, one for each value of B . Specifically,

$$f_1(b) = \sum_a P(a)P(b | a).$$

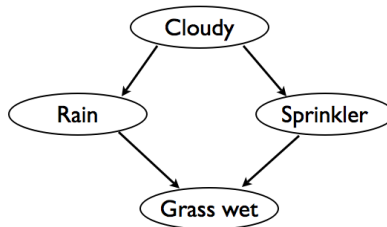
In this particular case, $f_1(b)$ turns out to be the marginal probability of $B = b$, but be aware that this won't hold true in general. *The factors we discuss here will not always correspond to marginal or conditional probabilities*, and we won't discuss their intuitive meaning any further. Once we have the factor $f_1(B)$, we can plug it into (5):

$$P(d) = \sum_c P(d | c) \sum_b P(c | b)f_1(b). \quad (6)$$

Just as we did before, we can compute the sum $\sum_b P(c | b)f_1(b)$ to get another factor $f_2(C)$, containing one number for each value of c . Finally, we plug $f_2(C)$ into (6) to get

$$P(d) = \sum_c P(d | c)f_2(c).$$

Let us consider one final example before we define variable elimination. Suppose we are trying to predict whether our neighbor's grass will be wet. Assume we have the following Bayes net structure:



If it is cloudy, it is more likely to rain. If our neighbors see that it is not cloudy, they are more likely to decide to turn on the sprinkler. Finally, either rain or the sprinkler being on can explain the wet grass.

Like before, we can express $P(W)$ as follows:

$$\begin{aligned} P(w) &= \sum_{r,s,c} P(w, r, s, c) \\ &= \sum_{r,s,c} P(w | r, s)P(r | c)P(s | c)P(c) \\ &= \sum_{r,s} P(w | r, s) \sum_c P(r | c)P(s | c)P(c). \end{aligned}$$

Let's first sum out the terms over C to get a factor $f_1(r, s)$, a table containing one value for each pair (r, s) . Then we compute:

$$P(w) = \sum_{r,s} P(w | r, s)f_1(r, s).$$

To summarize: the joint distribution of all of the variables in a Bayes net is defined by the chain rule for that Bayes net. By judiciously choosing subexpressions to compute first, we can avoid the exponential blowup that would occur if we tried to write out the entire joint distribution in a table.

8.1 Variable elimination

Now, let's generalize what we did in these examples into the **variable elimination** algorithm. Suppose we are trying to compute the marginal probability $P(q)$ of a query variable q .³

Let X_1, X_2, \dots, X_m be an ordering of the non-query variables (i.e. the variables other than q).

Consider the chain rule over the network structure:

$$\sum_{x_1} \sum_{x_2} \cdots \sum_{x_m} \prod_{j=1}^n P(x_j \mid \text{Parents}(X_j)).$$

For $i = 1, \dots, m$:

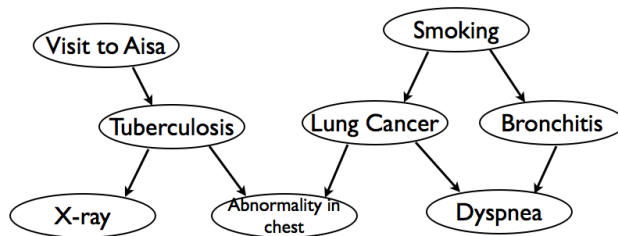
Leave in the summation for X_i only the factors which mention X_i .

Multiply out all of these factors, getting a factor f that contains a number for each of the possible joint assignments to the variables mentioned, including X_i .

Sum out the factor f over X_i , getting a factor f' which contains a number for each of the possible joint assignments, not including X_i .

Replace the sum $\sum_{x_i} \prod \cdots$ with the factor f' .

Now let's consider a more complex example of variable elimination. Consider the following network:



³We stated earlier that we often want to compute $P(q \mid \mathbf{E})$, the probability of the query given the evidence. Here, we suppose we do this by computing $P(q, \mathbf{E})$ and $P(\mathbf{E})$, and then taking the ratio. However, with small modifications, we can use variable elimination to compute $P(q \mid \mathbf{E})$ directly. This is often significantly faster than computing both $P(\mathbf{E})$ and $P(q, \mathbf{E})$.

Suppose we want to compute the probability of dyspnea. The joint distribution is defined as:

$$P(d^1) = \sum_{a,b,\ell,t,x,s,v} P(v)P(t | v)P(s)P(\ell | s)P(b | s)P(a | \ell, t)P(x | t)P(d^1 | l, b).$$

First, we eliminate V to get the factor

$$f_v(t) = \sum_V P(t | v)P(v).$$

For instance, if T can take on the values *no*, *mild*, or *severe*, f_V will be a factor with three numbers. In the next step, we eliminate S to get:

$$f_S(b, c) = \sum_s P(s)P(\ell | s)P(b | s).$$

If bronchitis has 3 values (*no*, *mild*, and *severe*) and lung cancer has 4, this gives a table of 12 numbers. We continue this process until we have eliminated all of the variables in the network (besides D).

What is the running time of variable elimination? The inner loop requires constructing a factor over some set of variables \mathbf{A} , and then marginalizing this factor with respect to one of the variables. The time required to do this will be roughly linear *in the size of the factor*. For instance, if all of the variables are binary and we produce a factor over four variables, that factor will have 16 entries. Since the inner loop is executed once for each non-query variable, the running time will be $O(mk^D)$, where m is the number of non-query variables, k is the size of the largest domain of any variable, and D is the largest number of variables mentioned in any factor produced by variable elimination. This is exponential in k , but Bayes net inference is NP-hard, and so (assuming $P \neq NP$) the worst-case complexity will be exponential in the size of the network, no matter what algorithm we use.⁴

Given that variable elimination's running time is exponential in the size of the largest factor, how large will the factors be in practice? As a rule of thumb, Bayes nets which are tightly connected (in that there are many active paths through which one variable influences another) will tend to produce large intermediate factors. If the network is sparsely connected, the factors will tend to remain small. Also, the size of the intermediate factors is heavily dependent on the particular variable ordering we choose. Typically, we choose the ordering by hand. Finding the best ordering is NP-hard in general, but several heuristics work well in practice. You will see examples of this in section.

⁴More specifically, the worst-case complexity is exponential in the number of CPT entries needed to specify the joint distribution.