Bayesian Networks

Logical Reasoning An overarching goal of AI is to build artificial agents that can reason about the world around them. You have already built reasoning systems for classical search problems, such as puzzles and games. In these settings, the problem domain is "hard-coded." The state space is defined explicitly, and is often finite; the transition model from one state to another is also made explicit, as the legal moves; the goal states are listed explicitly, as determined by the rules of the game. We thus studied search algorithms for problem with narrow (i.e., explicit) specifications. The goal of this next unit—knowledge representation and reasoning—is to expand our suite of search algorithms to problems with more implicit, and hence much broader, specifications. Systems that solve such problems using logical inference are called **expert systems**.

Expert systems are also sometimes called **rule-based systems**, since the agents' knowledge base is encoded as rules. The languages in which these rules are encoded are **symbolic**, or logical, languages, so that logical inference "meta-rules" (e.g., modus ponens or modus tollens) can applied to draw logical conclusions. For example, an agent's knowledge base might encode the rule, "CS 410 lectures are on Mondays, Wednesdays, and Fridays." If we then assert that "Today is Friday," the agent can apply modus ponens to logically infer that "a CS 410 lecture is happening today."

A simple way to encode this knowledge base would be to use propositional logic, as follows:

Proposition	Meaning
`TodayMonday` $\Rightarrow$ `ClassToday`	If today is Monday, then there is class today.
`TodayWednesday` $\Rightarrow$ `ClassToday`	If today is Wednesday, then there is class today.
`TodayFriday` $\Rightarrow$ `ClassToday`	If today is Friday, then there is class today.

Now, if we assert `TodayFriday`, meaning "Today is Friday," we can apply modus ponens to this fact and the third rule to conclude that there is class today.

We can express these same rules more generally and more compactly using predicate calculus as follows:

Predicate	Meaning
`class(Monday)`	There is class on Mondays.
`class(Wednesday)`	There is class on Wednesdays.
`class(Friday)`	There is class on Fridays.
`today(day)` $\wedge$ `class(day)` $\Rightarrow$ `class_today`	
`tomorrow(day)` $\wedge$ `class(day)` $\Rightarrow$ `class_tomorrow`	

Again, if we assert `today(Friday)`, we can bind the variable `day` to `Friday` and then, since `today(Friday)` and `class(Friday)` are both true, we can conclude—again, via modus ponenss—`class_today`.

Additionally, we can easy extend this system with the rules `today(Sunday)` $\Rightarrow$ `tomorrow(Monday)`, `today(Monday)` $\Rightarrow$ `tomorrow(Tuesday)`, `today(Tuesday)` $\Rightarrow$ `tomorrow(Wednesday)`, etc.
Then, if we assert `today(Sunday)`, we can conclude `class_tomorrow`.

Logical inference—deriving new sentences from old—can be very powerful in a *certain* world, e.g., one in which CS 410 lectures are never cancelled. But let's say that today is Sunday, and that there is a 90% chance of 1 foot of snow tomorrow,[1] and that Brown cancels classes with probability 50% whenever there is 1 foot of snow. Inference in an uncertain world is our next knowledge representation and reasoning topic.

[1] These lecture notes were for a Spring semester rendition of the course, on a day when there was indeed a chance of snow.

MYCIN, an expert system developed by Stanford researchers in the 1970s for medical diagnoses, was a rule-based system enhanced with "certainty factors" to represent uncertainty. While its method of logical reasoning was principled, its method of reasoning under uncertainty was ad-hoc. **Bayesian networks** were introduced in the 1980s as a principled alternative to expert systems for reasoning under uncertainty.

Probabilistic Models Modern AI systems rely on **probabilistic models** to reason about uncertainty. At their core, these models define a probability distribution over one or more random variables. Moreover, they incorporate structure that defines relationships among these variables, which may be causal, temporal, or relational.[2] Like other modern AI systems, probabilistic models are data-driven, meaning their parameters are learned from data. They can then be queried given observations relevant to the model (akin to asserting "Today is Friday") to make probabilistic inferences.

The basic reasoning unit in a probabilistic model is the **random variable**, which is a variable that may take on one of many values, each with some probability. For example, a die may take on one of six values (1 through 6), each with equal probability. Alternatively, a p-biased coin may take on one of two values (heads or tails), heads with probability p, and tails with probability $1 - p$. Given a set of random variables, a probabilistic model expresses relationships among those variables as a joint distribution over them.

For example, suppose we want to build an AI system to reason about symptoms and a disease. Suppose the symptoms are `fever`, `cough`, and `fatigue`, and suppose the diseases are the `common_cold` and the `flu`.

An expert system might include rules like:

$$\texttt{symptom(fever)} \land \texttt{symptom(cough)} \land \texttt{symptom(fatigue)} \Rightarrow \texttt{disease(flu)}$$
$$\texttt{symptom(cough)} \land \neg\texttt{symptom(fever)} \Rightarrow \texttt{disease(common\_cold)}$$
$$\texttt{symptom(fatigue)} \land \neg\texttt{symptom(fever)} \Rightarrow \texttt{disease(common\_cold)}$$

With these rules, the system would would draw the definitive conclusion `disease(flu)` from the symptoms `symptom(cough)` and `symptom(fatigue)`.[3]

By way of contrast, let's build a probabilistic model to represent these same symptoms and diseases. This model will have four binary random variables, `flu` (where $\neg$`flu` implies `common_cold`), and each of the three symptoms, `cough`, `fatigue`, and `fever`. The model then represents a joint distribution over all combinations of these variables, including the query of interest: $\Pr(\texttt{flu} \mid \texttt{cough} \land \texttt{fatigue})$. See Figure 2.

Reasoning about the relationships among random variables in a large probabilistic model can quickly become unwieldy. In many cases, however, it is not necessary to fully represent a joint distribution, because of independence relations among the random variables. For example, in the case of n coin flips, we can explicitly enumerate all 2^n outcomes, and compute probabilities for each, which we can then look up in a table to find the probability of a particular outcome (e.g., HTHHTT, assuming $n = 6$).[4] If the coin flips are independent, however, we can more compactly represent this same joint distribution using only n numbers, each corresponding to the probability p_i of heads of the ith coin, which we then multiply accordingly to query a particular outcome: e.g., $p_1(1 - p_2)p_3p_4(1 - p_5)(1 - p_6)$ for outcome HTHHTT.[5]

While independence may be a reasonable assumption for coin flips, it is far too strong of an assumption in many real-world applications. For example, when trying to diagnose a patient with flu-like symptoms, it may not be reasonable to assume that a headache is independent of a tired and achy body, or that a cough is independent of a sore throat.

[2] Logical languages are naturally relational, and temporal logics are, as their name suggests, temporal. But logic-based systems are not as amenable to representing causal relationships as probabilistic models are. On the other hand, most probabilistic models cannot represent infinite domains like systems based on first-order logic can; most generalize only propositional logic.

[3] after applying the closed-world assumption to conclude $\neg$`symptom(fever)`

[4] Exponential space; constant-time lookup.

[5] Linear space; linear-time lookup.

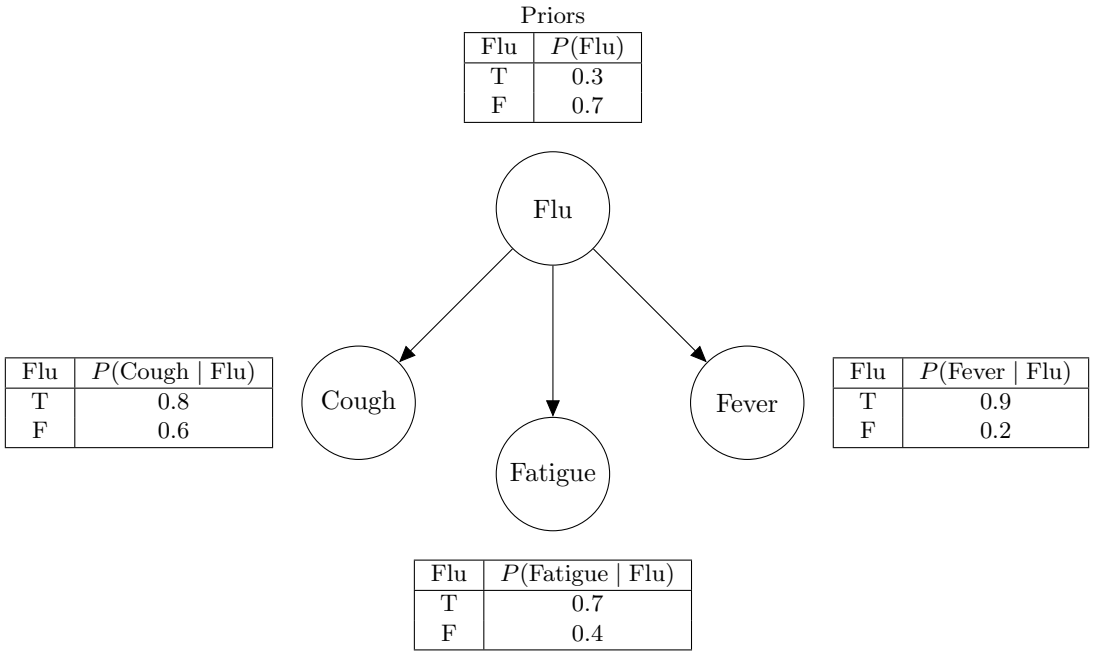


Figure 1: Simple Bayesian network describing symptoms of the flu. The structure of this Bayes net reflects the fact that each symptom is conditionally independent of the others, given that the patient has the flu.

On the other hand, a **conditional independence** relationship may hold among some of these symptoms. That is, knowing that the patient has the flu may render some symptoms, such as a cough and a sore throat independent of each other. Similarly, knowing that the patient has a fever may render headache and bodyache independent of each other. Conditional independence assumptions are more likely to hold than independence assumptions, and also lead to compact probabilistic models.

Bayesian networks (*a.k.a.* Bayes nets) are probabilistic *graphical* models that encode causal structure among their variables. That is, *edges in the graph point from cause to effect.* On the other hand, reasoning is generally "from effect to cause." That is, queries take the form, "I observed certain symptoms. What disease do I have?" In other words, this query asks, "What is the cause of my symptoms?"

With this causal structure, Bayes nets compactly represent joint probability distributions over random variables by encoding not only independence relations, but conditional independence relations as well. An example appears in Figure 2. Whereas explicitly representing this joint distribution would require $2^3 \cdot 3^2 = 72$ numbers, this Bayes net, which encodes conditional independence assumptions, requires only 22 numbers.[6]

The graphical structure of a Bayes net is typically devised by domain experts, as it encodes a great deal of domain knowledge. For example, it might model causal relationships among the variables, such as "the flu may cause a fever", and "knowing that I have the flu, whether or not I have a cough is independent of the whether or not I have a fever." Conditional probability tables (CPTs) then specify the relevant probabilities, given the assumed graphical structure. Global relationships (i.e., relationships among non-neighboring nodes in the graph) can be inferred from the specified local interactions (CPTs) using the laws of probability, together with algorithmic enhancements that improve computational efficiency.

[6]In fact, this Bayes net has only 13 parameters, since $P(\text{Flu} = F)$ can be inferred from $P(\text{Flu} = T)$, for example. Likewise, explicitly representing the joint distribution requires only $2^3 \cdot 3^2 - 1$ parameters.

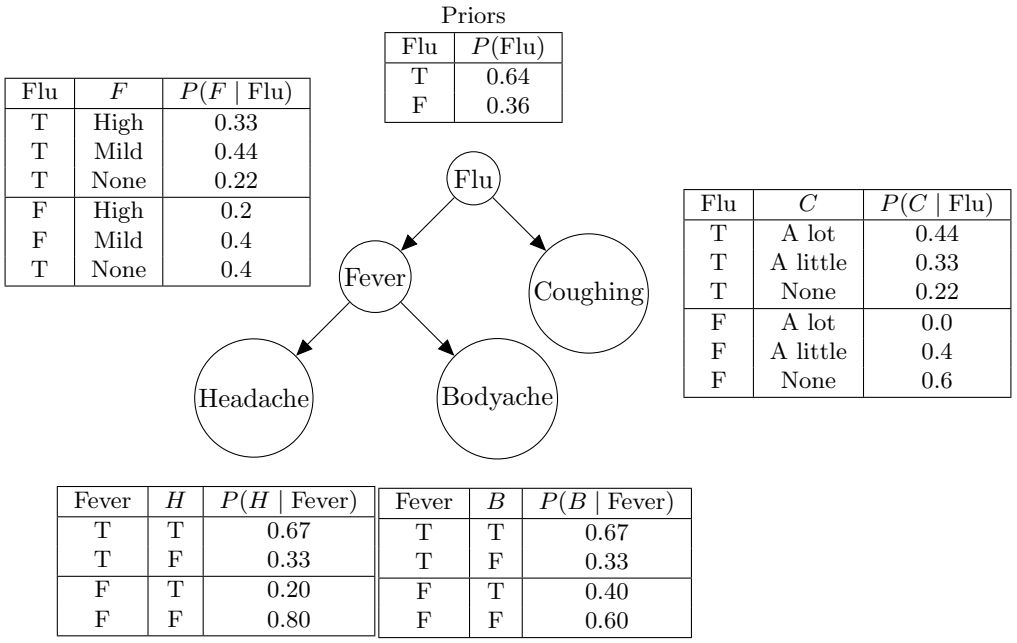


Figure 2: An example of a Bayesian network with conditional probability tables (CPTs).

Bayes nets are a general tool for modeling *noisy, causal processes*. As such, they are used to model many real-world phenomena beyond medical diagnoses, such as risk management, to assess the impact of economic indices on markets; and conservation efforts, to assess the impact of environmental factors on ecosystems. Dynamic (or iterated) Bayesian networks are used to model—and make predictions based on—time-series, or sequential, data. Hidden Markov models (HMMs), which power many speech recognition systems, are one such example: the goal is to identify phonemes based on a series of signals (sounds). Anomaly (e.g., credit card fraud) detection is another common use case.

1 Conditional Independence

Independence is a very strong assumption. When it is raining, I may carry an umbrella and at the same time, there may be traffic on the road. These two events—carrying my umbrella and traffic on the road—are not independent, as they have a common cause.

On the other hand, the probability that I carry my umbrella *given that it is raining* does not depend on whether or not there is traffic on the road. Similarly, the probability that there is traffic on the road *given that it is raining* does not depend on whether or not I carry my umbrella. This intuition is captured by a weaker condition than independence called **conditional independence**. Indeed "carrying my umbrella" and "traffic on the road" are independent, *given that it is raining.*

Like independence, conditional independence can be understood in various ways. See Table 1.

When X is conditionally independent of Y given Z, we write $X \perp\!\!\!\perp Y \mid Z$ (or $Y \perp\!\!\!\perp X \mid Z$). In particular, Traffic $\perp\!\!\!\perp$ Umbrella | Rain. Likewise, Umbrella $\perp\!\!\!\perp$ Traffic | Rain.

Independence	Conditional Independence
$P(X \mid Y) = P(X)$	$P(X \mid Y, Z) = P(X \mid Z)$
$P(X, Y) = P(X)P(Y)$	$P(X, Y \mid Z) = P(X \mid Z)P(Y \mid Z)$

Table 1: Independence vs. Conditional Independence

2 Simplifying the Chain Rule

The chain rule can be used to compute a joint probability as a series of conditional probabilities.

$$\begin{aligned} P(X_1, \ldots, X_n) &= P(X_1)P(X_2 \mid X_1)P(X_3 \mid X_2, X_1) \ldots P(X_n \mid X_{n-1}, \ldots, X_1) \\ &= \prod_{i=1}^{n} P(X_i \mid X_{i-1} \ldots X_1) \end{aligned}$$

A Bayes net encodes conditional independence assumptions that simplify this computation. For example, consider the linear Bayes net, as shown in Figure 3. Under these conditional independence assumptions,

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

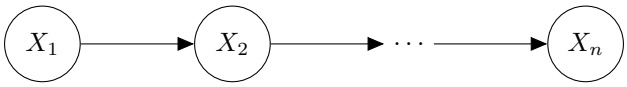


Figure 3: A linear Bayesian network. $X_3 \perp\!\!\!\perp X_1 \mid X_2$. $X_4 \perp\!\!\!\perp X_1, X_2 \mid X_3$. $X_n \perp\!\!\!\perp X_1, \ldots, X_{n-2} \mid X_{n-1}$. For example, assume $n = 3$, and let X_1 be the random variable "it is raining," X_2, the random variable "there is traffic," and X_3, the random variable "the professor is late for class." $X_3 \perp\!\!\!\perp X_1 \mid X_2$, since whether the professor is late for class depends only on whether there is traffic, and not on the root cause of the traffic.

Without any conditional independence assumptions, the CPT $P(X_i \mid X_{i-1} \ldots X_1)$ is exponential in $i - 1$. Assuming Bernoulli random variables, this CPT requires $2 \cdot 2^{i-1} - 1$ probabilities. A linear Bayes net, in contrast, with n variables, requires only $O(2n)$ probabilities, 2 per variable—one to describe $P(X_i \mid 1)$ and a second to describe $P(X_i \mid 0)$, for all $1 \leq i \leq n$.

In general, the edges in a Bayes net indicate dependencies among variables, so that

$$P(X_1, \ldots, X_n) = \prod_{i=1}^{n} P(X_i \mid \text{parents}(X_i))$$

A more complex Bayes net is shown in Figure 6. Imagine your burglar alarm is triggered while you are at the office. When this happens, sometimes your neighbor John calls you, and sometimes your neighbor Mary calls you. If you receive calls from one or both of them, how likely is it that an intruder has entered your home? Or did a small earthquake cause the alarm to sound?

By the chain rule,

$$\begin{aligned} P(B, E, A, J, M) &= P(M \mid J, A, E, B)P(J, A, E, B) \\ &= P(M \mid J, A, E, B)P(J \mid A, E, B)P(A, E, B) \\ &= P(M \mid J, A, E, B)P(J \mid A, E, B)P(A \mid E, B)P(E, B) \\ &= P(M \mid J, A, E, B)P(J \mid A, E, B)P(A \mid E, B)P(E \mid B)P(B) \end{aligned}$$

As per the edges in this graph, $J \perp\!\!\!\perp E, B \mid A$ and $M \perp\!\!\!\perp J, E, B \mid A$. Therefore, $P(M \mid J, A, E, B) = P(M \mid A)$ and $P(J \mid A, E, B) = P(J \mid A)$, so that

$$P(B, E, A, J, M) = P(M \mid A)P(J \mid A)P(A \mid E, B)P(E)P(B)$$

Notice that the joint probability is the product of the probability of each node conditioned on its parents.

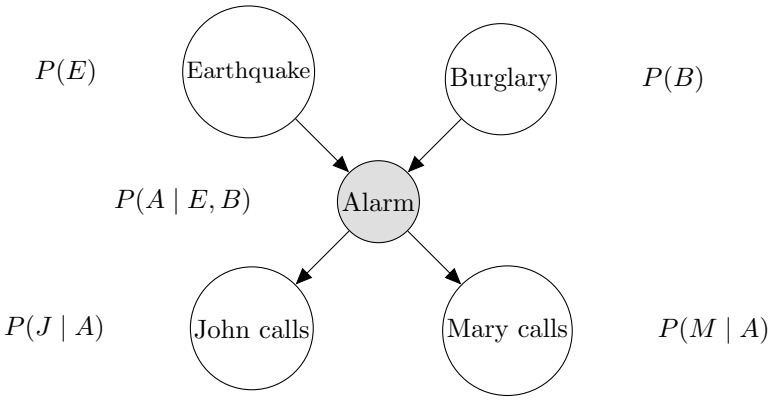


Figure 4: Example from R&N, originally by Judea Pearl.

3 Inference

It is typical to divide the random variables in a Bayes net into three categories: the **evidence** variables E, which are observed, the **query** variables Q of interest, and everything else, which are often referred to as the **latent** variables L (meaning the hidden, or unobserved, variables). We can then compute $P(Q \mid E)$ as per the definition of conditional probability, namely $P(Q, E)$ divided by $P(E)$, where $P(Q, E)$ is obtained by marginalizing over the latent variables: i.e., $P(Q, E) = \sum_L P(Q, E, L)$. For example, we can compute the probability a patient has the flu, given the symptoms observed.

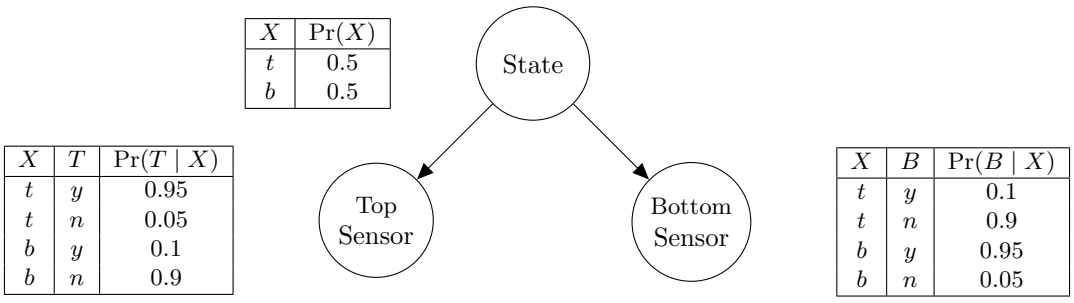
We can similarly compute $\arg\max_{q \in Q} P(Q \mid E)$ to find the most likely explanation for the evidence observed. For example, we can compute the most likely cause of a patient's symptoms, be it the flu, COVID, or just a common cold.

Next, we present an example of a Bayes net that represents a robot's state estimation problem.

Imagine a robot operating in a (very small) state space comprising only two states, top and bottom. Imagine further two sensors, one per state, which report whether the corresponding state is occupied or not. These sensors report information about the current state, but not entirely reliably. In particular, when the robot is in the top state, the top sensor reports this information accurately with probability 0.95. In addition, when the robot is in the *bottom* state, the *top* sensor reports this information accurately with probability only 0.9.

A Bayesian network for this scenario is depicted in Figure 5. We use X to denote the robot's state, T to denote the top sensor's output, and B to denote the bottom sensor's output. This network structure encodes the fact that the top and bottom sensors are independent of each other, given the robot's state. That is, if the robot's state is known, no further information can be gleaned from an additional sensor. We can express these conditional independence assumptions as $B \perp\!\!\!\perp T \mid X$ and $T \perp\!\!\!\perp B \mid X$. They enable us to simplify the computation of the joint probability distribution this Bayes nets encodes, as follows:

$$P(X, T, B) = P(X)P(T \mid X)P(B \mid T, X) \tag{1}$$
$$= P(X)P(T \mid X)P(B \mid X) \tag{2}$$



X	$\Pr(X)$
t	0.5
b	0.5

X	T	$\Pr(T \mid X)$
t	y	0.95
t	n	0.05
b	y	0.1
b	n	0.9

X	B	$\Pr(B \mid X)$
t	y	0.1
t	n	0.9
b	y	0.95
b	n	0.05

Figure 5: A robot's state estimation problem.

As an example, we will compute the probability that the robot is in the top state, given that both sensors report that it is in the top state, namely $P(X = t \mid T = y, B = n)$. In other words, X is our query, and the top and bottom sensors provide our evidence. (There are no latent variables in this small example.)

By the definition of conditional probability,

$$P(X = t \mid T = y, B = n) = \frac{P(X = t, T = y, B = n)}{P(T = y, B = n)}$$

Now, as per Equation 2,

$$\begin{aligned} P(X = t, T = y, B = n) &= P(X = t)P(T = y \mid X = t)P(B = n \mid X = t) \\ &= (0.5)(0.95)(0.9) \\ &= 0.4275 \end{aligned}$$

and

$$\begin{aligned} P(X = b, T = y, B = n) &= P(X = b)P(T = y \mid X = b)P(B = n \mid X = b) \\ &= (0.5)(0.1)(0.05) \\ &= 0.0025 \end{aligned}$$

Therefore,

$$\begin{aligned} P(T = y, B = n) &= \sum_{x \in \{t,b\}} P(X = x, T = y, B = n) \\ &= P(X = t, T = y, B = n) + P(X = b, T = y, B = n) \\ &= 0.4275 + 0.0025 \end{aligned}$$

Finally,

$$\begin{aligned} P(X = t \mid T = y, B = n) &= \frac{P(X = t, T = y, B = n)}{P(T = y, B = n)} \\ &= \frac{0.4275}{0.4275 + 0.0025} \\ &\sim .994 \end{aligned}$$

More Practice with Exact Inference

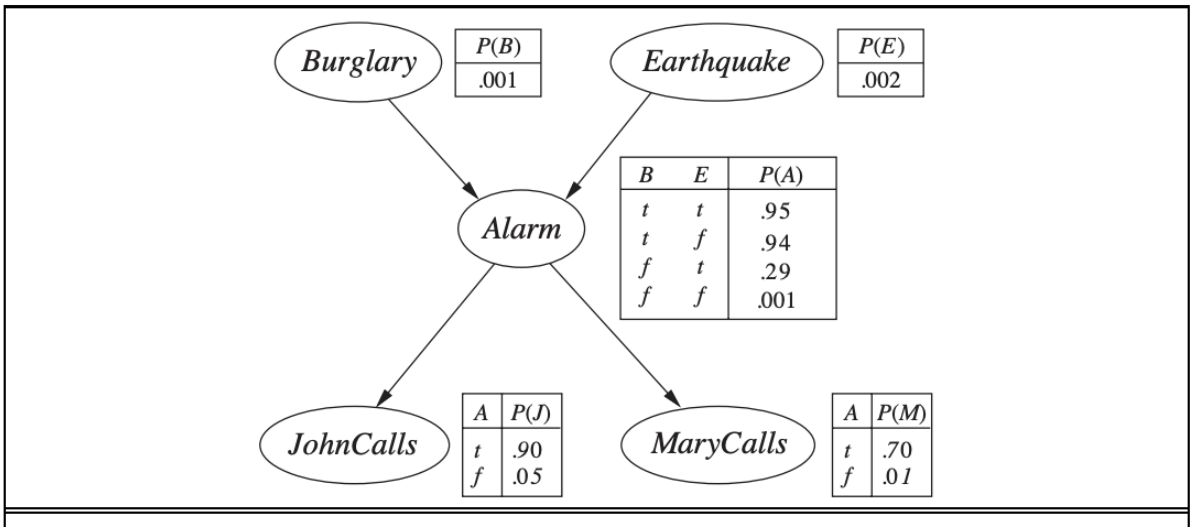


Figure 6: Example from R&N, originally by Judea Pearl.

A more complex Bayes net is shown in Figure 6. Your burglar alarm is ringing while you are at the office. When this happens, sometimes your neighbor John calls you, and sometimes your neighbor Mary calls you. If you receive calls from one or both of them, how likely is it that an intruder has entered your house? Or was it just a small earthquake that caused the alarm to sound?

As per the edges in this graph, $E \perp\!\!\!\perp B$, $J \perp\!\!\!\perp E, B \mid A$, and $M \perp\!\!\!\perp J, E, B \mid A$. Therefore, by the chain rule,

$$\begin{aligned} P(B, E, A, J, M) &= P(M \mid J, A, E, B)P(J, A, E, B) \\ &= P(M \mid J, A, E, B)P(J \mid A, E, B)P(A, E, B) \\ &= P(M \mid J, A, E, B)P(J \mid A, E, B)P(A \mid E, B)P(E, B) \\ &= P(M \mid J, A, E, B)P(J \mid A, E, B)P(A \mid E, B)P(E \mid B)P(B) \end{aligned}$$

By conditional independence (i.e., $P(M \mid J, A, E, B) = P(M \mid A)$), we can simplify even further:

$$P(B, E, A, J, M) = P(M \mid A)P(J \mid A)P(A \mid E, B)P(E)P(B)$$

Notice that the joint probability is the product of each node conditioned on its parents.

Computing Joint Probabilities

To compute the joint probability, for example of $P(B = T, E = F, A = T, J = T, M = F)$, we can simply use the expression derived above:

$$P(B = T, E = F, A = T, J = T, M = F) = P(M = F \mid A = T)P(J = T \mid A = T) \\ P(A = T \mid E = F, B = T)P(E = F)P(B = T)$$

Each term corresponds to a row in a conditional probability table.

Using the values from the tables:

$$P(B = T, E = F, A = T, J = T, M = F) = P(M = F \mid A = T) \cdot P(J = T \mid A = T) \cdot \\ P(A = T \mid E = F, B = T) \cdot P(E = F) \cdot P(B = T) \\ = 0.3 \cdot 0.9 \cdot 0.94 \cdot 0.998 \cdot 0.001 \\ = 2.53 \times 10^{-4}$$

Marginalization Examples

But what if we don't want the entire joint distribution? What if we'd like to know $P(A = T, J = T, M = F, E = F)$? We have an unknown variable—we don't know whether a burglary has taken place or not. We can marginalize over Burglary to find the answer.

$$P(A = T, J = T, M = F, E = F) = P(A = T, J = T, M = F, E = F, B = T) + P(A = T, J = T, M = F, E = F, B = F)$$

Answer:

$$P(A = T, J = T, M = F, E = F, B = T) = 0.3 \cdot 0.9 \cdot 0.94 \cdot 0.998 \cdot 0.001 = 2.53 \times 10^{-4}$$
$$P(A = T, J = T, M = F, E = F, B = F) = 0.3 \cdot 0.9 \cdot 0.001 \cdot 0.998 \cdot 0.999 = 2.70 \times 10^{-4}$$
$$P(A = T, J = T, M = F, E = F) = 2.53 \times 10^{-4} + 2.70 \times 10^{-4} = 5.23 \times 10^{-4}$$

And what if we had two missing variables? We'd like to know $P(A = T, J = T, M = F)$. Again, we need to marginalize over all the possible other outcomes of the other variables.

Answer:

$$P(A = T, J = T, M = F) = \sum_{E,B} P(A = T, J = T, M = F, E, B) \\ = P(A = T, J = T, M = F, E = T, B = T) + P(A = T, J = T, M = F, E = T, B = F) \\ + P(A = T, J = T, M = F, E = F, B = T) + P(A = T, J = T, M = F, E = F, B = F) \\ = (0.3 \cdot 0.9 \cdot 0.95 \cdot 0.002 \cdot 0.001) + (0.3 \cdot 0.9 \cdot 0.29 \cdot 0.002 \cdot 0.999) \\ + (0.3 \cdot 0.9 \cdot 0.94 \cdot 0.998 \cdot 0.001) + (0.3 \cdot 0.9 \cdot 0.001 \cdot 0.998 \cdot 0.999) \\ = 5.13 \times 10^{-7} + 1.56 \times 10^{-4} + 2.53 \times 10^{-4} + 2.70 \times 10^{-4} \\ = 6.80 \times 10^{-4}$$

Bayesian Inference

What if we wanted to know the probability that there had been a burglary given that the alarm went off, John called, and Mary did not call? $P(B = T \mid A = T, J = T, M = F)$

Using Bayes' theorem:

$$P(B = T \mid A = T, J = T, M = F) = \frac{P(A = T, J = T, M = F, B = T)}{P(A = T, J = T, M = F)}$$

We need to calculate $P(A = T, J = T, M = F, B = T)$ by marginalizing over the earthquake variable:

$$\begin{aligned} P(A = T, J = T, M = F, B = T) &= P(A = T, J = T, M = F, B = T, E = T) + P(A = T, J = T, M = F, B = T, E = F) \\ &= (0.3 \cdot 0.9 \cdot 0.95 \cdot 0.002 \cdot 0.001) + (0.3 \cdot 0.9 \cdot 0.94 \cdot 0.998 \cdot 0.001) \\ &= 5.13 \times 10^{-7} + 2.53 \times 10^{-4} \\ &= 2.54 \times 10^{-4} \end{aligned}$$

From our previous calculation, we know that $P(A = T, J = T, M = F) = 6.80 \times 10^{-4}$.

Therefore:

$$P(B = T \mid A = T, J = T, M = F) = \frac{2.54 \times 10^{-4}}{6.80 \times 10^{-4}} = 0.374$$

Answer: The probability of a burglary given the observed evidence is approximately 37.4%.

This result demonstrates how Bayesian inference updates our beliefs. Even though burglaries are very rare (0.1% prior probability), the evidence of the alarm sounding and John calling (while Mary does not call) significantly increases our belief in a burglary to about 37%. The fact that Mary didn't call is actually important evidence—if there had been an earthquake, we would expect both neighbors to call more frequently.

4 Approximate Inference

Running exact inference (like done above) is possible for small networks, but for larger Bayesian Networks, it becomes impractical to compute the exact probability of some query. The provided examples are quite small, but are already quite tedious because of the number of intermediate probabilities that you may have to compute. In general, there is no easy way to perform exact inference and exact inference falls the NP-Hard class of problems, that includes the likes of TSP, SAT, and many other problems we've discussed this semester.

Even though computing exact probabilities may be hard, we can still approximate the probabilities in a straightforward way.

4.1 Monte Carlo Methods

Bayesian Networks represent joint probability distributions in a compact way. The key insight for approximate inference methods, is that we can sample from that probability distribution. Instead of computing $P(X|E = e)$ exactly, we generate many samples from the joint distribution and count how often our query variable X takes specific values when the evidence $E = e$ is observed.

The Law of Large Numbers ensures that our sample-based estimates will converge to the true probabilities as the number of samples increases:

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} f(x^{(i)}) = \mathbb{E}[f(X)]$$

where $x^{(i)}$ are samples from the distribution and f is any function of interest.

4.2 Prior Sampling

The simplest approach is **prior sampling**, where we generate samples from the joint distribution by sampling each variable in topological order according to its conditional probability given its parents.

Algorithm 1 Prior Sampling

Order variables topologically: $X_1, X_2, \ldots, X_n$ (parents before children)
for each variable X_i in order **do**
    Sample $x_i \sim P(X_i|\text{parents}(X_i))$
end for
return complete assignment $(x_1, x_2, \ldots, x_n)$

4.3 Rejection Sampling

Prior sampling becomes inefficient when we need to condition on evidence, especially rare events. **Rejection sampling** addresses this by filtering samples that match our evidence.

Algorithm 2 Rejection Sampling

Initialize count $N_{\text{accepted}} = 0$, count $N_{\text{query}} = 0$
while $N_{\text{accepted}} <$ desired sample size **do**
    Generate sample $\mathbf{x} = (x_1, \ldots, x_n)$ using prior sampling
    if $\mathbf{x}$ matches evidence $\mathbf{e}$ (i.e., $x_i = e_i$ for all evidence variables) **then**
        $N_{\text{accepted}} = N_{\text{accepted}} + 1$
        if query condition satisfied **then**
            $N_{\text{query}} = N_{\text{query}} + 1$
        end if
    end if
end while
return $\hat{P}(\text{query}|\mathbf{e}) = \frac{N_{\text{query}}}{N_{\text{accepted}}}$

Rejection Sampling creates an assignment of each node in the Bayesian Network using prior sampling. If that **does not** match the evidence provided, then it does not provide useful information for our query, and we can reject that sample (i.e., we don't use it). If it does match our evidence, then we can check if the queried variable or condition is satisfied. The final probability $P(\text{query}|\text{evidence})$ is simply the number of relevant samples (samples where the evidence is true) where the query is also true.