Massachusetts Institute of Technology

6.034 Artificial Intelligence

Assignment # **8** Due: **3 Nov 98** 6.034 Item # **13**

Goals

This assignment is meant to teach you about methods of reasoning with uncertainty. The main part focuses on *Bayes Networks*, a graphical formalism that represents probabilistic relationships among random variables. These methods can be used in many different kinds of reasoning tasks in AI, but here we focus on diagnostic applications.

Background

Much of what we know about the world is known with various degrees of certainty. Although many different numerical models have been proposed for representing this uncertainty, many researchers today believe that probability theory—the most extensively studied uncertainty calculus—is an appropriate basis for building computerized reasoning systems.[1] We assume that you have read (and understood) the paper "Uncertainty and Decisions in Medical Informatics" by Szolovits, handed last Thursday, which reviews much of the required background.

Mathematical Preliminaries

We take the world of interest to consist of a set of random variables, $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$. Each of the X_i can take on one of a discrete set of values, $\{x_{i1}, x_{i2}, \ldots, x_{ik_i}\}$.[2] Each possible combination of assignments of values to each of the variables represents a possible state of this world. There are $\prod_{i=1}^{n} k_i$ such states. This is clearly exponential in the number of variables; for example, if each variable is binary, all of the $k_i = 2$ and the number of states is 2^n. Each state may be identified with the particular values that it assigns to each variable. We might have, for example, $S_{12} = \{X_1 = a, X_2 = e, \ldots, X_n = b\}$.

A probability function, P, assigns a probability to each of these possible states. The probability for each state, $P(S_i)$, is the *joint probability* of that particular assignment of values

[1] The other major alternatives pursued by some include *fuzzy set theory* and *Dempster-Shafer theory*.

[2] Formulations using continuous variables are also possible, but we will not pursue them here. A continuous variable can, of course, be approximated by a large number of discrete values.

to the variables. All states are distinct, and they exhaustively enumerate the possibilities; therefore,

$$\sum_{j=1}^{\prod_{i=1}^{n} k_i} P(S_j) = 1.0$$

We will often be interested in the probability not of individual states, but of certain combinations of particular variables taking on particular values. For instance, we may be interested in the probability that $\{X_3 = a, X_5 = b\}$. In such cases, we wish to treat other variables as "don't cares." We call such a partial description of the world a *circumstance*,[3] C, and the variables that have values assigned the *instantiation-set* of the circumstance, $I(C)$. The probability of a circumstance C can be computed by summing the probabilities of all states that assign the same values to the variables in the instantiation set $I(C)$ as C. If we re-order variables $\mathcal{X}$ so that the first m are the instantiation set of C, then (in a shorthand notation):

$$P(C) = \sum_{X_{m+1}} \dots \sum_{X_n} P(S).$$

For example, if the world consists of four binary variables, W, X, Y and Z, then the circumstance $\{X = \text{true}, Z = \text{false}\}$ is given by

$$P(\{X = \text{true}, Z = \text{false}\}) = \sum_{u \in \{\text{true,false}\}} \sum_{v \in \{\text{true,false}\}} P(\{W = u, X = \text{true}, Y = v, Z = \text{false}\})$$

The computational difficulty of calculating the probability of a circumstance comes precisely from the need to sum over a possibly vast number of states. The number of such states to sum over is exponential in the number of "don't cares" in a circumstance.

Defining P for each state of the 2^n world states is a rather tedious and unintuitive way to describe probabilities in the world, although it is quite flexible. It might happen that the world really *is* so complex that only such an exhaustive enumeration of the probability of each state is adequate, but fortunately many variables appear to be *independent*. For example, in medical diagnosis, the probability that you get a strep infection is essentially independent of the probability that you catch valley fever (a fungal infection of the lungs prevalent in California agricultural areas). Formally, this means that for all possible values s of the variable strep and all possible values v of the variable vf (for valley fever),

$$P(\text{strep} = s, \text{vf} = v) = P(\text{strep} = s)P(\text{vf} = v).$$

When two variables both depend on the same set of variables but not directly on each other, they are *conditionally independent*. For example, if an infection causes both a fever and diarrhea, but there is no other correlation between these symptoms, then for all (true and false) values f of fever, d of diarrhea, and i of infection,

$$\begin{aligned}&P(\text{fever} = f, \text{diarrhea} = d \mid \text{infection} = i) = \\ &\quad P(\text{fever} = f \mid \text{infection} = i)P(\text{diarrhea} = d \mid \text{infection} = i)\end{aligned}$$

[3] "Circumstance" is not a commonly-used term for partial descriptions of the world. Other terms include "partially-specified state" and "evidence".

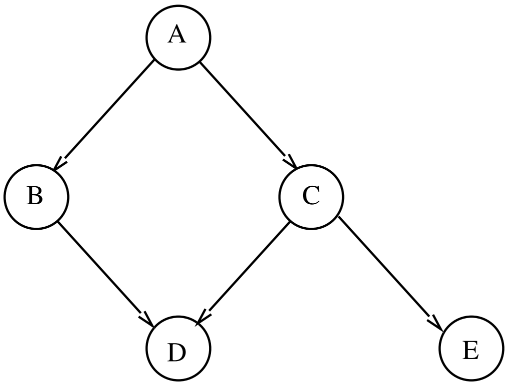

Figure 1: The MCBN1 network shows probabilistic dependencies among the variables A, B, C, D and E. Numerical details are not shown.

The dependencies among variables in a domain support a convenient graphical notation, called a *Bayes network*. In it, each variable is a node, and each probabilistic dependency is drawn as a directed arc to a dependent node from the node it depends on. This notion of probabilistic dependency does not necessarily correspond to what we think of as causal dependency, but it is often convenient to identify causality with dependency. When there is no directed arc from one variable to another, then we say that the second does not depend on the first. For example, in the network in Figure 1, E depends only on C. Therefore the influence of any of the other variables on E can impinge only by changing the probability of C. For each node X, we call those nodes from which directed arcs exist in the network to X the *parents* of X, written $\pi(X)$. Some nodes have no parents ($\pi(X) = \emptyset$); they have only an *a priori* probability and do not depend on others.[4]

In this representation, instead of assessing joint probabilities for each state, we define the probability distribution by writing down for each node X

$$P(X \mid \pi(X))$$

For MCBN1, this requires $P(A)$, $P(B \mid A)$, $P(C \mid A)$, $P(D \mid B, C)$ and $P(E \mid C)$. Each of these is actually a multidimensional array, one dimension corresponding to each variable in the conditional probability, and the final dimension spanning the possible values of that variable. Because all of the MCBN1 variables are binary, this means $P(A)$ holds just two values, corresponding to the probability that $A =$ true and $A =$ false, and $P(D \mid B, C)$ is a $2 \times 2 \times 2$ array. Note that for any particular set of values for parent variables, the probabilities for all possible values of the variable associated with a node must sum to one. Thus, in the case of the $2 \times 2 \times 2$ array $P(D \mid B, C)$, we need to determine only 2×2 values: the probabilities that D is true under the four possible circumstances defined by the possible values of the parents. In each circumstance, the probability that $D =$ false is just $1 - P(\{D = \text{true}\} \mid B, C)$.[5] In general, if a node has p parents, each of which can take on

[4]The *a priori* probability of an event is the probability that it will happen given no additional information.

[5]As a notational convenience, at times we don't bother to specify the values that B and C are to take on. In this case, we mean the equation to apply to all possible sets of such values.

k values, and if the node itself can take on k values, then the total number of parameters to determine is $(k-1)k^p$. One way to (drastically) reduce the number of parameters that need to be estimated is to assume that the parents influence the variable at this node in some simpler, systematic way. For example, if we believe that the influence of the parents on this node may be modeled by a *noisy-or*—i.e., each parent is a (probabilistic) cause of this node and that these causal influences are independent—then we need only p parameters to characterize the node if all variables are binary. (See the section, below, that describes *noisy-or* in greater detail.)

The probability of a state is a product over all variables of the probability that the variable takes on its particular value (in that state) given that its parents take on their particular values:

$$P(X_1 = v_1, \ldots, X_n = v_n) = \prod_{i=1,\ldots,n} P(X_i \mid \pi(X_i))$$

The right hand side is an abbreviation for

$$\prod_{i=1,\ldots,n} P(X_i \mid \pi(X_i)) = \prod_{i=1,\ldots,n} P(X_i = v_i \mid X_j = v_j, \ldots, X_l = v_l)$$

where $\pi(X) = \{X_j, \ldots, X_l\}$. As described above, to find the probability of a circumstance, we must still sum over all of the states that are consistent with the circumstance—i.e., a number of states exponential in the number of "don't cares."

Much of what is complicated about performing inference in Bayes networks is that we cannot afford to do the computations just as they are described above. Instead, we seek to find efficient decompositions of the networks so that computation can be done more efficiently. As usual, even the problem of finding such optimal decompositions is *NP*-hard, so in practice all we can hope for is a reasonably efficient strategy for finding reasonably good decompositions. In the code used in this assignment, we employ an implementation of an algorithm developed by Greg Cooper, who was at Stanford at the time,[6] and implemented by Peter Szolovits, originally in Common Lisp.

Noisy or

When a node in a Bayes net has many parent nodes, the size of the conditional probability table that specifies how likely each value of the node is under all combinations of values of its parents can get to be quite large. For example when a node has 10 parents (10 nodes with links that point to it), if they are all binary, then the number of parameters in the conditional distribution is $2^{10} = 1024$. That is a lot of numbers to specify, too many for most practical problems. In addition, we may often know something systematic about the relation between the node and its parents; in this case, just specifying lots of numbers hides that relationship inappropriately. In the special, but common, case that each parent of a

[6]Cooper, G. F. (1990). Bayesian Belief-network Inference Using Recursive Decomposition (Technical Report No. KSL 90-05). Stanford University.

node is a potential independent *cause* of it, we can use a much simpler model for how the node depends on its parents, called a *noisy-or* model. In this special case, we get to express the specific relationship between parents and child explicitly, and we can reduce the number of parameters drastically, in the case of our 10-parent example down from 1024 to 11. Ten of these are the *causal probabilities* that each parent causes the state represented by the node, and the eleventh is the *leak* probability that the state occurs even though none of its possible causes is present. This is equivalent to adding an additional parent that is always there, which can cause the event with the leak probability. The leak is used as a way to model the "unmodeled" causes of a node.

The intuition behind *noisy-or* is that if each of a number of parents can cause (with some likelihood) some event, then the probability that the event actually occurs is just the probability that at least one of its parents (or the leak) actually caused it. Consider a simple coin-flipping situation, such as one you might have seen in high school: If there are two fair coins being flipped, what is the probability that I get at least one heads. In this case, we assume there is no leak. The easiest way to compute this is to consider the opposite, namely what is the probability that I get no heads.

$$P(\text{no heads}) = P(\text{coin 1 was not heads})P(\text{coin 2 was not heads})$$

or

$$P(\text{no heads}) = (1 - P(\text{coin 1 was heads}))(1 - P(\text{coin 2 was heads}))$$

or

$$1 - P(\text{at least one heads}) = (1 - P(\text{coin 1 was heads}))(1 - P(\text{coin 2 was heads}))$$

Before we proceed, it is useful to introduce a notational shortcut to simplify our equations. Until now, we have been writing formulas such as $P(A = \text{true})$ to represent specific numerical probabilities, and formulas such as $P(A)$ to represent tables of probabilities (in this case, the probability that A is true and the probability that it is false). In the following discussion, we will abbreviate $P(A = \text{true})$ as $P(a)$ and $P(A = \text{false})$ as $P(\bar{a})$. In other words, for binary variables, we let the lower-case letter stand for the variable having the value false, and the lower-case letter with an over-bar for the variable having the value true. With this notation, for example, the particular probability that A is true given that B is false, $P(A = \text{true} \mid B = \text{false})$ would be written more concisely as $P(a \mid \bar{b})$. Expressions such as $P(A \mid B)$ continues to be the table of four elements: $P(a \mid b), P(\bar{a} \mid b), P(a \mid \bar{b}), P(\bar{a} \mid \bar{b})$. The clue that we mean a single probability, rather than a table, is the use of lowercase letters.

Now that notation is taken care of let's look at a trivial noisy-or example (see Figure 2). In this case there is a single parent, A, of node B. This example is so simple, there will be no *advantage* to using the noisy-or, but it will illustrate some important ideas. In a classical Bayes net we would need to know the prior, $P(A)$, and the conditional distribution of B given A, $P(B \mid A)$. We store each as a simple table: $P(A)$ requires one number (since $P(a) + P(\bar{a}) = 1$, and $P(B \mid A)$ requires two (since $P(b \mid a) + P(\bar{b} \mid a) = 1$ and $P(b \mid \bar{a}) + P(\bar{b} \mid \bar{a}) = 1$). The noisy-or formulation will require three parameters also, $P(A)$

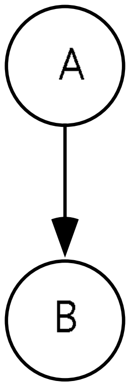

Figure 2: A rather uninteresting example of *noisy-or*—a node B with just a single cause A and no leak.

which is a table as before, and two new parameters: c_a the causal probability of B given that A happened and l the leak probability (the leak probability is the probability that B happens even when A is not there to cause it). When we use noisy-or we choose to write out $P(B \mid A)$ in a different form than a table:

$$
\begin{aligned}
P_{NO}(b \mid a) &= 1 - (1 - c_a)(1 - l) && (1)\\
P_{NO}(\bar{b} \mid a) &= (1 - c_a)(1 - l) && (2)\\
P_{NO}(b \mid \bar{a}) &= 1 - (1 - l) = l && (3)\\
P_{NO}(\bar{b} \mid \bar{a}) &= 1 - l && (4)
\end{aligned}
$$

(Note: we have used the subscript $P_{NO}()$ to emphasize that this is the noisy-or probability.) In order to specify a noisy-or relationship between A and B we must specify c_a and l. If we had the conditional probability table for $P(B \mid A)$, finding c_a and l requires some algebra. Take the following conditional distribution:

$P(A \mid B)$	b	$\bar{b}$
a	0.6	0.4
$\bar{a}$	0.5	0.5

Since $P(b \mid \bar{a})$ should equal $P_{NO}(b \mid \bar{a})$ which equals l, the correct "leak" probability is $l = 0.5$ (this is the easy one). Since $P(b \mid a)$ must equal $P_{NO}(b \mid a)$ which equals $1 - (1 - c_a)(1 - l)$, the correct causal probability is:

$$
\begin{aligned}
1 - (1 - c_a)(1 - l) &= 0.6 && (5)\\
(1 - c_a)(1 - l) &= 0.4 && (6)
\end{aligned}
$$

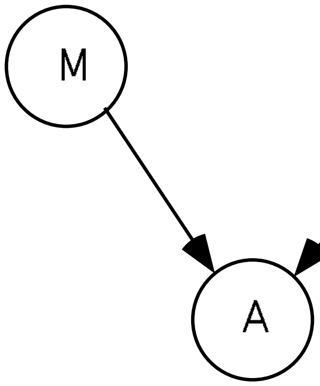

Figure 3: The joint probability that attendance at a faculty meeting by two distinguished professors will result in an argumentative meeting. If we believe that either professor can cause arguments, that there is no special reason why the presence of both will cause extra arguments or fewer, and arguments may arise even in their absence, then *noisy or* is an appropriate model.

$$(1 - c_a) = \frac{0.4}{1-l} = \frac{0.4}{0.5} = 0.8 \quad (7)$$
$$c_a = 0.2 \quad (8)$$

So in the end we can see that the causal probability of A causing B is 0.2 which is nothing like $P(b \mid a)$ which equals 0.6. Can this be thought of more intuitively? Yes, somewhat. We already know that in the absence of A the event B happens about 50% of the time. When A happens, the event B now happens 60% of the time. This is a *weak* effect. The parameter c_a models the increase in probability of B given A. It is a small increase, so c_a is small.

Now we are past the easy stuff. Let's take the following less simple example (see Figure 3). This Bayes net is meant to model faculty meetings at MIT. The node M represents the event that Minsky comes to the meeting (a rare occurence, $P(M = \text{true}) = P(m) = 0.1$). The node C represents the event that Chomsky comes to the meeting (also a rare occurence, $P(C = \text{true}) = P(c) = 0.05$). The node A represents the event that the meeting is argumentative. We have reason to believe that the presence of Minsky makes meetings more argumentative. We also have reason to believe that Chomsky makes meetings more argumentative. Now we have a choice: we could model the relationships between A, C and M with a classical Bayes net, or a noisy-or model. In a classical Bayes net we would be free to represent interactions between Minsky and Chomsky. It could be the case that when they both attend there is never an argument. This would imply there is some interaction between the causes of the argument. The drawback is that we would need more parameters. If we were willing to make the assumption that the causes of the arguments were independent, we could model these interactions with as a noisy-or. By the way, this Bayes net is more complex than the first, but it is still small. The advantages of noisy-or will still be minimal.

For the full conditional model, we would need to specify a table with 4 independent entries: $P(a \mid mc)$, $P(a \mid \bar{m}c)$, $P(a \mid m\bar{c})$ and, $P(a \mid \bar{m}\bar{c})$ (the other 4 depend on these 4). For the noisy-or model there are three parameters: c_m the causal probability that Minsky caused the argument, c_c the causal probability that Chomsky caused the argument, and the l the leak probability. In one case there are 3 parameters, in the other 4. From this very simple fact you should conclude that the full conditional model is more flexible than the noisy-or model.

Just for clarity let's write out the functional form of P_{NO} (remember that $P(\bar{a} \mid mc) = 1 - P(a \mid mc)$): :

$$\begin{aligned}
P_{NO}(a \mid mc) &= 1 - (1 - c_m)(1 - c_c)(1 - l) && (9)\\
P_{NO}(a \mid m\bar{c}) &= 1 - (1 - c_m)(1 - l) && (10)\\
P_{NO}(a \mid \bar{m}c) &= 1 - (1 - c_c)(1 - l) && (11)\\
P_{NO}(a \mid \bar{m}\bar{c}) &= 1 - (1 - l) = l && (12)
\end{aligned}$$

In order to find c_m, c_c, and l we may need to do some real detective work. So for example we may be told that when neither Minsky nor Chomsky is there the probability of an argument is 0.3, or in other words $P(a \mid \bar{m}\bar{c})$. From this we conclude that $l = 0.3$. We may also be told directly that the causal probability of Minsky causing an argumentative meeting is 0.5. This is direct information so we can conclude that $c_m = 0.5$. In order to understand a bit more about this information let's, for a moment, think of the *leak* as another cause L. From the equations above we can see that L can be thought of as another cause, one that we assume is always present. Sort of like that old guy in the corner that comes to all the meetings. Even when Minsky and Chomsky are not there, this old guy might cause an argument (with probability l). When someone tells you that the causal probability of Minsky causing the argument is 0.5, they are telling you the probability that Minsky causes the argument given that no other person caused the argument, even the leak.

We are not yet done specifying the parameters, c_c is still missing. Someone might tell you that $P(a \mid c) = 0.7$. Is this the same thing as telling you c_c? No. This information does not say that we know that Chomsky caused the argument. It could have been Minsky. It could have been the leak. What it does say is that arguments occur 70% of the time when Chomsky is around. How do we use this information to find c_c? Algebra, my dear Watson, algebra.

First of all we must compute $P_{NO}(a \mid c)$. We have not said what this is yet, but it is a function of c_m, c_c, and l. We can get it by summing over the possible values of M (to make the algebra simpler we will find $P_{NO}(\bar{a} \mid c)$):

$$\begin{aligned}
P_{NO}(\bar{a} \mid c) &= P_{NO}(\bar{a} \mid mc)P(m) + P_{NO}(\bar{a} \mid \bar{m}c)P(\bar{m}) && (13)\\
P_{NO}(\bar{a} \mid c) &= (1 - c_m)(1 - c_c)(1 - l)P(m) + (1 - c_c)(1 - l)P(\bar{m}) && (14)\\
(1 - 0.7) &= (1 - c_m)(1 - c_c)(1 - l)0.1 + (1 - c_c)(1 - l)0.9 && (15)\\
0.3 &= (1 - 0.5)(1 - c_c)(1 - 0.3)0.1 + (1 - c_c)(1 - 0.3)0.9 && (16)\\
0.3 &= 0.5 \times (1 - c_c) \times 0.7 \times 0.1 + (1 - c_c) \times 0.7 \times 0.9 && (17)
\end{aligned}$$

$$
\begin{aligned}
0.3 &= 0.665(1 - c_c) \qquad (18)\\
c_c &\approx 0.55 \qquad (19)
\end{aligned}
$$

In a larger Bayes net, it may be quite sensible to model some dependencies by noisy or and others by giving full conditional probability tables. Because the noisy or model can always be re-expressed in terms of the full conditional model, general algorithms for computing with Bayes networks can work. More clever implementations will take advantage of the noisy or models without translating them.

Code Preliminaries

The programs for this assignment are contained in the following files:

1. `load-ps7.scm` simply loads up the other files.
2. `cooper.scm` implements most of the programs, including convenient ways to define Bayes networks, to decompose them for efficient computation, and to perform the computations.
3. `utils.scm` contains some utilities programs.
4. `structs.scm` defines the data structures for Bayes networks, nodes in such a network, and the record structures that hold the decomposition. Note that these data structures are circular (e.g., nodes include lists of their parents and children, which must of course similarly include the original node, etc.). Therefore, it is not possible to print these structures with the Scheme printer.
5. `debug.scm` holds a few functions that allow one to print out parts of the circular structures.
6. `examples.scm` defines two Bayes networks, `mcbn1` and `mcbn2`, which have the structure shown in Figure 1 but with different numerical models.

We do not include a listing of the above programs with the problem set. *You are not expected to understand the details of the implementation in* `cooper.scm`. The code is there to help you do experiments which should reinforce your understanding of the material. It is quite complex...

`Define-bnet` is a function that makes it convenient to define a net. Here is an example of its use, to define `MCBN1`.[7]

[7] In code comments, we write a tilde followed by `a` where we have been writing $\bar{a}$, above. We also try to carry through the convention that capital letters stand for the variables, while lower-case letters stand for the true values of the variables. Scheme, of course, makes no distinctions based on capitalization.

```
(define mcbn1
  (define-bnet 'mcbn1
    '((A () (0.8 0.2))                    ;; P(a) = 0.2
      (B (A) ((0.9 0.1) (0.3 0.7)))       ;; P(b|~a) = 0.1, P(b|a) = 0.7
      (C (A) ((0.7 0.3) (0.4 0.6)))       ;; ...
      (D (B C) (((0.9 0.1) (0.7 0.3)) ((0.6 0.4) (0.2 0.8))))
      ;; P(d|~b,~c) = 0.1, P(d|~b,c) = 0.3, P(d|b,~c) = 0.4, P(d|b,c) = 0.8
      (E (C) ((0.9 0.1) (0.6 0.4))))))
```

See the comments in `cooper.scm` for the exact syntax and options. Defining the network also calls the procedure `dissect`, which computes the decomposition that makes computation more efficient.

Our code provides just a single program that can pose a query to the Bayes net: `infer`. It takes as its arguments a Bayes net and a list that specifies which variables should be set to which values. For example,

```
(infer mcbn1 '((D #t)))
```

asks for the probability of the circumstance in which the variable D has the value *true*. The probability of a state can be computed by making sure that every variable appears in this list.

```
(infer mcbn1 '((A #t) (B #f) (C #t) (D #t) (E #t)))  ==> 0.00432
```

Finally, the situation in which no variables are set is consistent with all states, and must have probability 1.0.

```
(infer mcbn1 '())  ==> 1.0
```

Problem 1 Warmup

You are asked to analyze a new type of molecular test for left-handedness based on the following information: 10% of the people are left-handed; the test correctly determines your handed-ness 85% of the time. What is the probability that a person is left-handed given that the test is positive?

Problem 2 Conditional Probability

`Infer` computes the probability of a circumstance. Often, we will be interested instead in a conditional probability. Recalling the definition of the conditional probability, write a function, `conditional-probability`, that takes a Bayes net and two lists of variable/value specifications (like the second argument to `infer`), and computes the conditional probability that the first circumstance holds given that the second holds. A correct answer to this problem is a very short program that does not require you to understand the internals of `cooper.scm`. For example, you should be able to compute

```
(conditional-probability mcbn1 '((E #f)) '((A #t)))
```

Calculate this value by hand and check it against your program's output.[8] Turn in your program and show its results on the following problems, all using `mcbn1`:

$$P(E = \texttt{\#f} \mid B = \texttt{\#t})$$

$$P(D = \texttt{\#t}, E = \texttt{\#t} \mid A = \texttt{\#t}, B = \texttt{\#f})$$

$$P(A = \texttt{\#t} \mid D = \texttt{\#t}, E = \texttt{\#t})$$

Problem 3 Independence

Sometimes you may wish to know if two variables in a Bayes Net are dependent or independent. Recall that variables A and B are independent if $P(A, B) = P(A)P(B)$ for every value of A and B. For example based on the information in a particular Bayes net you may conclude that the color of a car is not dependent on its price.

Part A

Write a program called `independent?` which takes a Bayes net, two variable names, and returns `#t` if the variables are independent. A correct answer to this question requires no more than 15 lines of code. Turn in your program and its results on the following problems:

```
(independent? mcbn1 'A 'B)
(independent? mcbn1 'D 'E)
(independent? mcbn1 'A 'D)
```

[8] If you ever get a probability from a program that is not in the range between 0 and 1 (inclusive), you have a good hint that you've made a mistake.

We expect your program to use numerical relationships to determine independence, though structural relationships should also allow computation of an answer. The structural approach, though, is more complex than we wish to explore deeply in this homework. The next part examines the simplest case.

Part B

One reason that Bayes nets are popular is that they provide a graphical network representation of a problem. Consider a Bayes net that contains, among others, the nodes X and Y, but in which no observations of the values of any variables have been made. In such a network, what simple property of the graph relating nodes X and Y would allow you to conclude that X and Y are independent?

Problem 4 Conditional Independence

Two variables may be dependent in most cases, but independent in the context of particular circumstances. Imagine that two professors Peter and Tomas are driving into MIT to meet with Patrick on a particular day. Peter and Tomas come from different directions, and drive different cars. Both Peter and Tomas are prompt and responsible, but not too responsible. Knowing nothing else, we might assume that the probability that Peter is late will be 0.1 and the probability that Tomas is late will be 0.05. Both Peter and Tomas are conscientious drivers. When it is raining, they drive slowly and each is more likely to be late (i.e., for each professor, the probability that he is late when it is raining is 0.5).

Part A

Draw a Bayes net that represents this situation (the net should have no more than 3 nodes and 3 links). Create a Bayes net using `define-bnet` called `prof`. Is there information missing from the problem description? In order to complete your net feel free to make an educated guess about any missing information.

Part B

Turn in the answers to the following questions:

1. If I tell you that Peter is late, would you be more likely to conclude that Tomas is late?
2. Does the probability that Peter is late depend on whether Tomas is late?

3. Would your answers be different if you knew that it was raining? How?

Discuss the reasoning behind each conclusion briefly. Show how your could answer these questions using the Bayes net programs described above (and perhaps `c-independent?` described below).

Part C

Write a program called `c-independent?` which takes a Bayes net, two variable names, and an instantiation of a condition, and which returns `#t` if the variables are conditionally independent. Turn in your program and its results on the following problems:

```
(c-independent? mcbn1 'A 'B '((C #t)))
(c-independent? mcbn1 'D 'E '((C #t)))
(c-independent? mcbn1 'A 'D '((B #f) (C #t)))
(c-independent? mcbn1 'A 'D '((B #t) (C #f)))
```

The only difference between the last two examples is the state of variable B and C. In general, is conditional independence a function of the value of the variables in the circumstance? How about in the specific case of Bayes nets?

Problem 5 Modeling complex probabilistic dependence

Here is a summary of an extended example used by Judea Pearl to introduce Bayes networks in his classic book, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, 1986:

> Holmes is worried that his house will be burgled. For the time period of interest, there is a 10^{-4} *a priori* chance of this happening, and Holmes has installed a burglar alarm to try to forestall this event. The alarm is 99% reliable in sounding when a burglary happens, but also has a false positive rate of 0.2%. Holmes' neighbor, Watson, is sure (well, say 85% sure) to call Holmes at his office if the alarm sounds, but he is also a bit of a practical joker, and knowing Holmes' concern, might (25%) call even if the alarm is silent. Holmes' other neighbor, Mrs. Gibbons, is a well-known lush and often befuddled, but Holmes believes that she is five times more likely to call him if there is an alarm than if not.

Part A Building the model

Draw a Bayes network that describes the above dependencies. Now define a Bayes net, `Holmes`, that models this. Think carefully about the probabilistic dependencies and exactly what they mean. Turn in a listing of the call to `define-bnet` that defines your network.

You will have noticed that the story above does not tell the actual conditional probability that Gibbons will call if the alarm sounds or that she will call if the alarm is silent. Indeed, these probabilities would be quite difficult for Holmes to estimate. Fortunately, it is only the likelihood ratio between these conditions that matters. You can verify that your choice or particular numbers to use for the conditional probabilities leading to Gibbons' call are immaterial so long as their ratio is 5.

Part B Calculations

Within a few minutes of each other, Holmes is phoned by both Watson and Gibbons. Use your network to calculate the probability that Holmes has suffered a burglary.

Part C Explaining away

Realizing that Holmes lives in Los Angeles, it occurs to us that there might be another explanation for the (probable) alarm at Holmes' house. If a sufficiently large earthquake occurs, it may set off the burglar alarm even without there being a burglary. Define a new Bayes net, `quake`, based on the same facts that you used for `holmes`, but with the following additional story:

> Holmes, always the anxious type, listens to KNX, "News Radio," while at work, which will report the occurrence of a sufficiently large quake 97% of the time. Assume that KNX never makes a false positive report of a quake. Holmes thinks that a burglary and an earthquake can each set off his burglar alarm in independent ways that make *noisy-or* a good model. From his original belief about the probabilistic connection between burglary and alarm, Holmes can compute a causal probability from burglary to alarm and a leak probability for an alarm occurring in the absence of a burglary. He estimates the probability of a quake causing an alarm as 70%. The incidence of quakes in Los Angeles is equal to the incidence of burglaries.

Use the `quake` network to calculate the probability of Holmes' house having been burgled if he receives calls from Watson and Gibbons, but also hears the report of a quake on KNX.