

# Artificial Intelligence

## Bayesian Networks

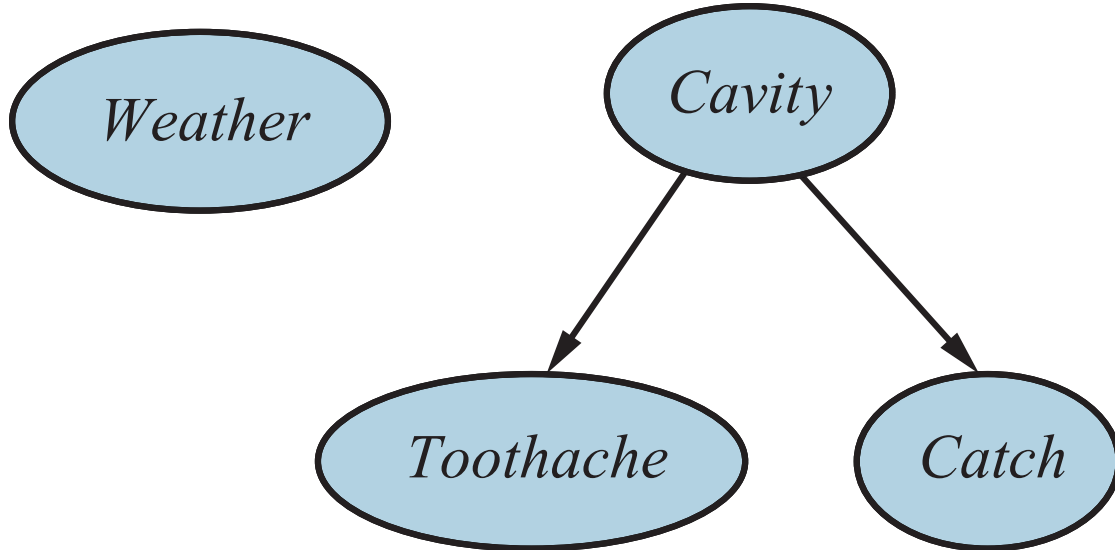
Christopher Simpkins

### 1 Representation of Uncertain Knowledge (AIMA 13.1–13.2)

A Bayesian network is a directed graph in which each node is annotated with quantitative probability information. The full specification is as follows:

1. Each node corresponds to a random variable, which may be discrete or continuous.
2. Directed links or arrows connect pairs of nodes. If there is an arrow from node  $X$  to node  $Y$ , then  $X$  is said to be a parent of  $Y$ . The graph has no directed cycles and hence is a directed acyclic graph, or DAG.
3. Each node  $X_i$  has associated probability information  $\theta(X_i|Parents(X_i))$  that quantifies the effect of the parents on the node using a finite number of parameters.

### 2 Bayesian Network Topology



- The topology of the network – the set of nodes and links – specifies the conditional independence relationships that hold in the domain.
  - *Toothache* and *Catch* are conditionally independent given *Cavity*.
- Intuitively, and arrow  $X \rightarrow Y$  means  $X$  has a direct influence on  $Y$  – so parents should be **causes** of effects.

It is usually easy for a domain expert to decide what direct influences exist in the domain – much easier, in fact, than actually specifying the probabilities themselves.

### 3 Example: Earthquake vs. Burglary

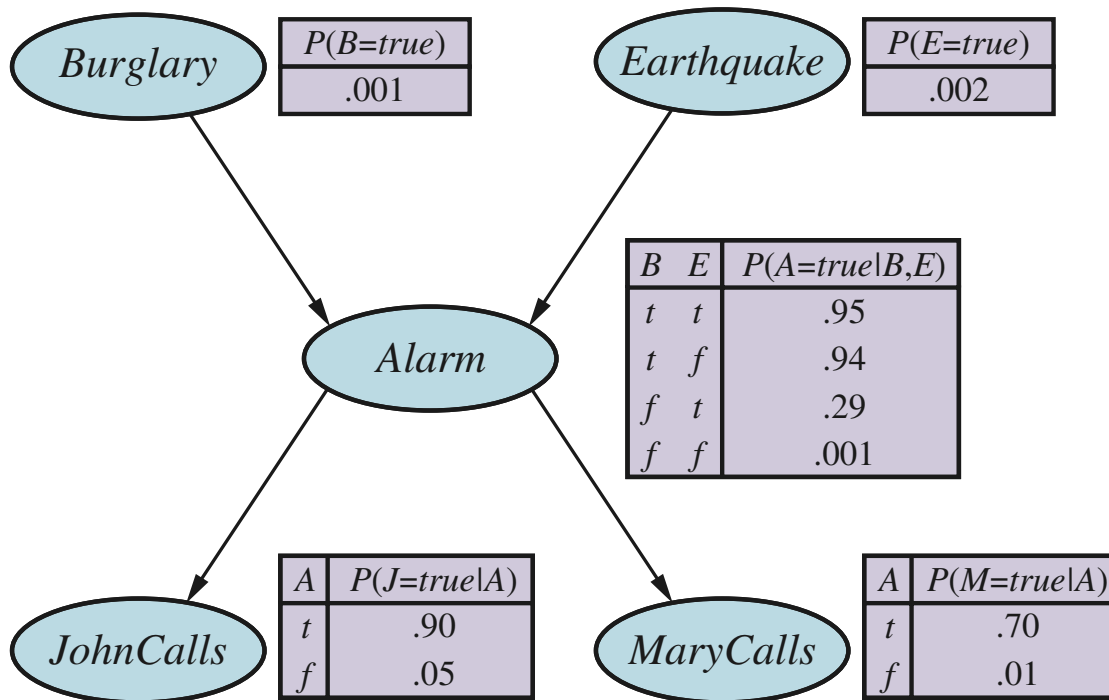
You have a burglar alarm fairly reliable at detecting a burglary, but is occasionally set off by minor earthquakes.

- Two neighbors, John and Mary, who promise to call when they hear the alarm.
- John nearly always calls when he hears the alarm, but sometimes confuses the telephone ringing with the alarm and calls then, too.
- Mary, on the other hand, likes rather loud music and often misses the alarm altogether.

Given the evidence of who has or has not called, we would like to estimate the probability of a burglary.

### 4 Bayes Net for Earthquake vs. Burglary Reasoning

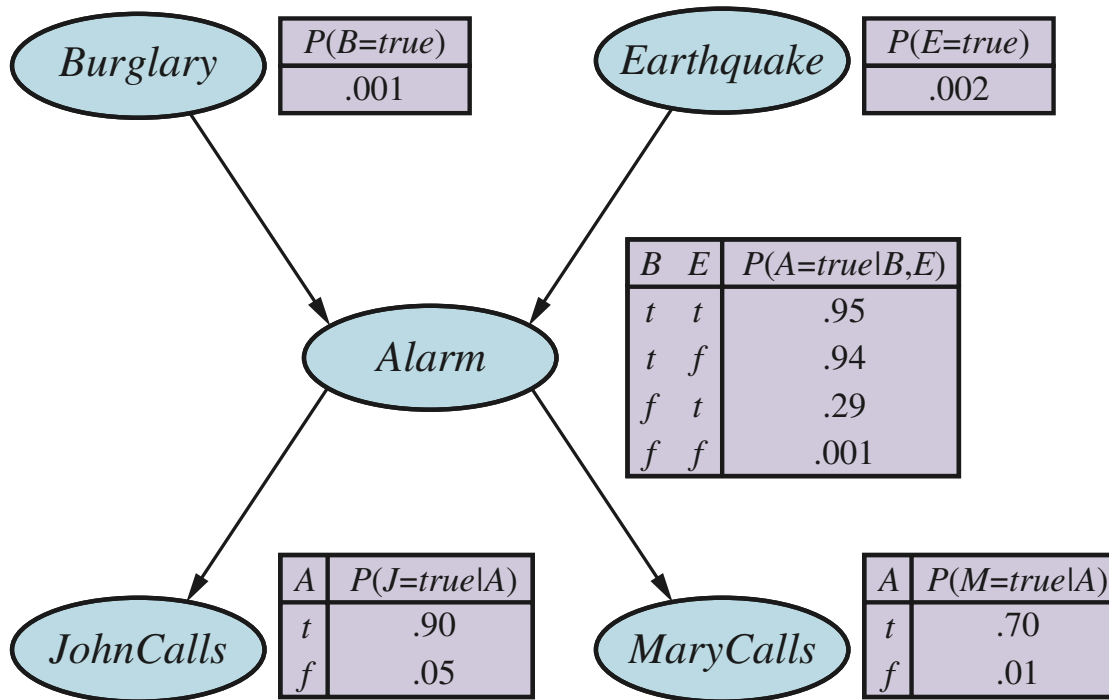
The **syntax** of a Bayes net consists of a directed acyclic graph (DAG) with some local probability information attached to each node.



The full joint distribution for all the variables is defined by the topology and the local probability information recorded in conditional probability tables (CPTs).

### 5 Conditional Probability Tables

Each row in a CPT contains the conditional probability of each node value for a conditioning case. A conditioning case is a possible combination of values for the parent nodes – a miniature possible world.



- Each row must sum to 1, because the entries represent an exhaustive set of cases for the variable.
- For Boolean variables, once you know the probability of *true* is  $p$ , the probability of *false* must be  $1 - p$ , so we often omit the second number.
- In general, a table for a Boolean variable with  $k$  Boolean parents contains  $2^k$  independently specifiable probabilities.
- A node with no parents has only one row, representing the prior probabilities of each possible value of the variable.

## 6 Many Possibilities, Few Variables of Interest

Network does not explicitly represent Mary currently listening to loud music or telephone ringing and confusing John.

- These factors are summarized in the uncertainty associated with the links from *Alarm* to *JohnCalls* and *MaryCalls*.
- This shows both laziness and ignorance in operation: a lot of work to find out the likelihood of those factors, and we have no reasonable way to obtain the relevant information anyway.

The probabilities actually summarize a potentially infinite set of circumstances:



- The alarm might fail to sound (humidity, power failure, dead battery, cut wires, a dead mouse stuck in the bell, etc.) or
- John or Mary might fail to call and report it (out to lunch, on vacation, temporarily deaf, passing helicopter, etc.).

In this way, a small agent can cope with a very large world, at least approximately.

## 7 Semantics of Bayesian Networks

The **semantics** defines how the syntax – a DAG with local probabilities – corresponds to a joint distribution over the variables of the network.

A Bayes net contains:

- $n$  variables,  $X_1, \dots, X_n$ , and
- (implicit) joint distributions  $P(X_1 = x_1 \wedge \dots \wedge X_n = x_n)$ , or  $P(x_1, \dots, x_n)$ .

Each entry in the joint distribution is defined by:

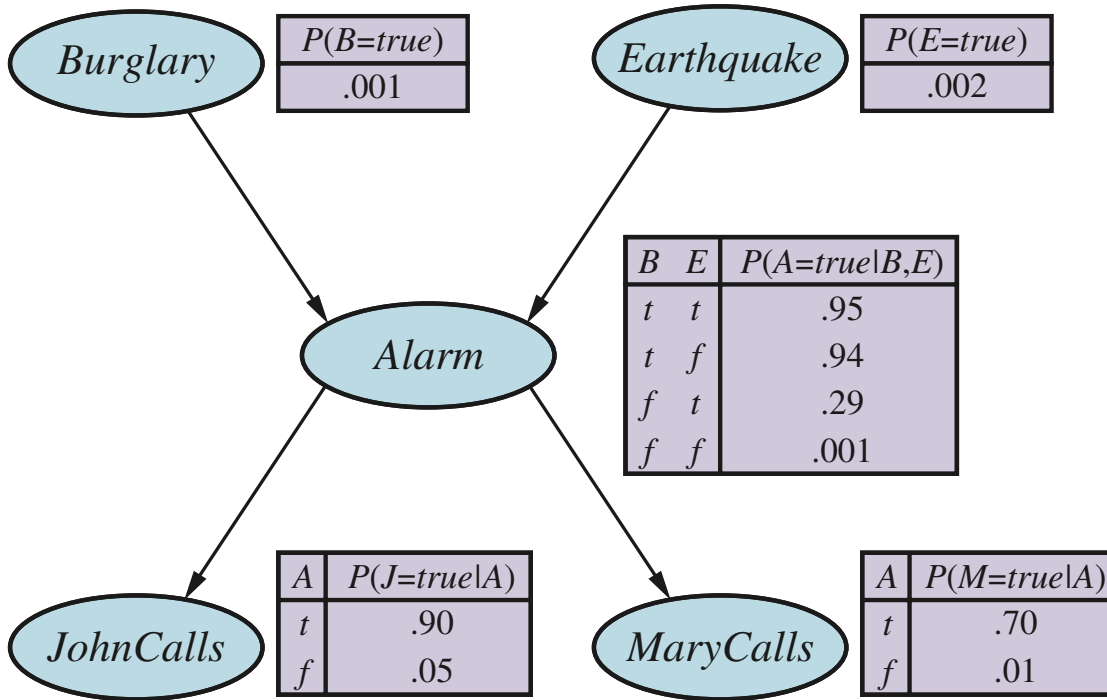
$$P(x_1, \dots, x_n) = \prod_{i=1}^n \theta(x_i | \text{parents}(X_i)) \quad (1)$$

where  $\text{parents}(X_i)$  denotes the values of  $\text{Parents}(X_i)$  that appear in  $x_1, \dots, x_n$ .

So each entry in the joint distribution is the product of appropriate elements of the local CPTs in the Bayes net.

## 8 Example: An Alarm, Two Calls, but No Events of Interest

What is the probability that John and Mary call, but no earthquake or burglary occur?



- Take each entry  $i$  in each CPT  $\theta$  to mean  $P(x_i | \text{parents}(X_i))$ 
  - The entries in the CPTs must be accurate conditional probabilities for the variables given their parents for the Bayes net to be useful in performing probabilistic inference.
- Use those values in the calculation of the joint probability using

Using the abbreviations  $j, m, a, b$  and  $e$ :

$$\begin{aligned}
 P(x_1, \dots, x_n) &= \prod_{i=1}^n P(x_i | \text{parents}(X_i)) & (13.2) \\
 P(j, m, a, \neg b, \neg e) &= P(j|a)P(m|a)P(a|\neg b, \neg e)P(\neg b)Pr(\neg e) \\
 &= 0.90 \times 0.70 \times 0.001 \times 0.99 \times 0.98 \\
 &= 0.000628
 \end{aligned}$$

## 9 Constructing Bayesian Networks

A Bayesian network is a correct representation of the domain only if each node is conditionally independent of its other predecessors in the node ordering, given its parents. Mathematically, if  $Parents(X_i) \subseteq \{X_{i-1}, \dots, X_1\}$ , then:

$$P(X_i | X_{i-1}, \dots, X_1) = P(X_i | Parents(X_i)) \quad (13.3)$$

We can construct a valid Bayes net with this methodology:

1. Nodes: First determine the set of variables that are required to model the domain. Now order them,  $\{X_1, \dots, X_n\}$ . Any order will work, but the resulting network will be more compact if the variables are ordered such that causes precede effects.

- For the Burglary-Earthquake domain,  $B, E, A, J, M$  or  $E, B, A, M, J$  work.

2. Links: For  $i = 1$  to  $n$  do:

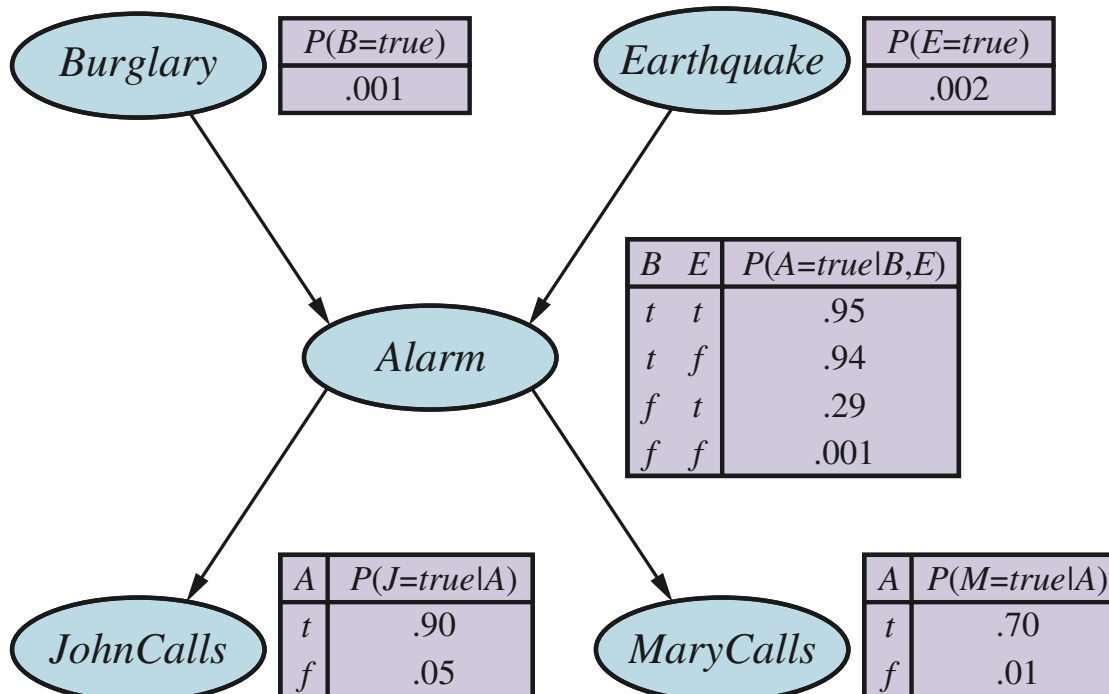
- Choose a minimal set of parents for  $X_i$  from  $X_1, \dots, X_{i-1}$ , such that Equation (13.3) is satisfied.
- For each parent insert a link from the parent to  $X_i$ .
- CPTs: Write down the conditional probability table,  $P(X_i | Parents(X_i))$ .

Intuitively, the parents of node  $X_i$  should contain all those nodes in  $\{X_1, \dots, X_{i-1}\}$  that directly influence  $X_i$ .

## 10 Knowledge Engineering Considerations in Bayes Nets

Suppose we have completed the network except for the choice of parents for *MaryCalls*. We know:

- *MaryCalls* is influenced by Burglary or Earthquake, but not directly. Our domain knowledge tells us that these events influence Marys calling behavior only through their effect on the alarm.
- Also, given the state of the alarm, whether John calls has no influence on Marys calling.



Formally, we believe that the following conditional independence statement holds:  
 Thus,  $P(MaryCalls, JohnCalls, Alarm, Earthquake, Burglary) = P(MaryCalls | Alarm)$ . (2)  
 Because

- each node is connected only to earlier nodes, this construction method guarantees that the network is acyclic, and
- there are no redundant probability values,

there is no chance for inconsistency: it is impossible for the knowledge engineer or domain expert to create a Bayesian network that violates the axioms of probability.

## 11 Compactness of Bayesian Network Models

Bayes nets are complete and nonredundant, but their **compactness** is their "secret sauce."

- Bayes nets are a kind of **locally structured**, or **sparse** system.
  - In locally structured systems, each component interacts only with a bounded number,  $k$ , of other components, where  $k \ll n$ , the total number of components.
  - Local structure typically results in linear rather than exponential complexity.

Example: assume boolean variables.

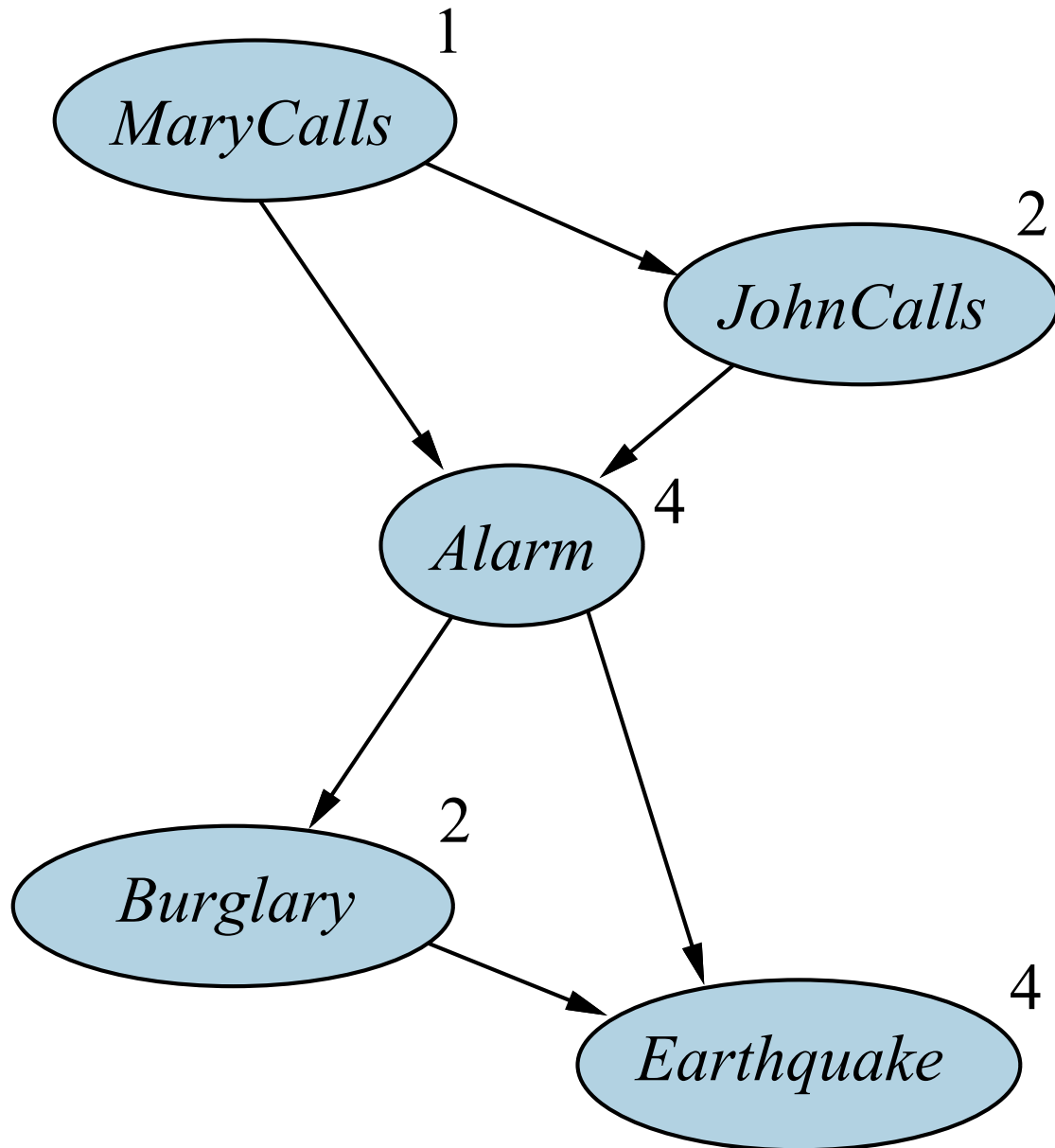
- With  $n$  variables, the full joint distribution is  $2^n$  numbers.
  - With 30 variables that's  $2^{30} = 1,073,741,824$  numbers.
- With  $n$  variables, each of which is influenced by  $k$  parents, each CPT is  $2^k$  numbers and a full Bayesian network has  $n \cdot 2^k$  numbers.
  - With 30 variables that's  $30 \cdot 2^5 = 960$  numbers.

## 12 Effects of Node Ordering

We saw earlier that  $B, E, A, J, M$  is a good node ordering because causes come before effects. With this node ordering we get 10 conditional probabilities. What if we choose a different ordering, like  $M, J, A, B, E$ ?

- Add *MaryCalls*: No parents.
- Add *JohnCalls*: If Mary calls, alarm probably sounded, which increases prob that John calls. So *JohnCalls* needs *MaryCalls* as a parent.
- Add *Alarm*: The more calls, the more likely alarm sounded, so *Alarm* needs *MaryCalls* and *JohnCalls* as parents.
- Add *Burglary*: Given knowledge of alarm, calls irrelevant:  $P(Burglary | Alarm, JohnCalls, MaryCalls) = P(Burglary | Alarm)$ . So just *Alarm* as parent.
- Add *Earthquake*: With alarm, earthquake more likely. But burglary explains alarm, and probability of an earthquake only slightly higher than normal. So need both *Alarm* and *Burglary* as parents.

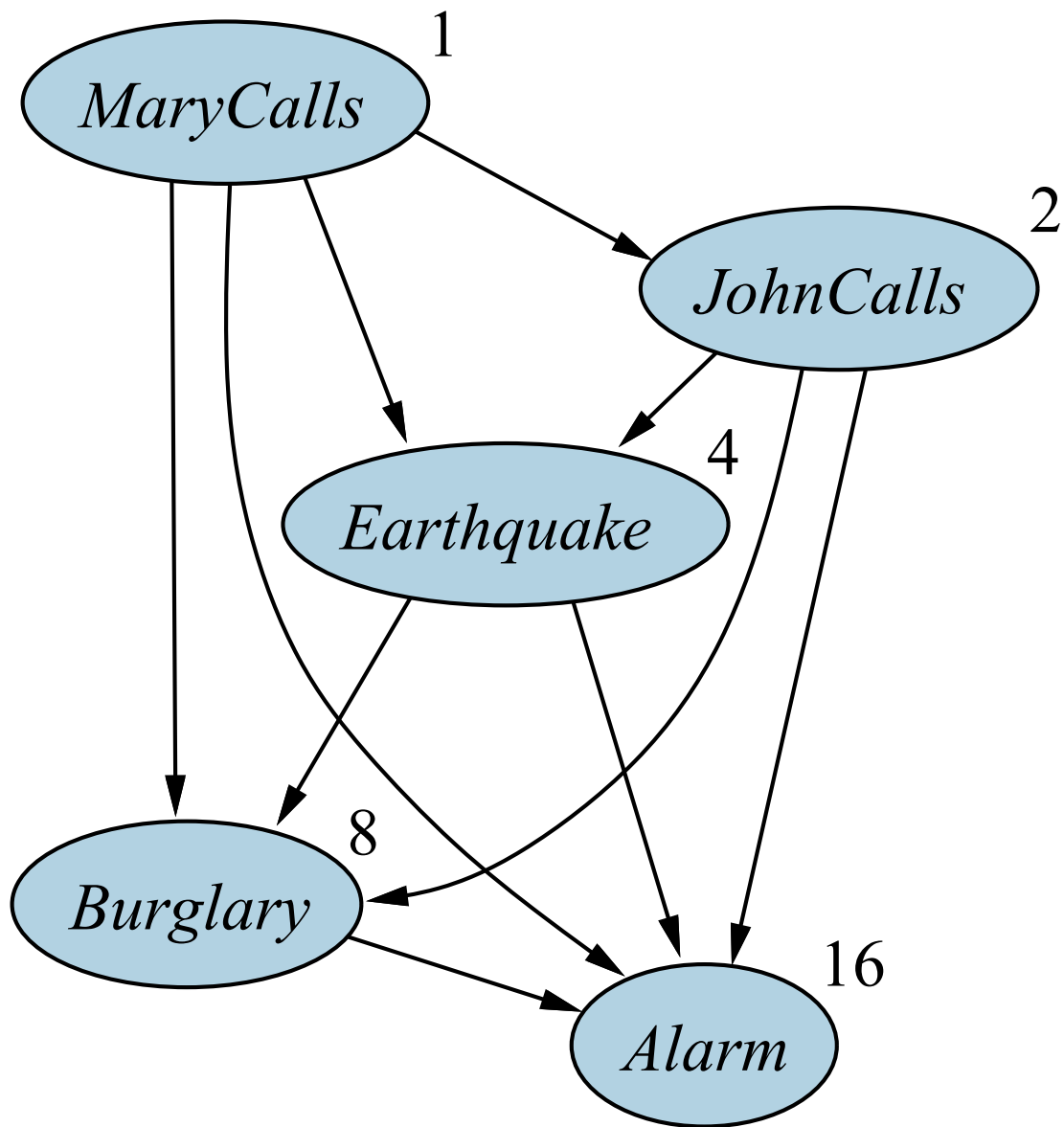
We end up with 13 conditional probabilities.



### 13 Knowledge Engineering in Probabilistic Systems

There is still knowledge engineering involved, which includes design choices.

- You may choose to trade a small amount of accuracy for the improved performance of leaving out an influence variable.
  - If there’s a large earthquake, John and Mary probably won’t call even if they hear the alarm. Could choose to include links from *Earthquake* to *JohnCalls* and *MaryCalls*. This would increase accuracy slightly, but probably not worth the extra complexity.
- And as we just saw, node ordering matters, potentially quite a bit.
  - With node ordering *M, J, E, B, A* the resulting Bayes net requires 31 probabilities – same as the full joint distribution.



Any Bayes net represents the same joint distribution, but some are far more efficient.

> Key takeaway: stick to causal models. They are easier to specify, easier to get right, and they lead to more efficient Bayes nets.

## 14 Exact Inference in Bayesian Networks (AIMA 13.3)

Most common task in probabilistic inference: compute the **posterior probability** of a set of **query variables** given some **event** represented as a set of **evidence variables**.

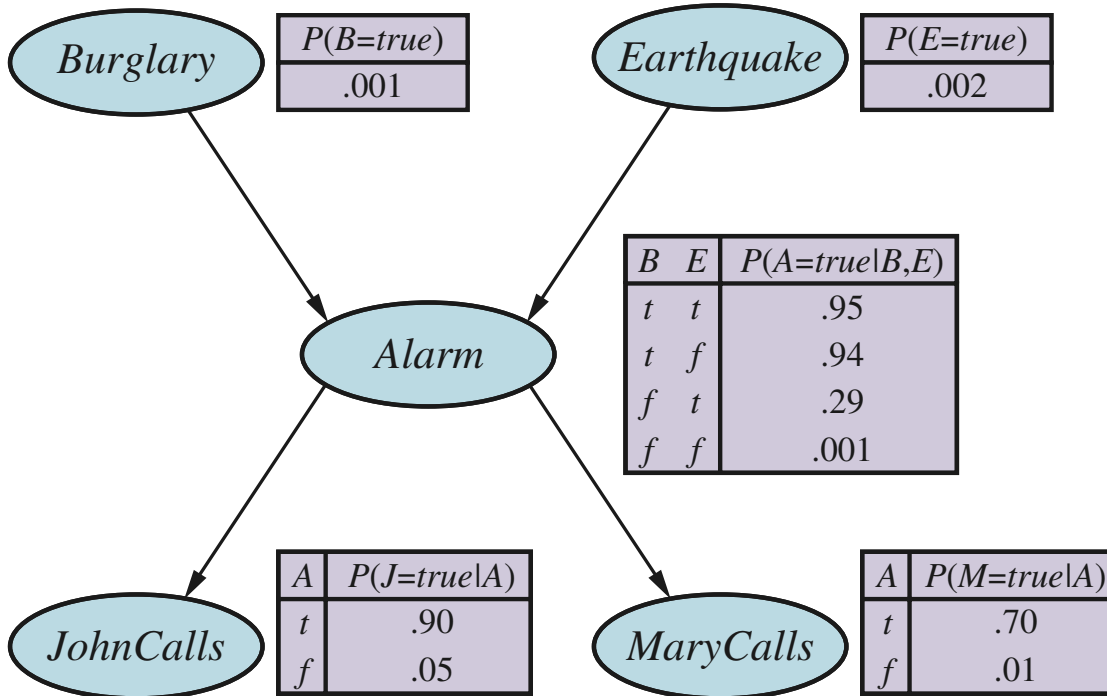
Notation:

- Query variable:  $X$
- Set of evidence variables:  $E = \{E_1, \dots, E_m\}$
- Particular observed event:  $e$

- Hidden (nonevidence, nonquery) variables:  $Y = \{Y_1, \dots, Y_l\}$
- Typical query:  $P(X | e)$

Example:

- $X$  is the boolean random variable *Burglary*
- $E = \{JohnCalls, MaryCalls\}$
- $e = \{JohnCalls = true, MaryCalls = true\}$
- $Y = \{EarthQuake, Alarm\}$



$$P(\text{Burglary} \mid \text{JohnCalls} = \text{true}, \text{MaryCalls} = \text{true}) = \langle 0.284, 0.716 \rangle . \quad (3)$$

## 15 Inference by Enumeration

Recall that we can use the full joint distribution to answer any query:

$$P(X | e) = \frac{P(e | X)P(X)}{P(e)} = \frac{P(e, X)}{P(e)} = \frac{P(X, e)}{P(e)} = \alpha P(X, e) = \alpha \sum_y P(X, e, y) \quad (12.9)$$

And that a Bayes net completely represents the full joint distribution, so we can reduce the computation of a joint to:

$$P(x_1, \dots, x_n) = \prod_{i=1}^n P(x_i | \text{parents}(X_i)) \quad (13.2)$$

Using these two equations we can enumerate the appropriate probabilities to calculate the answer to any probabilistic query.

- In particular, we can get the answer by computing sums of products of conditional probabilities from a Bayes net.

## 16 Example: $P(\text{Burglary} \mid \text{JohnCalls} = \text{true}, \text{MaryCalls} = \text{true})$ .

Using abbreviations and substituting into Eq 12.9 above ( $e$  and  $a$  are hidden):

$$P(B \mid j, m) = \alpha P(B, j, m) = \alpha \sum_e \sum_a P(B, j, m, e, a) \quad (4)$$

Then we substitute Eq 13.2 for  $P(B, j, m, e, a)$  to get (only showing Burglary=true by the notation  $b$ ):

$$P(b \mid j, m) = \alpha \sum_e \sum_a P(b)P(e)P(a \mid b, e)P(j \mid a)P(m \mid a) \quad (1)$$

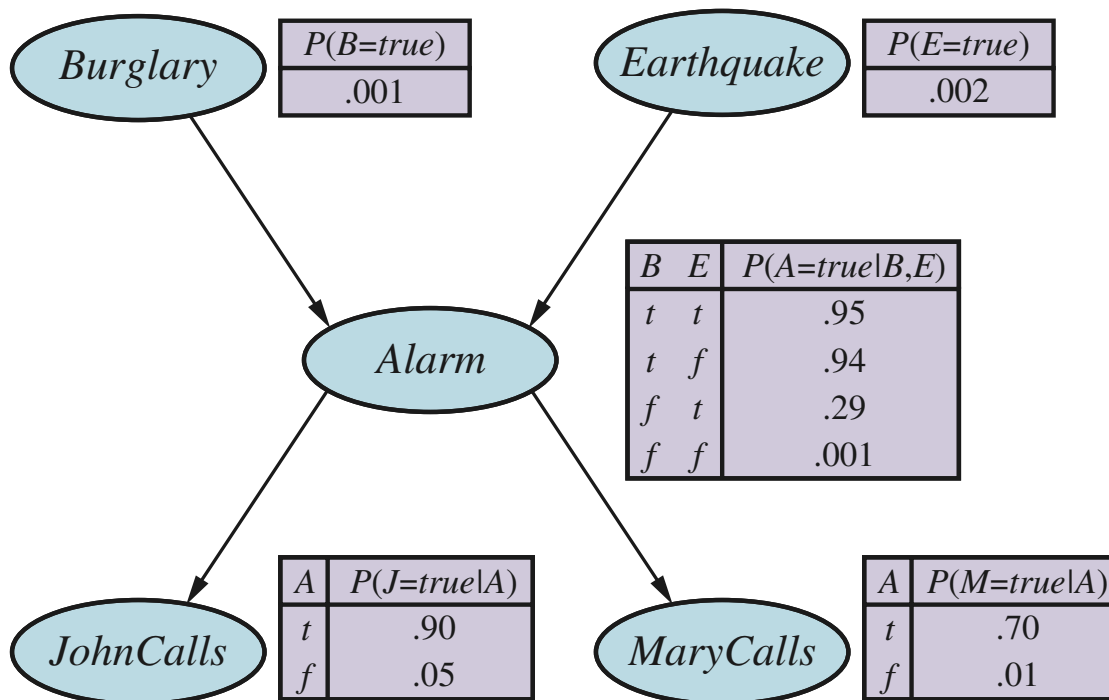
$$= \alpha P(b) \sum_e \sum_a P(e)P(a \mid b, e)P(j \mid a)P(m \mid a) \quad (2)$$

$$= \alpha P(b) \sum_e P(e) \sum_a P(a \mid b, e)P(j \mid a)P(m \mid a) \quad (3)$$

1. Substitute Eq 13.2 for  $P(B, j, m, e, a)$
2. Pull out  $P(b)$  from summations because it doesn't depend on the other variable and is thus a constant in all the summation terms.
3. Pull out  $P(e)$  from the summation over the  $a$  values because each value of  $e$  doesn't depend on the other variables in the summation over the  $a$  values and is thus a constant in the summation terms over the values of  $a$ .

Steps 2 and 3 above reduce the complexity of the computation from  $O(n2^n)$  to  $O(2^n)$ .

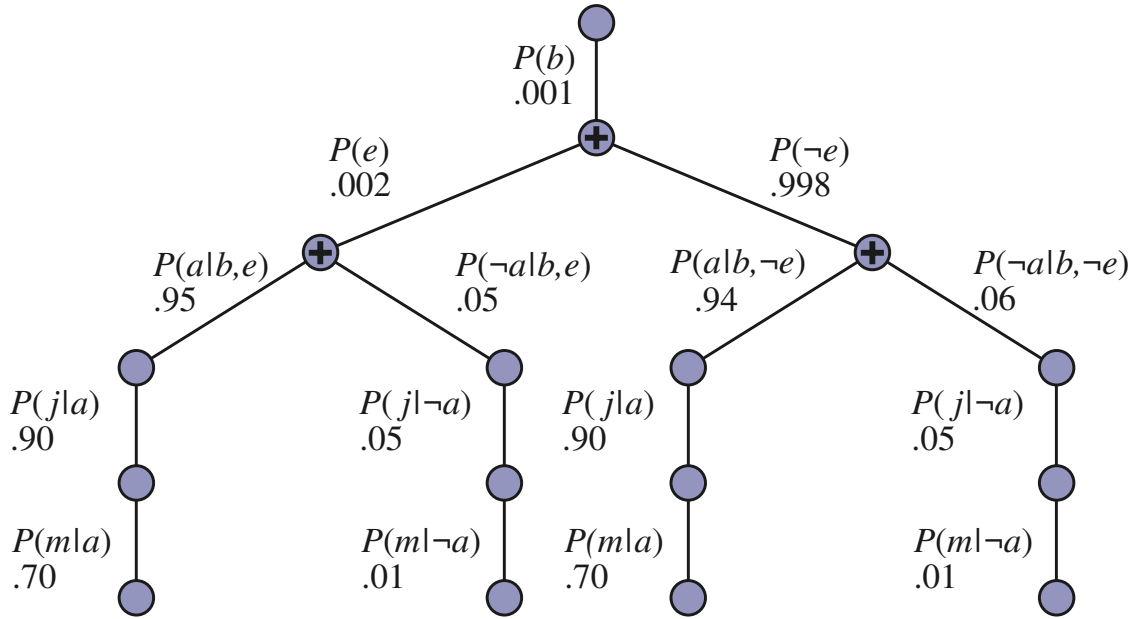
## 17 Calculation of $P(b \mid j, m)$



Substituting the values from the CPTs in the Bayes net into

$$\alpha P(b) \sum_e P(e) \sum_a P(a \mid b, e)P(j \mid a)P(m \mid a) \quad (5)$$

we get the expression tree:



## 18 Enumeration Algorithm

The ENUMERATION-ASK algorithm evaluates these expression trees using depth-first, left-to-right recursion.

**function** ENUMERATION-ASK( $X$ ,  $\mathbf{e}$ ,  $bn$ ) **returns** a distribution over  $X$

**inputs:**  $X$ , the query variable  
 $\mathbf{e}$ , observed values for variables  $\mathbf{E}$   
 $bn$ , a Bayes net with variables  $vars$

$\mathbf{Q}(X) \leftarrow$  a distribution over  $X$ , initially empty

**for each** value  $x_i$  of  $X$  **do**

$\mathbf{Q}(x_i) \leftarrow$  ENUMERATE-ALL( $vars$ ,  $\mathbf{e}_{x_i}$ )  
     where  $\mathbf{e}_{x_i}$  is  $\mathbf{e}$  extended with  $X = x_i$

**return** NORMALIZE( $\mathbf{Q}(X)$ )

**function** ENUMERATE-ALL( $vars$ ,  $\mathbf{e}$ ) **returns** a real number

**if** EMPTY?( $vars$ ) **then return** 1.0

$V \leftarrow$  FIRST( $vars$ )

**if**  $V$  is an evidence variable with value  $v$  in  $\mathbf{e}$

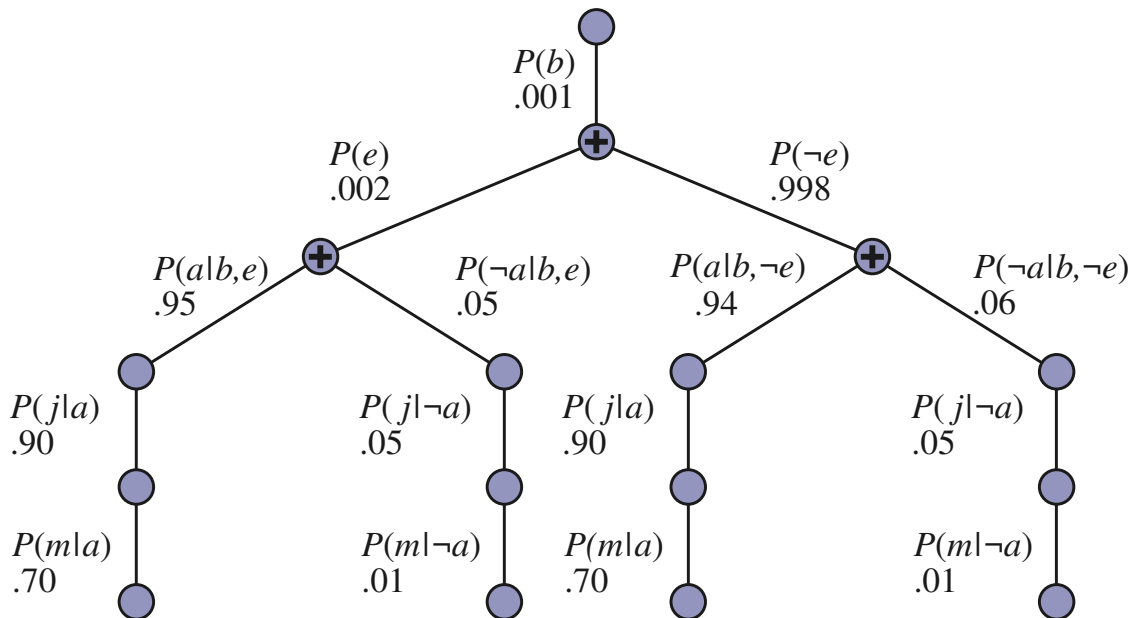
**then return**  $P(v|parents(V)) \times$  ENUMERATE-ALL(REST( $vars$ ),  $\mathbf{e}$ )

**else return**  $\sum_v P(v|parents(V)) \times$  ENUMERATE-ALL(REST( $vars$ ),  $\mathbf{e}_v$ )  
     where  $\mathbf{e}_v$  is  $\mathbf{e}$  extended with  $V = v$

Unfortunately, its time complexity is  $O(s^n)$ . But we can improve it ...

## 19 Repeated Calculations

Notice that the subexpressions for the products  $P(j | a)P(m | a)$  and  $P(j | \neg a)P(m | \neg a)$  are computed twice, once for each value of  $E$ .



## 20 Variable Elimination

The enumeration algorithm can be improved substantially by eliminating repeated calculations.

- Idea: do the calculation once and save the results for later use.
- This is a form of dynamic programming.
- Several versions of this approach; variable elimination algorithm is simplest.

Variable elimination works by evaluating expressions such as

$$P(b | j, m) = \alpha P(b) \sum_e P(e) \sum_a P(a | b, e) P(j | a) P(m | a) \quad (13.5)$$

in right-to-left order (that is, bottom up in the expression tree), storing intermediate results, and only doing summations for portions of the expression that depend on the variable.

## 21 Example: Variable Elimination in Burglary Network

First, annotate the \*factor\*s in the expression for the network:

$$P(B | j, m) = \alpha \underbrace{P(B)}_{f_1(B)} \sum_e \underbrace{P(e)}_{f_2(E)} \sum_a \underbrace{P(a | B, e)}_{f_3(A,B,E)} \underbrace{P(j | a)}_{f_4(A)} \underbrace{P(m | a)}_{f_5(A)} \quad (6)$$

- Each factor is a matrix indexed by the values of its argument variables.
- Notice that the factors for  $P(j | a)$  and  $P(m | a)$  do not include  $j$  and  $m$ . This is because the values of  $j$  and  $m$  ( $JohnCalls = true$  and  $MaryCalls = true$ ) are fixed by the query.

So the factors are:

$$f_1(B) = \begin{bmatrix} P(b) \\ P(\neg b) \end{bmatrix} = \begin{bmatrix} 0.001 \\ 0.999 \end{bmatrix} \quad (7)$$

$$f_2(E) = \begin{bmatrix} P(e) \\ P(\neg e) \end{bmatrix} = \begin{bmatrix} 0.002 \\ 0.998 \end{bmatrix} \quad (8)$$

$$f_4(A) = \begin{bmatrix} P(j | a) \\ P(j | \neg a) \end{bmatrix} = \begin{bmatrix} 0.090 \\ 0.05 \end{bmatrix} \quad (9)$$

$$f_5(A) = \begin{bmatrix} P(m | a) \\ P(m | \neg a) \end{bmatrix} = \begin{bmatrix} 0.070 \\ 0.01 \end{bmatrix} \quad (10)$$

$f_3(A, B, E)$  is a little more complicated ...

## 22 $f_3(A, B, E)$

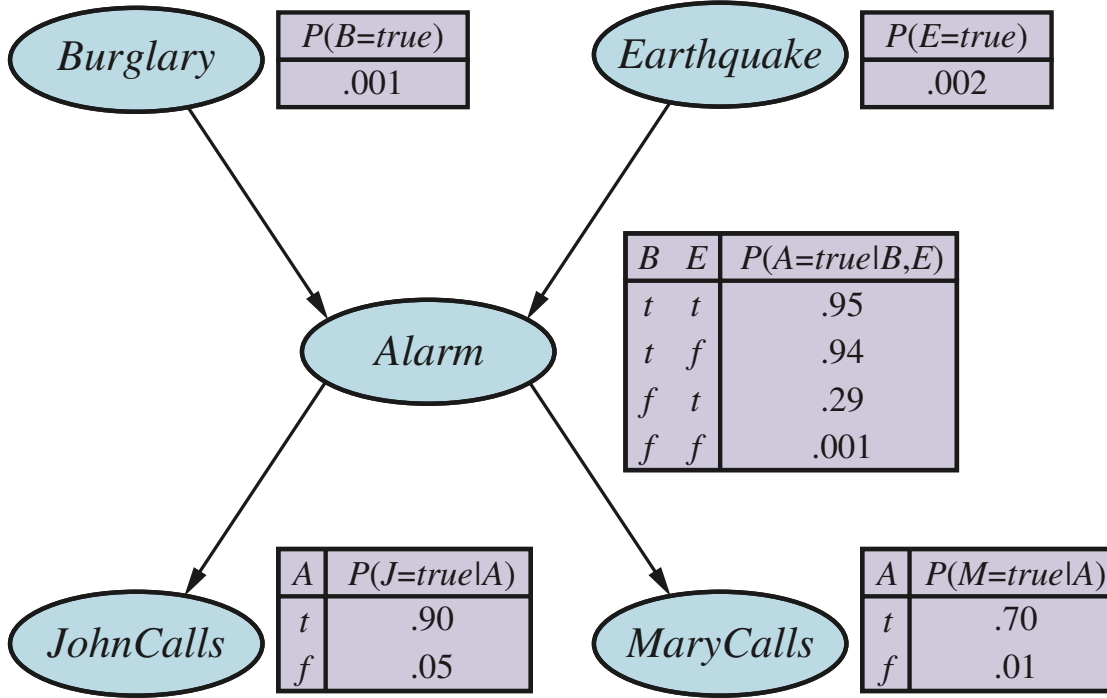
$$P(B | j, m) = \alpha \underbrace{P(B)}_{f_1(B)} \sum_e \underbrace{P(e)}_{f_2(E)} \sum_a \underbrace{P(a | B, e)}_{f_3(A, B, E)} \underbrace{P(j | a)}_{f_4(A)} \underbrace{P(m | a)}_{f_5(A)} \quad (11)$$

$f_3(A, B, E)$  is a  $2 \times 2 \times 2$  matrix (or a rank-3 tensor). Here's one way to think about it:

- First index with  $A$ , yielding two  $2 \times 2$  submatrices (one for each of the two values of  $A$ ).
- Rows of each submatrix is indexed by  $B$  and columns by  $E$ .
- The entries in the submatrices are the values of  $P(A | B, E)$

$$f_3^{(a)}(B, E) = \begin{bmatrix} P(a | b, e) & P(a | b, \neg e) \\ P(a | \neg b, e) & P(a | \neg b, \neg e) \end{bmatrix} = \begin{bmatrix} 0.95 & 0.94 \\ 0.29 & 0.001 \end{bmatrix} \quad (12)$$

$$f_3^{(\neg a)}(B, E) = \begin{bmatrix} P(\neg a | b, e) & P(\neg a | b, \neg e) \\ P(\neg a | \neg b, e) & P(\neg a | \neg b, \neg e) \end{bmatrix} = \begin{bmatrix} 0.05 & 0.06 \\ 0.71 & 0.999 \end{bmatrix} \quad (13)$$



## 23 Factorized Query

From our original query:

$$P(b | j, m) = \alpha P(b) \sum_e P(e) \sum_a P(a | b, e) P(j | a) P(m | a) \quad (13.5)$$

We annotated the factors:

$$P(B | j, m) = \alpha \underbrace{P(B)}_{f_1(B)} \sum_e \underbrace{P(e)}_{f_2(E)} \sum_a \underbrace{P(a | B, e)}_{f_3(A, B, E)} \underbrace{P(j | a)}_{f_4(A)} \underbrace{P(m | a)}_{f_5(A)} \quad (14)$$

And now we substitute the factor expressions for the original expressions so we can manipulate the factors using the **pointwise product** operation, denoted with  $\times$  here:

$$P(B | j, m) = \alpha f_1(B) \times \sum_e f_2(E) \times \sum_a f_3(A, B, E) \times f_4(A) \times f_5(A) \quad (15)$$

Now we are ready to evaluate the expression ...

## 24 Expression Evaluation

Given our factorized query:

$$P(B | j, m) = \alpha f_1(B) \times \sum_e f_2(E) \times \sum_a f_3(A, B, E) \times f_4(A) \times f_5(A) \quad (16)$$

First, sum out A from the pointwise product of  $f_3(A, B, E)$ ,  $f_4(A)$ , and  $f_5(A)$  yielding a new  $2 \times 2$  factor,  $f_6(B, E)$ :

$$f_6(B, E) = \sum_a f_3(A, B, E) \times f_4(A) \times f_5(A)$$

Now the query expression is  $P(B | j, m) = \alpha f_1(B) \times \sum_e f_2(E) \times f_6(B, E)$

Next, sum out  $E$  from the product of  $f_2(E)$  and  $f_6(B, E)$ , yielding a new factor  $f_7(B)$ :

$$f_7(B) = \sum_e f_2(E) \times f_6(B, E)$$

$$= f_2(e) \times f_6(B, e) + f_2(-e) \times f_6(B, -e)$$

Which leaves our final form of the query:  $P(B | j, m) = \alpha f_1(B) \times f_7(B)$

This expression can be evaluated by taking the pointwise product and normalizing the result.

## 25 Operations on Factors

In the preceding manipulations we saw the two basic operations in variable elimination:

1. the pointwise product operation, and
2. summing out hidden variables from products of factors.

Now let's see how we calculate them.

## 26 Pointwise Product Example

The pointwise product of two factors  $f$  and  $g$  yields a new factor  $h$  whose variables are the union of the variables in  $f$  and  $g$  and whose elements are given by the product of the corresponding elements in the two factors.

Given  $X, Y, Z$  boolean variables, result of pointwise product  $f(X, Y) \times g(Y, Z) = h(X, Y, Z)$  is:

$X$	$Y$	$\mathbf{f}(X, Y)$	$Y$	$Z$	$\mathbf{g}(Y, Z)$	$X$	$Y$	$Z$	$\mathbf{h}(X, Y, Z)$
$t$	$t$	.3	$t$	$t$	.2	$t$	$t$	$t$	$.3 \times .2 = .06$
$t$	$f$	.7	$t$	$f$	.8	$t$	$t$	$f$	$.3 \times .8 = .24$
$f$	$t$	.9	$f$	$t$	.6	$t$	$f$	$t$	$.7 \times .6 = .42$
$f$	$f$	.1	$f$	$f$	.4	$t$	$f$	$f$	$.7 \times .4 = .28$
						$f$	$t$	$t$	$.9 \times .2 = .18$
						$f$	$t$	$f$	$.9 \times .8 = .72$
						$f$	$f$	$t$	$.1 \times .6 = .06$
						$f$	$f$	$f$	$.1 \times .4 = .04$

## 27 Summing out Variables

$X$	$Y$	$\mathbf{f}(X, Y)$	$Y$	$Z$	$\mathbf{g}(Y, Z)$	$X$	$Y$	$Z$	$\mathbf{h}(X, Y, Z)$
$t$	$t$	.3	$t$	$t$	.2	$t$	$t$	$t$	$.3 \times .2 = .06$
$t$	$f$	.7	$t$	$f$	.8	$t$	$t$	$f$	$.3 \times .8 = .24$
$f$	$t$	.9	$f$	$t$	.6	$t$	$f$	$t$	$.7 \times .6 = .42$
$f$	$f$	.1	$f$	$f$	.4	$t$	$f$	$f$	$.7 \times .4 = .28$
						$f$	$t$	$t$	$.9 \times .2 = .18$
						$f$	$t$	$f$	$.9 \times .8 = .72$
						$f$	$f$	$t$	$.1 \times .6 = .06$
						$f$	$f$	$f$	$.1 \times .4 = .04$

Summing out a variable from a product of factors is done by adding up the submatrices formed by fixing the variable to each of its values in turn. For example, to sum out  $X$  from  $h(X, Y, Z)$ , we write

$$\begin{aligned} h_2(Y, Z) &= \sum_x h(X, Y, Z) \\ &= h(x, Y, Z) + h(\neg x, Y, Z) \\ &= \begin{bmatrix} .06 & .24 \\ .42 & .28 \end{bmatrix} + \begin{bmatrix} .18 & .72 \\ .06 & .04 \end{bmatrix} \\ &= \begin{bmatrix} .24 & .96 \\ .48 & .32 \end{bmatrix} \end{aligned}$$

## 28 Variable Elimination Algorithm

With these two basic operations, we can implement the variable elimination algorithm:

**function** ELIMINATION-ASK( $X, \mathbf{e}, bn$ ) **returns** a distribution over  $X$   
**inputs:**  $X$ , the query variable  
 $\mathbf{e}$ , observed values for variables  $\mathbf{E}$   
 $bn$ , a Bayesian network with variables  $vars$

```

factors ← []
for each  $V$  in ORDER( $vars$ ) do
    factors ← [MAKE-FACTOR( $V, \mathbf{e}$ )] + factors
    if  $V$  is a hidden variable then factors ← SUM-OUT( $V, factors$ )
return NORMALIZE(POINTWISE-PRODUCT(factors))

```

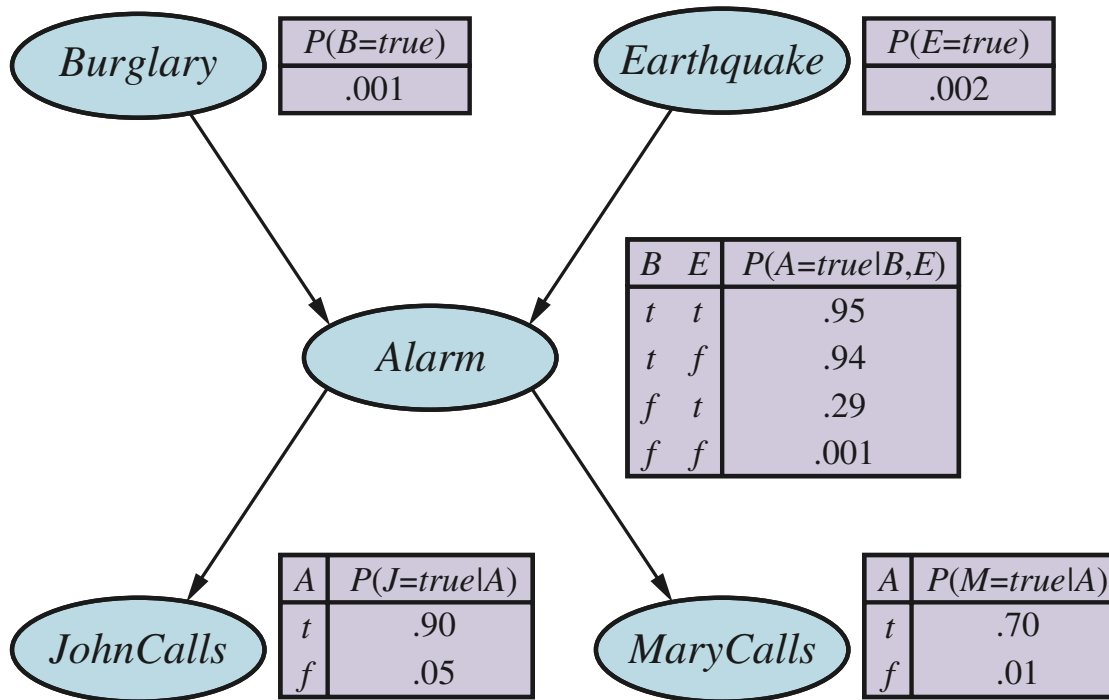
Notes about the ‘order’ function:

- Any ordering works, some orderings lead to more efficient algorithms.
- No tractable algorithm for determining optimal ordering.
- One heuristic: eliminate whichever variable minimizes the size of the next factor to be constructed.
- General rule: every variable that is not an ancestor of a query variable or evidence variable is irrelevant to the query.

## 29 Complexity of Exact Inference in Polytrees

Notice that the Alarm Bayes net is **singly connected**, a.k.a., a **polytree**:

- there is at most one undirected path between any two nodes in the network.

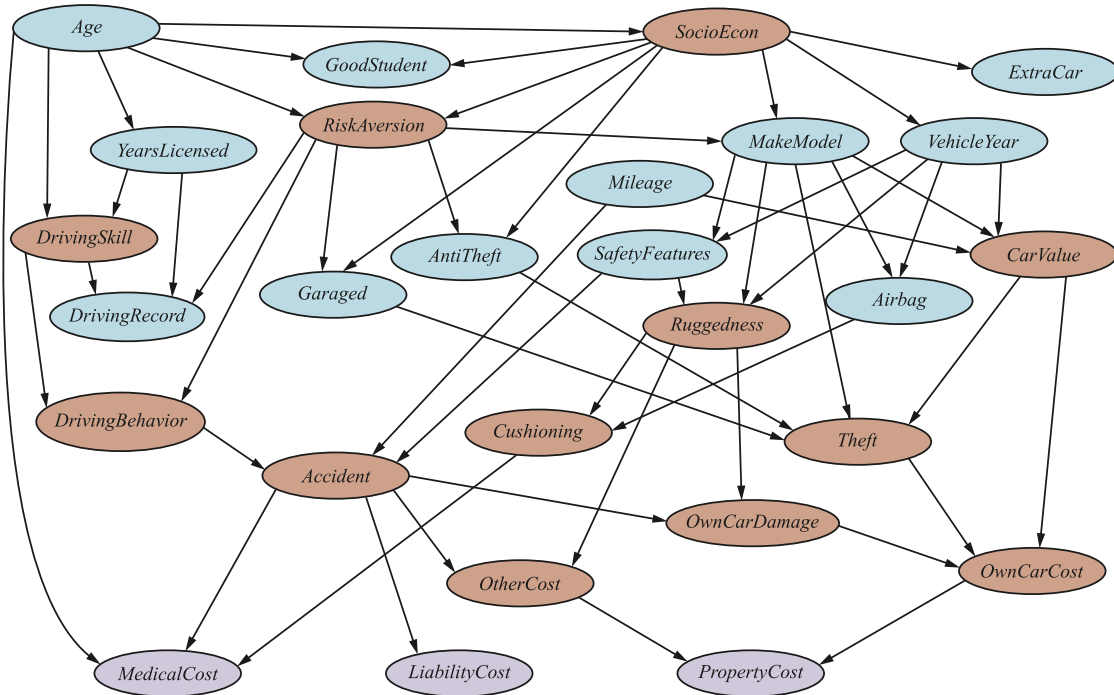


The time and space complexity of polytrees is linear in the size of the network.

- Size of network is defined as number of CPT entries.
- If  $|parents(X_i)| \leq c, \forall i \in n$  for some constant  $c$  and number of nodes  $n$ , then complexity is also linear in number of nodes.

### 30 Complexity of Exact Inference in Multiply-connected Networks

Now consider **multiply-connected** networks such as the car insurance network:



- Variable elimination can have exponential worst-case time and space complexity in multiply-connected networks.
- Since inference in Bayes nets includes inference in propositional logic as a special case, Bayes net inference is **NP-hard**.

## 31 Closing Thoughts

- A Bayesian network is a directed acyclic graph whose nodes correspond to random variables; each node has a conditional distribution for the node, given its parents.
- Bayesian networks provide a concise way to represent conditional independence relationships in the domain.
- A Bayesian network specifies a joint probability distribution over its variables. The probability of any given assignment to all the variables is defined as the product of the corresponding entries in the local conditional distributions. A Bayesian network is often exponentially smaller than an explicitly enumerated joint distribution.
- Many conditional distributions can be represented compactly by canonical families of distributions. Hybrid Bayesian networks, which include both discrete and continuous variables, use a variety of canonical distributions.