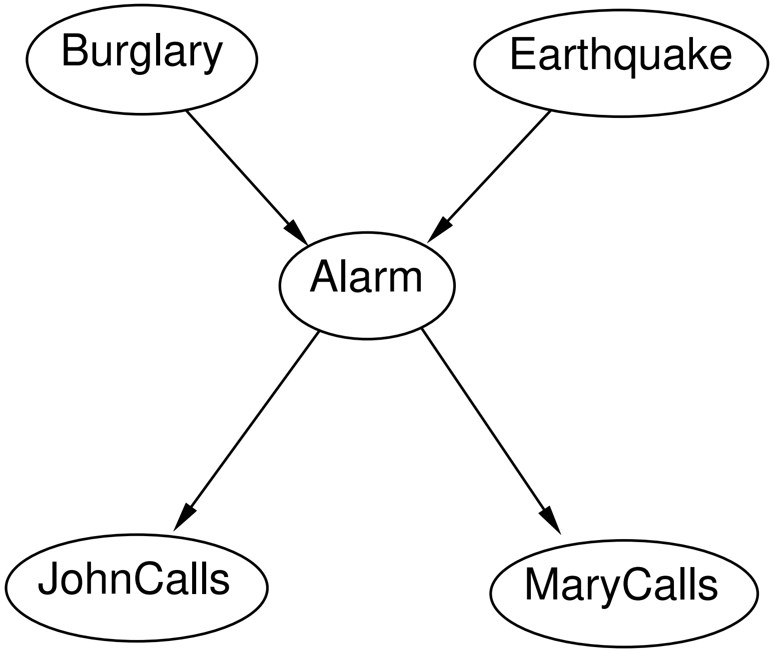
Reasoning with Belief Networks



Directed Acyclic Graph:

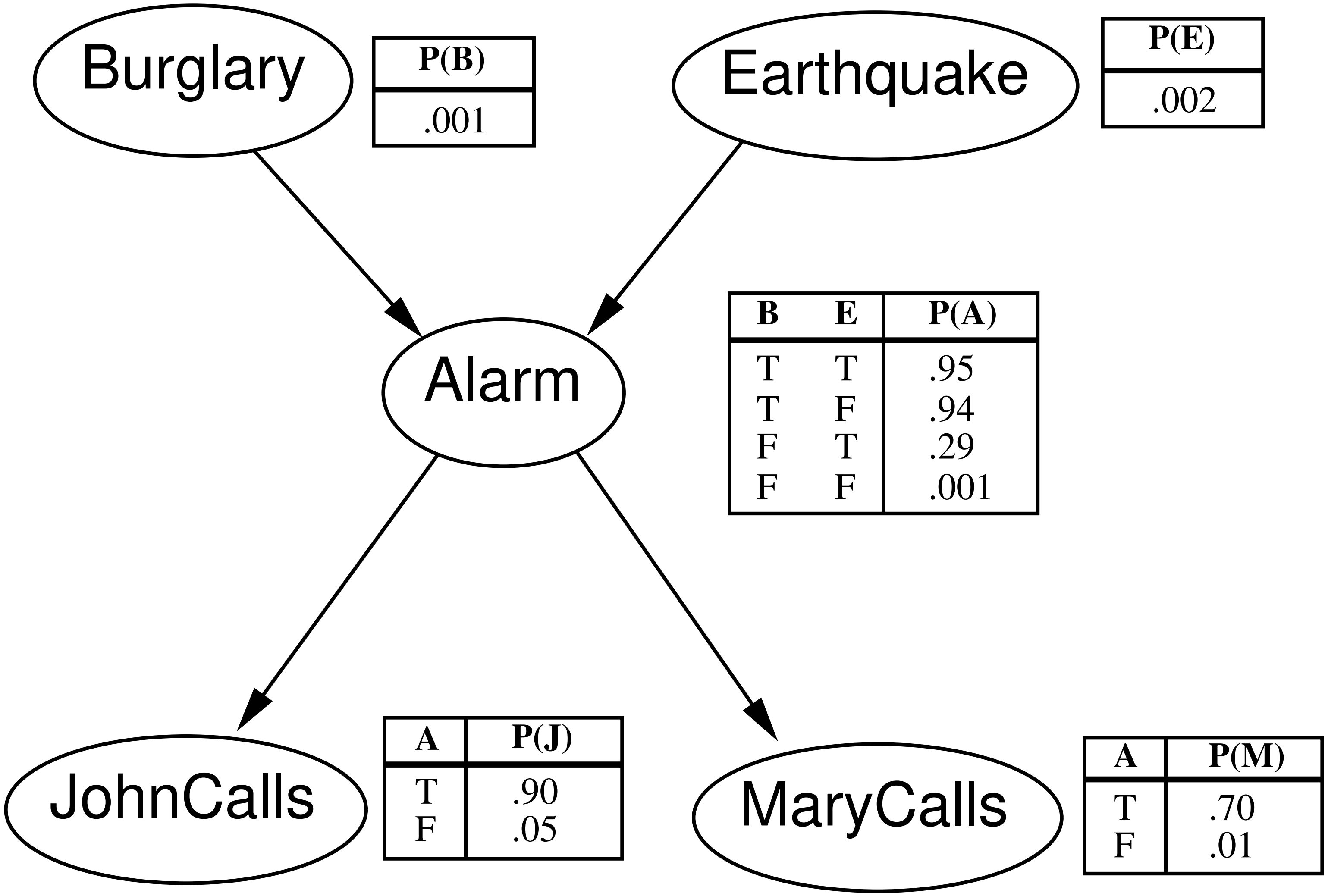
- **Nodes:** one for each random variable
- **Arcs:** one for each direct influence between two random variables.
- **CPT:** each node stores a conditional probability table $P(Node|Parents(Node))$.

Example Conditional Probability Table

Burglary	Earthquake	$\mathbf{P}(Alarm \mid Burglary, Earthquake)$	
		True	False
False	False	0.001	0.999
False	True	0.290	0.710
True	False	0.950	0.050
True	True	0.950	0.050

This table has 4 independent parameters.

Conditional Probability Tables



P(B)
.001

P(E)
.002

B	E	P(A)
T	T	.95
T	F	.94
F	T	.29
F	F	.001

A	P(J)
T	.90
F	.05

A	P(M)
T	.70
F	.01

Conditional Independence Relations in the Belief Network

Let A and B be any two nodes in a belief network such that B is not a descendant of A. Then A and B are conditionally independent, given the parents of A:

$$\mathbf{P}(A, B|Parents(A)) = \mathbf{P}(B|Parents(A)) \cdot \mathbf{P}(A|Parents(A))$$

Examples:

- Burglary and Earthquake are independent (Burglary has no parents).
- JohnCalls and MaryCalls are independent given Alarm.

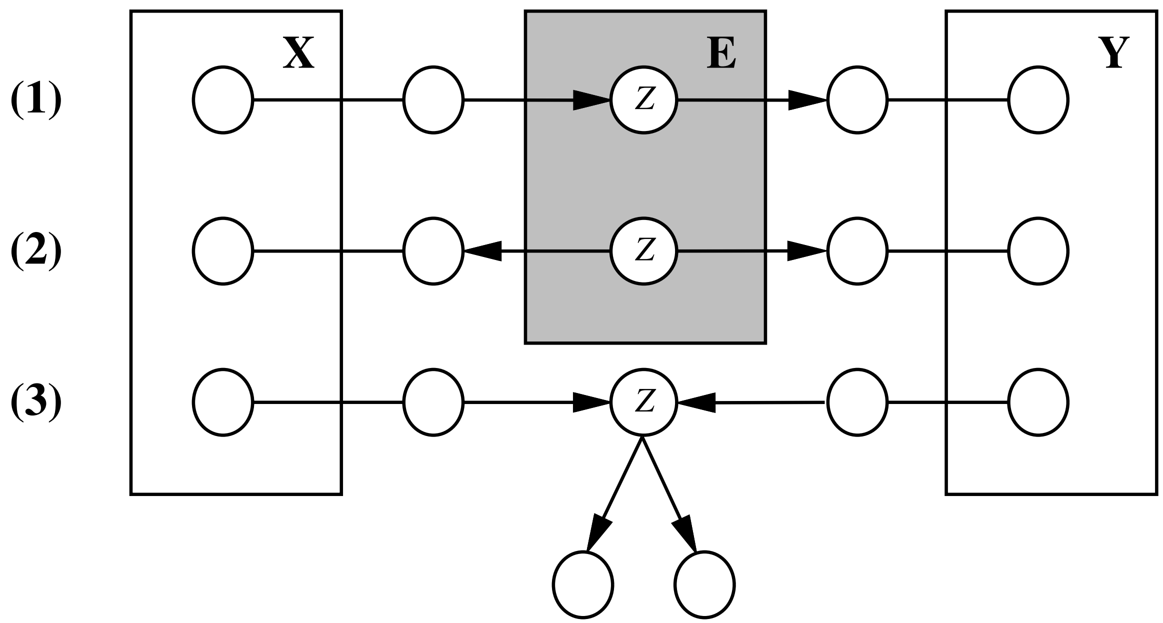
Conditional Independence and D-Separation

D-separation implies Conditional Independence

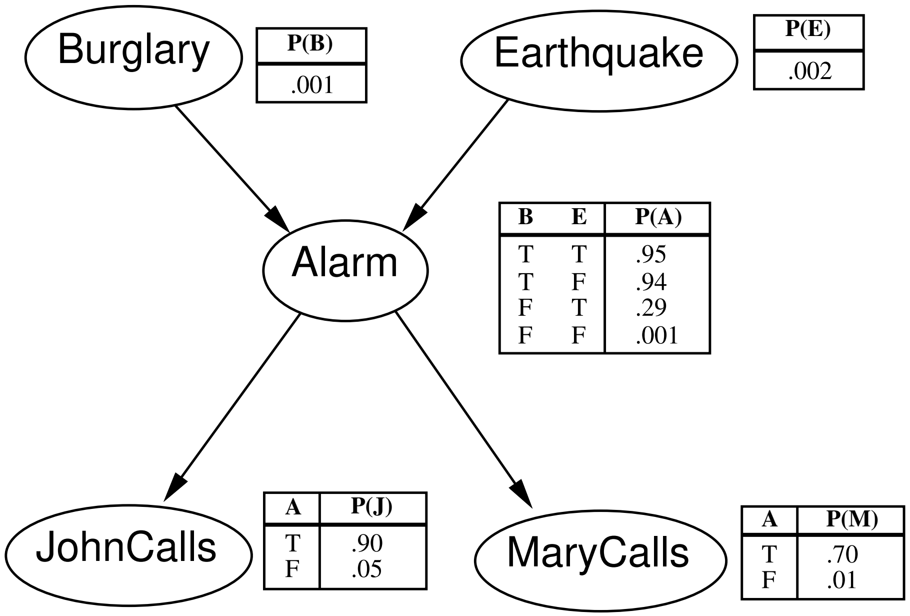
If node X and node Y are D-separated by evidence E, then X and Y are conditionally independent given E.

Definition of D-separation

Two nodes X and Y are **D-Separated** by a set of evidence nodes E if every undirected path from X to Y is **blocked** by one of the following:



More Examples



P(B)
.001

P(E)
.002

B	E	P(A)
T	T	.95
T	F	.94
F	T	.29
F	F	.001

A	P(J)
T	.90
F	.05

A	P(M)
T	.70
F	.01

- Burglary and JohnCalls are conditionally independent given Alarm (case 1).
- JohnCalls and MaryCalls are conditionally independent given Alarm (case 2).
- Burglary and Earthquake are independent given no other information (case 3).

The Joint Distribution

By the product rule:

$$\mathbf{P}(B, E, A, J, M) = \mathbf{P}(J|B, E, A, M) \cdot \mathbf{P}(M|B, E, A) \cdot \mathbf{P}(A|B, E) \cdot \mathbf{P}(B|E) \cdot \mathbf{P}(E)$$

By conditional independence:

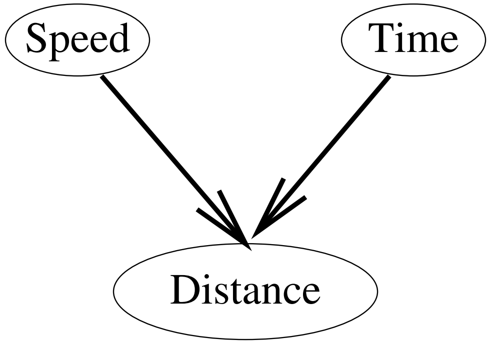
$$\mathbf{P}(B, E, A, J, M) = \mathbf{P}(J|A) \cdot \mathbf{P}(M|A) \cdot \mathbf{P}(A|B, E) \cdot \mathbf{P}(B) \cdot \mathbf{P}(E)$$

In general, the joint distribution is

$$\mathbf{P}(X_1, \ldots, X_n) = \prod_i^n \mathbf{P}(X_i|parents(X_i))$$

CPT Internal Structure: Deterministic Nodes

Sometimes there is structure *within* the CPT. A common case is a deterministic node. For example:

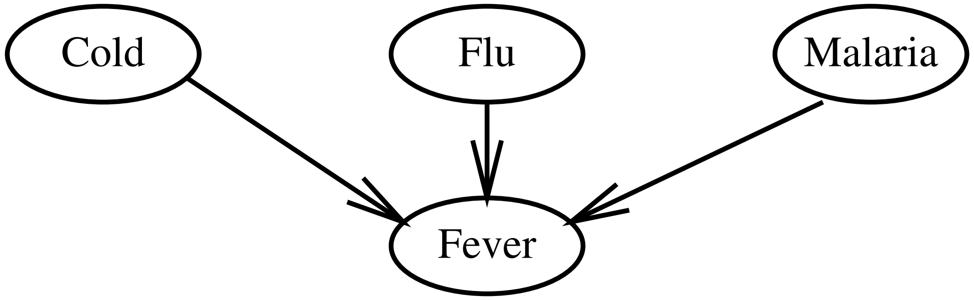


$$P(Distance|Rate, Time) = \begin{cases} 1.0 & \text{if } Distance = Rate \cdot Time \\ 0.0 & \text{otherwise} \end{cases}$$

Noisy-OR Table

Another common CPT structure is the Noisy-OR

- Each cause is independent of the others.
- All possible causes are listed.
- Whatever inhibits cold from causing fever is independent of whatever inhibits flu from causing fever.



Noisy-OR Table (2)

$$P(Fever) = 1 - \prod_i (1 - P_i)$$

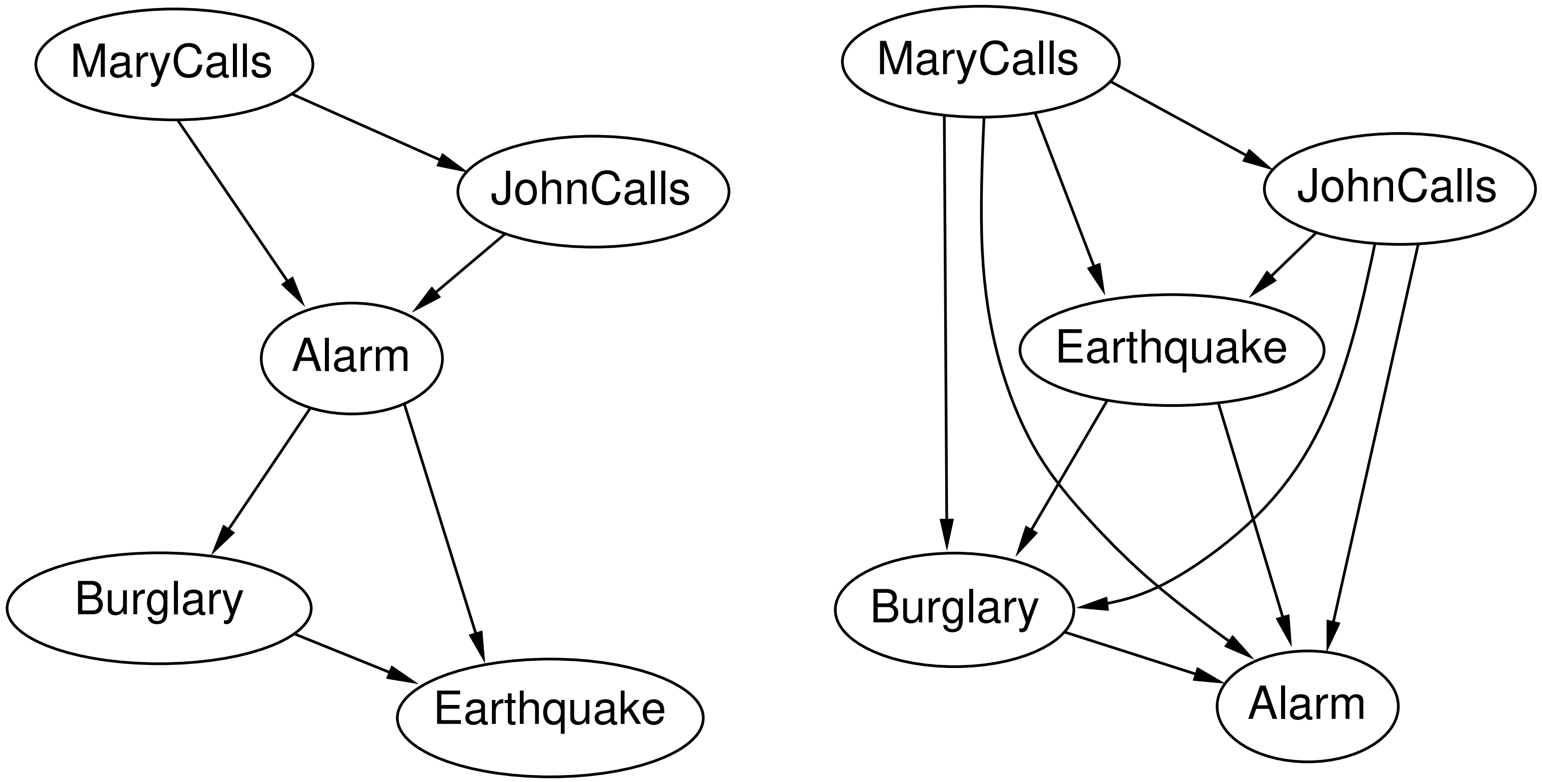
Cold	Flu	Malaria	$P(Fever)$	$P(\neg Fever)$
F	F	F	0.0	1.0
F	F	T	0.9	0.1
F	T	F	0.8	0.2
F	T	T	0.98	$0.02 = 0.2 \times 0.1$
T	F	F	0.4	0.6
T	F	T	0.94	$0.06 = 0.6 \times 0.1$
T	T	F	0.88	$0.12 = 0.6 \times 0.2$
T	T	T	0.988	$0.012 = 0.6 \times 0.2 \times 0.1$

Building a Network

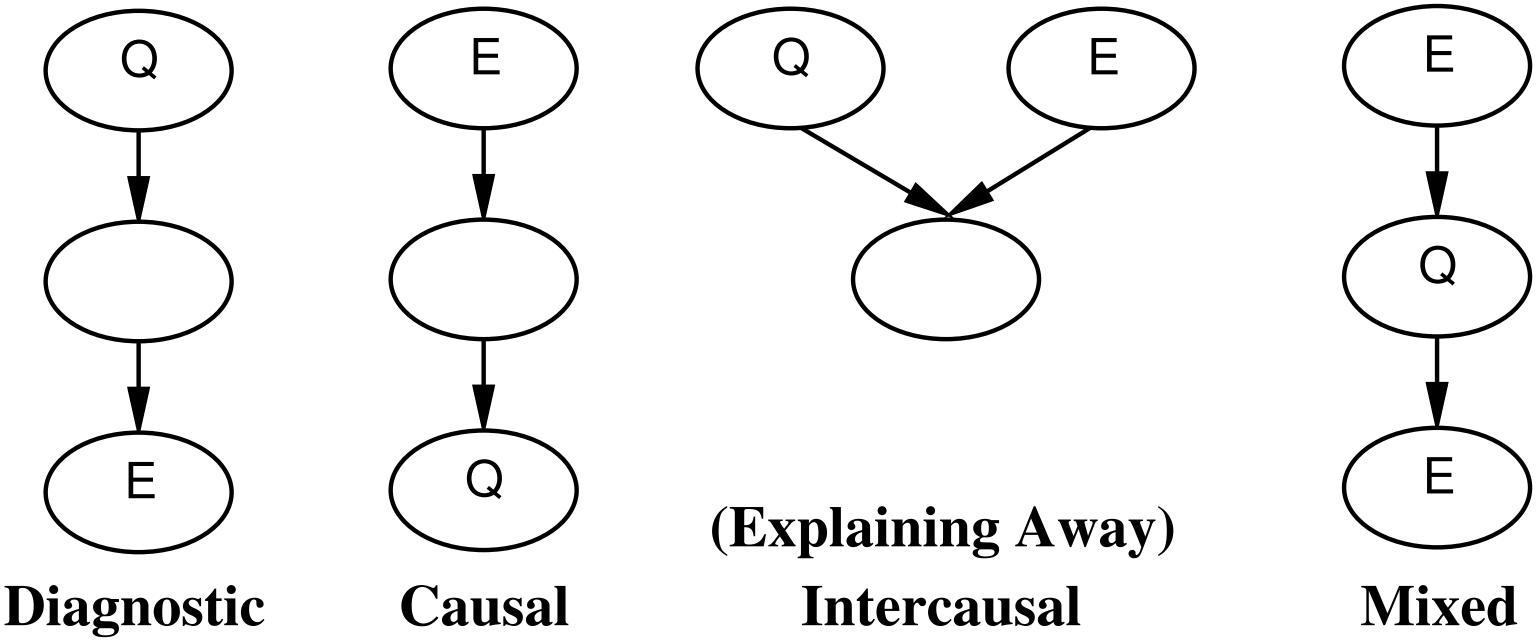
1. Choose the set of relevant variables X_i that describe the domain.
2. Choose an ordering for the variables such that each variable depends only on the variables preceding it.
3. While there are variables left:
   (a) Pick a variable X_i and add a node to the network for it.
   (b) Set $Parents(X_i)$ to be some minimal set of nodes already in the net such that $P(X_i|\text{all nodes in graph}) = P(X_i|Parents(X_i))$
   (c) Define the conditional probability table for X_i.

This guarantees that the axioms of probability are satisfied.

Network Structure Depends on the Order of Introduction



Types of Reasoning



- **Diagnostic:** $P(Burglary|JohnCalls, MaryCalls)$
- **Causal:** $P(JohnCalls|Burglary)$
- **Explaining Away:** $P(Burglary|Earthquake)$
- **Mixed:** $P(Alarm|Burglary, JohnCalls)$

Implementing Tell and Ask for Belief Nets

```
>(tell *bn* 'johncalls 1)
>(tell *bn* 'marycalls 1)
>(ask *bn* 'burglary)
(BURGLARY) Probability
0           0.7158
1           0.2842
```

Goal: Compute $\mathbf{P}(B|J, M)$

SPI Algorithm

$$
\begin{aligned}
\mathbf{P}(B|J,M) &= \frac{\mathbf{P}(B,J,M)}{\mathbf{P}(J,M)} \\
&= \frac{\sum_{E,A}\mathbf{P}(E,A,B,J,M)}{\mathbf{P}(J,M)} \\
&= \alpha \sum_{E,A} \mathbf{P}(E,A,B,J,M) \\
&= \alpha \sum_{E,A} \mathbf{P}(E)\cdot\mathbf{P}(B)\cdot\mathbf{P}(A|E,B)\cdot\mathbf{P}(J|A)\cdot\mathbf{P}(M|A) \\
&= \alpha \left[\sum_{A}\left[\sum_{E}\mathbf{P}(E)\cdot\mathbf{P}(A|E,B)\right]\cdot\mathbf{P}(J|A)\cdot\mathbf{P}(M|A)\right]\cdot\mathbf{P}(B)
\end{aligned}
$$

Let us look at a particular cell of $\mathbf{P}(B|J,M)$:

$$
P(B=0|J=1,M=0) = \alpha\left[\sum_{A}\left[\sum_{E}P(E)\cdot P(A|E,B=0)\right]\cdot P(J=1|A)\cdot P(M=0|A)\right]\cdot P(B=0)
$$

Conformal Products

Given two CPT's, the conformal product returns a new array called a **potential**. It cannot necessarily be interpreted as a CPT—it is just a term in the algebraic expression for the joint distribution.

$$P[C, B, A] = P[C, B] \cdot P[B, A]$$

C	B	$value$
0	0	0.4
0	1	0.8
1	0	0.6
1	1	0.2

$\cdot$

B	A	$value$
0	0	0.3
0	1	0.5
1	0	0.7
1	1	0.5

$=$

C	B	A	$value$
0	0	0	0.12
0	0	1	0.20
0	1	0	0.56
0	1	1	0.40
1	0	0	0.18
1	0	1	0.30
1	1	0	0.14
1	1	1	0.10

To fill cell 100 (CBA), take the product of cell 10 (CB) and cell 00 (BA).
When a potential is generated that contains *all* CPT's involving variable E, it is legal to marginalize out E.

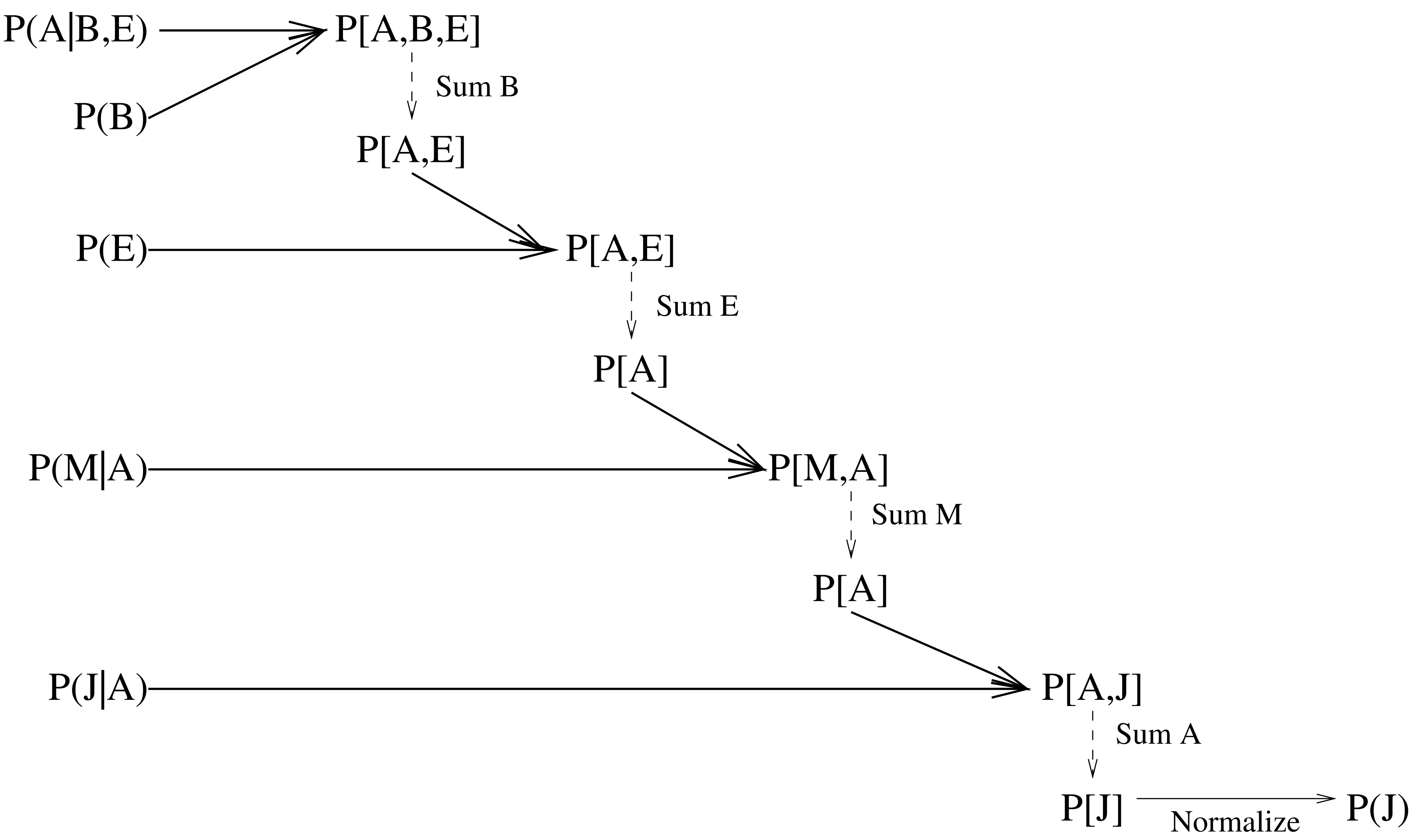
SPI Algorithm: Computing a marginal distribution

function SPI(Q)
let $O =$ all other variables
let $N =$ list of potentials (initially the list of CPTs)
while $(|N| > 1)$ **do**

    find the two elements A and $B \in N$ and corresponding
    legal marginalizers $O' \subset O$ such that
    $\Sigma_{O'} A \cdot B$ has smallest size
    delete A and B from N
    add $\Sigma_{O'} A \cdot B$ to N

end (while)
return first(N)
end SPI

SPI on the Alarm Network: $P(J)$



SPI Algorithm Example

Candidate Product	Sum over	Table Size
(burglary) * (earthquake)	nil	∞
(burglary) * (burglary earthquake alarm)	(burglary)	2
(burglary) * (alarm john-calls)	nil	∞
(burglary) * (alarm mary-calls)	nil	∞
(earthquake) * (burglary earthquake alarm)	(earthquake)	2
(earthquake) * (alarm john-calls)	nil	∞
(earthquake) * (alarm mary-calls)	nil	∞
(burglary earthquake alarm) * (alarm john-calls)	nil	4
(burglary earthquake alarm) * (alarm mary-calls)	(mary-calls)	3
(alarm john-calls) * (alarm mary-calls)	(mary-calls)	2

SPI Second Iteration

Candidate Product	Sum over	Table Size
(earthquake alarm) * (earthquake)	(earthquake)	1
(earthquake alarm) * (alarm john-calls)	nil	3
(earthquake alarm) * (alarm mary-calls)	(mary-calls)	2
(earthquake) * (alarm john-calls)	nil	∞
(earthquake) * (alarm mary-calls)	nil	∞
(alarm john-calls) * (alarm mary-calls)	(mary-calls)	2

SPI Final Iterations

Candidate Product	Sum over	Table Size
(alarm) * (alarm john-calls)	nil	2
(alarm) * (alarm mary-calls)	(mary-calls)	1
(alarm john-calls) * (alarm mary-calls)	(mary-calls)	2

(alarm) * (alarm john-calls)	(alarm)	1

SPI and `tell`

To implement `(tell *bn* var val)`:

For each CPT in the network that contains `var`

    delete all rows from the CPT where `var` $\neq$ `val`

    delete the `var` column from the CPT

    if a CPT has no variables, delete it from the network.

In `ask`, process as usual, except that the final CPT must be normalized before printing.

Probability Tables

The file `bayes-net.lisp` implements an abstract data type called a `ptable`.

```
(setq pt1 (make-ptable '((alarm earthquake burglary)
      .999 ; f f f
      .06  ; f f t
      .71  ; f t f
      .05  ; f t t
      .001 ; t f f
      .94  ; t f t
      .29  ; t t f
      .95  ; t t t
     ))
```

The order in which the probabilities are interpreted depends on the order of the variables on `*variables*`.

`(ptable-variables pt1)` returns `(alarm earthquake burglary)`

Operations on Probability Tables

- **Conformal Product** `(conformal-product pt1 pt2)`

  Returns a new probability table that is the conformal product of tables `pt1` and `pt2`.

- **Marginalizing** `(sum-over pt1 '(mary-calls earthquake))`

  Returns a new probability table where `mary-calls` and `earthquake` have been marginalized away.

- **Normalizing** `(normalize pt1 nil)`

  Modifies `pt1` so that all of its cells sum to 1. More generally, `(normalize pt1 '(mary-calls))` would normalize `pt1` so that all of the cells corresponding to each value of `mary-calls` would sum to 1.

- **Project** `(project pt1 'mary-calls 0)`

  Delete all rows from `pt1` for which `mary-calls` is not zero. Delete the `mary-calls` column from `pt1`.

Useful Lisp functions

- `(intersection '(a b d) '(b c d e)) => (b d)`
- `(union '(a b d) '(b c d e)) => (a b c d e)`
- `(set-difference '(a b c) '(b c d e)) => (a)`
- `most-positive-fixnum` The largest positive integer that can be stored in one word.