Course 440: Introduction To Artificial Intelligence
Lecture 8

Bayesian Networks

Abdeslam Boularias

Friday, October 28, 2016

Outline

We show how graphs can be used to efficiently represent joint probability distributions with a large number of variables.

1. Definition and examples
2. Independance assumptions in Bayesian networks
3. Exact inference
4. Approximate inference

Example problem : Lung cancer

A patient has been suffering from shortness of breath (called **dyspnoea**) and visits the doctor, worried that he has lung **cancer**. The doctor knows that other diseases, such as tuberculosis and bronchitis, are possible causes, as well as lung cancer. She also knows that other relevant information includes whether or not the patient is a **smoker** (increasing the chances of cancer and bronchitis) and what sort of **air pollution** he has been exposed to. A **positive X-ray** would indicate either TB or lung cancer.[a]

[a] Kevin B. Korb, Ann E. Nicholson. *Bayesian Artificial Intelligence*. CRC Press

Example problem : Lung cancer

A patient has been suffering from shortness of breath (called **dyspnoea**) and visits the doctor, worried that he has lung **cancer**. The doctor knows that other diseases, such as tuberculosis and bronchitis, are possible causes, as well as lung cancer. She also knows that other relevant information includes whether or not the patient is a **smoker** (increasing the chances of cancer and bronchitis) and what sort of **air pollution** he has been exposed to. A **positive X-ray** would indicate either TB or lung cancer.[a]

[a] Kevin B. Korb, Ann E. Nicholson. *Bayesian Artificial Intelligence*. CRC Press

Assume the patient has dyspnoea, is not a smoker, and has not been exposed to high pollution. If the x-ray comes positive, what is the probability that the patient has cancer?

- This problem is known as *medical diagnostic*: we have several possible causes of observed effects (symptoms).

Node name	Type	Values
Pollution	Binary	{*low, high*}
Smoker	Boolean	{*T, F*}
Cancer	Boolean	{*T, F*}
Dyspnoea	Boolean	{*T, F*}
X-ray	Binary	{*pos, neg*}

Kevin B. Korb, Ann E. Nicholson. *Bayesian Artificial Intelligence*. CRC Press

- Obviously, this problem cannot be solved using deterministic inference rules.
- We need to learn a probabilitic model, and to compute

$$P(cancer = true \mid dyspnoea = true, \\ X\text{-}ray = positive, \\ smoker = false, \\ pollution = low).$$

$P(cancer = t \mid dyspnoea = t, X\text{-}ray = pos, smoker = f, pollution = low)$ can be estimated from data as:

$$\frac{\text{number of cases } (cancer = t, dyspnoea = t, X\text{-}ray = pos, smoker = f, pollution = low)}{\text{number of cases } (dyspnoea = t, X\text{-}ray = pos, smoker = f, pollution = low)}$$

However, such an estimation will be hard to obtain accurately given the large number of parameters defining each instance.

$P(cancer = t \mid dyspnoea = t, X\text{-}ray = pos, smoker = f, pollution = low)$ can be estimated from data as:

$$\frac{\text{number of cases } (cancer = t, dyspnoea = t, X\text{-}ray = pos, smoker = f, pollution = low)}{\text{number of cases } (dyspnoea = t, X\text{-}ray = pos, smoker = f, pollution = low)}$$

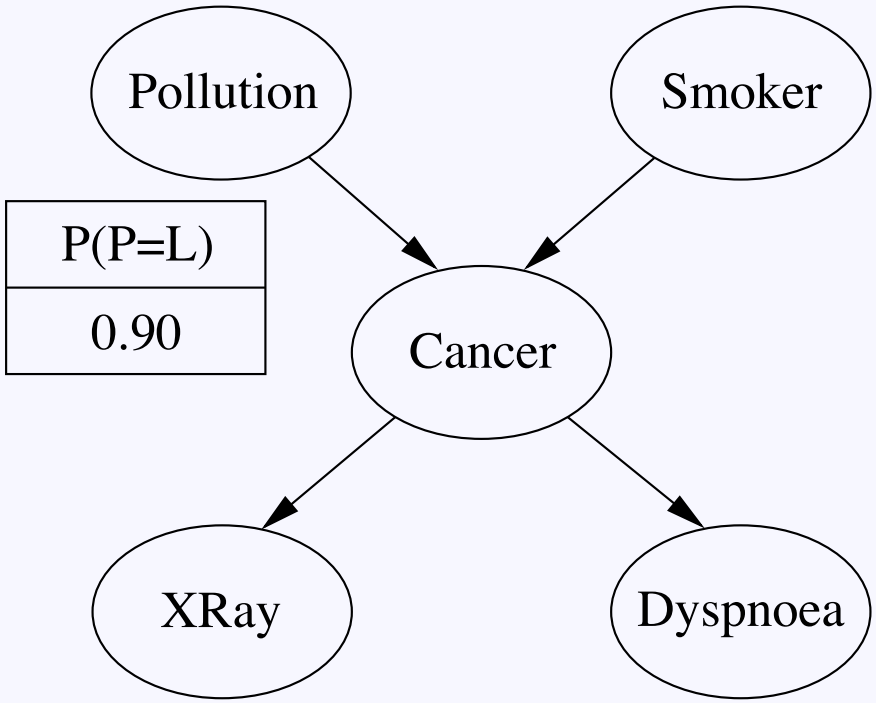
However, such an estimation will be hard to obtain accurately given the large number of parameters defining each instance.

- For example, assume there are no cases with non-smokers and low pollution, but a lot of cases with non-smokers and high pollution and a lot of cases with smokers and and low pollution.
- This should not be a problem if we know in advance that there is no strong causal relation between smoking and air pollution.

Representing joint probability distributions

- Instead of estimating $P(cancer \mid \text{everything else})$ (which is difficult), we model simpler relations such as $P(cancer \mid smoker, pollution)$ and $P(X\text{-}ray \mid cancer)$.
- These simpler relations are derived from the causal relations between different variables: smoking causes cancer, cancer causes positive X-rays.
- These simpler relations are easier to estimate from data.
- From the marginal probabilities of these simpler relations, one can compute the joint probability distribution $P(cancer = t \mid dyspnoea = t, X\text{-}ray = pos, smoker = f, pollution = low)$.
- The joint probability distribution can be used to compute other interesting probabilities, such as $P(Dyspnoea = pos \mid cancer)$.

Representing joint probability distributions



P(P=L)
0.90

P(S=T)
0.30

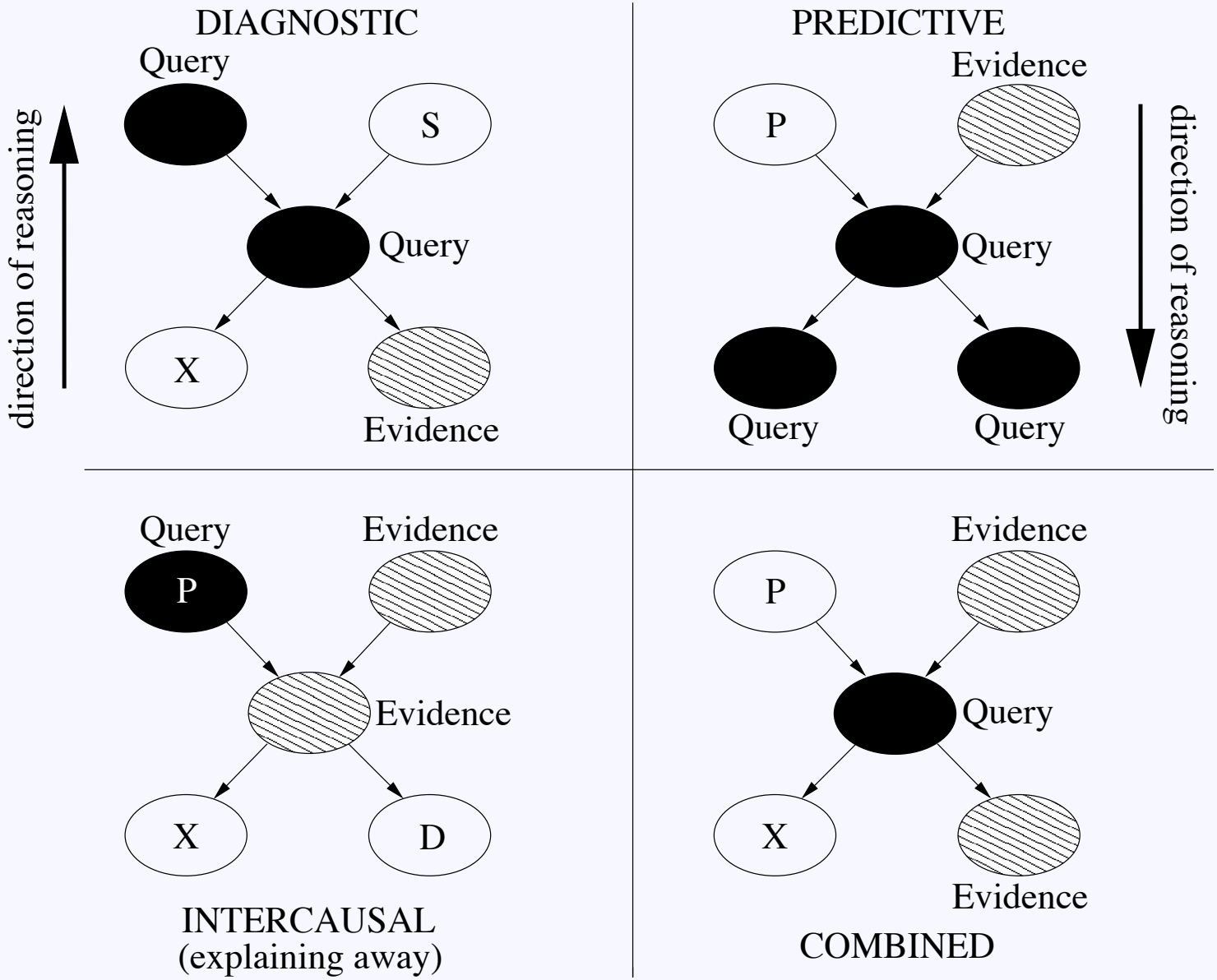
P	S	P(C=T|P,S)
H	T	0.05
H	F	0.02
L	T	0.03
L	F	0.001

C	P(X=pos|C)
T	0.90
F	0.20

C	P(D=T|C)
T	0.65
F	0.30

Kevin B. Korb, Ann E. Nicholson. *Bayesian Artificial Intelligence*. CRC Press

Types of reasoning with Bayesian Networks



Kevin B. Korb, Ann E. Nicholson, *Bayesian Artificial Intelligence*, CRC Press

Definition

A Bayesian network is a Directed Acyclic Graph (DAG) where

- Each node corresponds to a random variable.
- If there is an arrow from node X to node Y, X is said to be a parent of Y.
- Each node X_i has a conditional probability distribution $P(X_i \mid Parents(X_i))$ that quantifies the effect of the parents on the node.

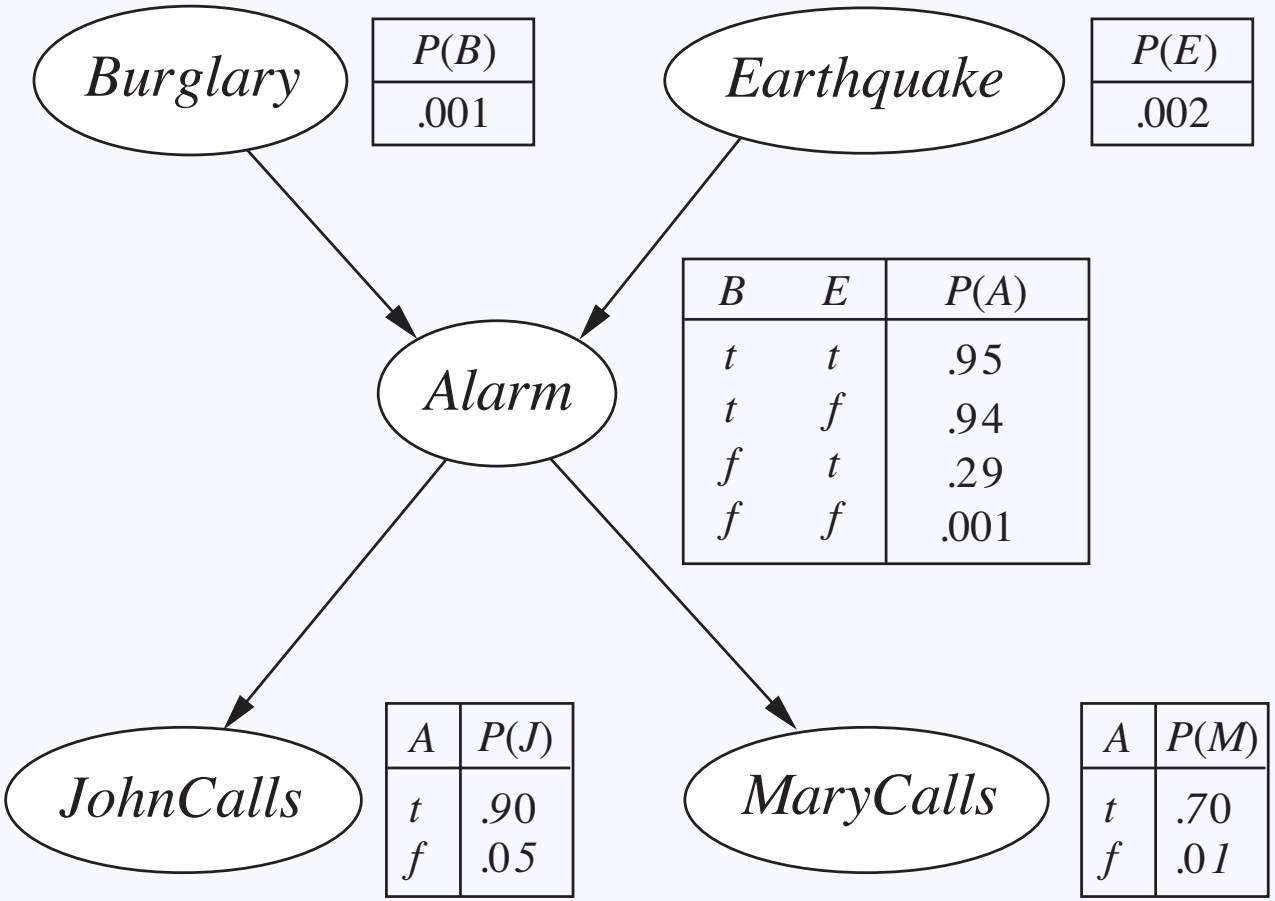
What is the meaning of the links?

The links in a Bayesian network reflect causal relations between variables.

Why do we use Bayesian networks?

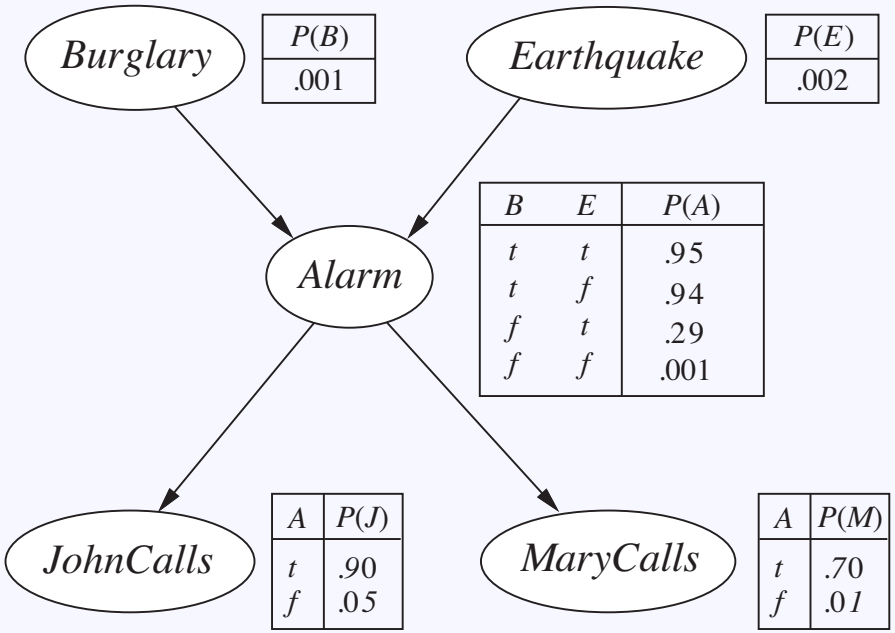
- The conditional probabilities represented in the links involve a smaller number of variables, which makes them easier to estimate and interpret.
- Inference is faster compared to when we use full joint distributions.
- Compact representation of full joint distributions.

Example of a Bayesian Network



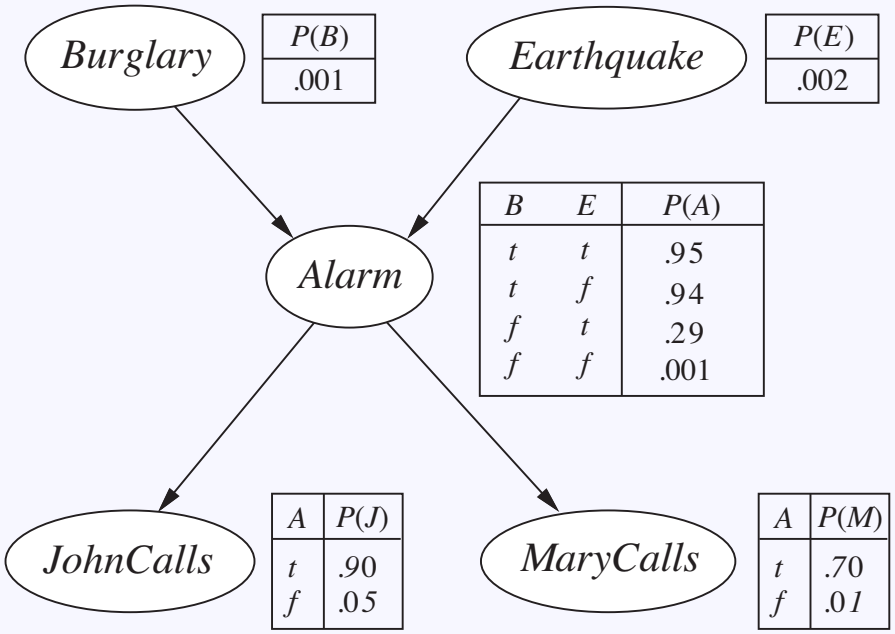
Bayesian network with conditional probability tables (CPTs).
B, E, A, J, and M stand for *Burglary, Earthquake, Alarm, JohnCalls*, and *MaryCalls* , respectively.

Example of a Bayesian Network



- Each row in a Conditional Probability Table (CPT) should sum to one.
- For Boolean variables, once you know that the probability of a true value is p, the probability of false must be $1 - p$. We omit the second value in the table.
- The CPT of a variable with k parents has 2^k rows, if the parents are Boolean.

Example of a Bayesian Network

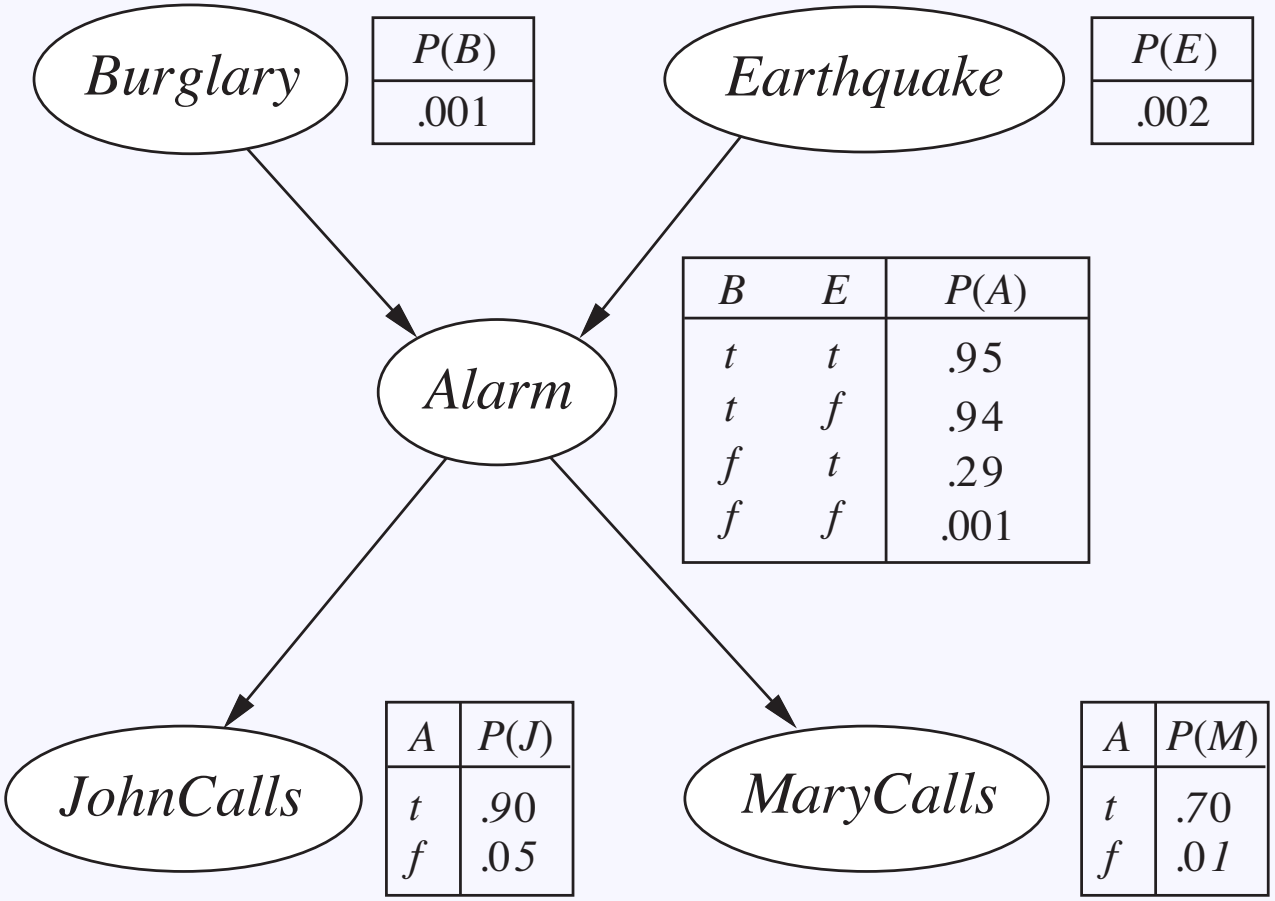


Assumptions:

- John and Mary do not communicate regarding the calls.
- John and Mary do not notice burglaries or minor earthquakes.

Notice that the network does not have nodes corresponding to Mary's currently listening to loud music or to the telephone ringing and confusing John. These factors are summarized in the uncertainty associated with the links from *Alarm*.

Example of a Bayesian Network



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

Representing the full joint distribution

Let $\{X_1, \ldots, X_n\}$ be the variables of a Bayesian network, and let $\theta(X_i \mid Parents(X_i))$ be the values in the corresponding CPTs. If values $\theta(X_i \mid Parents(X_i))$ are non-negative and each row in the CPTs sums to one, and

$$P(x_1, x_2 \ldots, x_n) = \prod_{i=1}^{n} \theta(x_i \mid parents(X_i)),$$

then $\theta(x_i \mid parents(X_i))$ are exactly the conditional probabilities $P(x_i \mid parents(X_i))$. Hence, we can write:

$$P(x_1, x_2 \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid parents(X_i)),$$

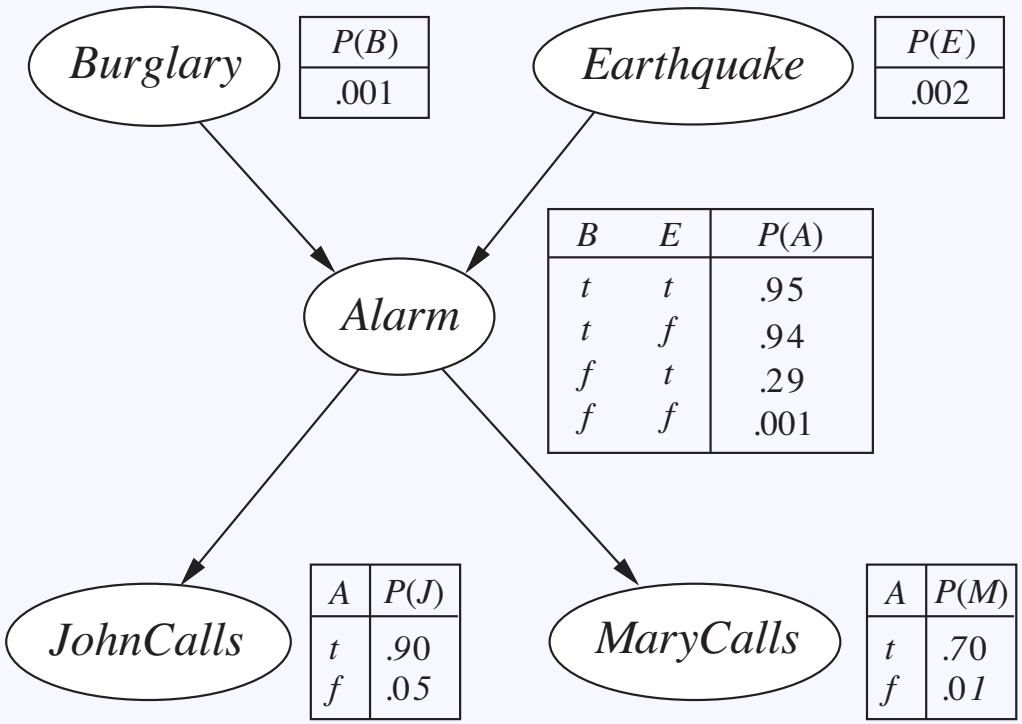
Representing the full joint distribution

The full joint distribution

$$P(x_1, x_2 \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid parents(X_i)),$$

can be used to answer any query about the domain.

Example



Probability that both John and Mary call, the alarm has sounded, but neither a burglary nor an earthquake has occurred:

$$\begin{aligned} P(j, m, a, \neg b, \neg e) &= P(j|a)P(m|a)P(a|\neg b \wedge \neg e)P(\neg b)P(\neg e) \\ &= 0.90 \times 0.70 \times 0.001 \times 0.999 \times 0.998 = 0.000628. \end{aligned}$$

Conditional Independence relations in Bayesian Networks

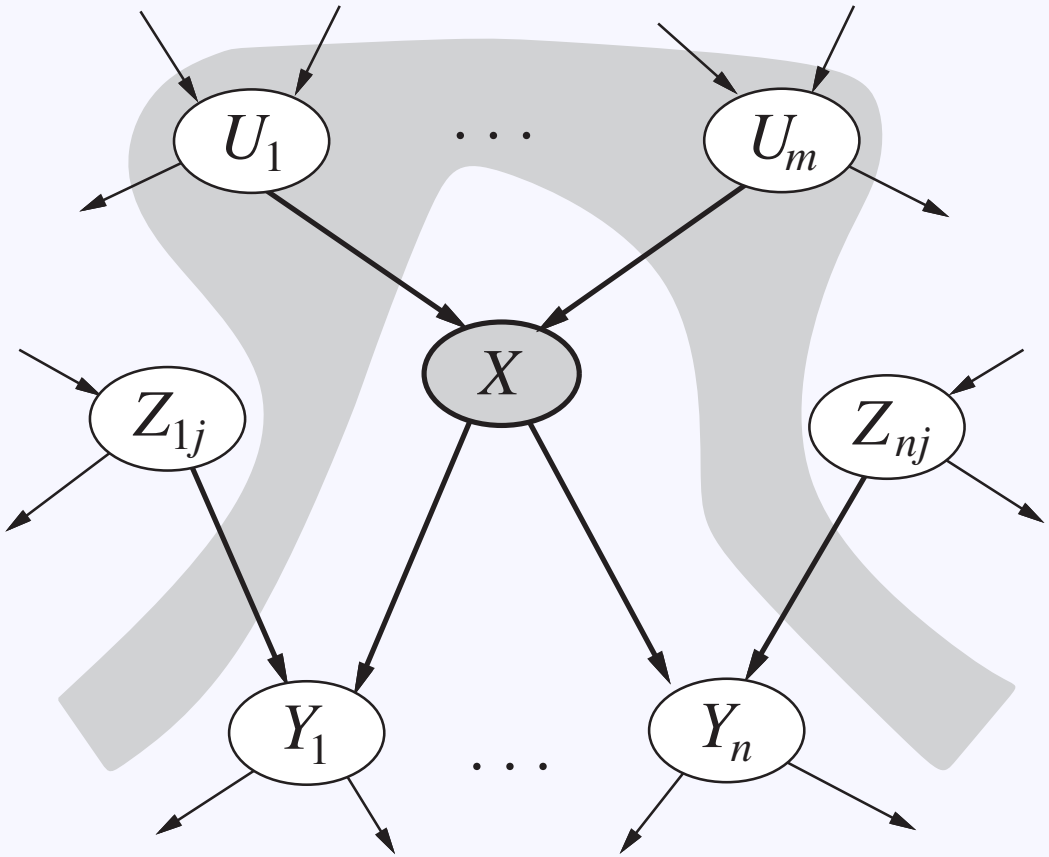
Given the definition of a Bayesian Network (directed acyclic graph + CPTs), i.e.

$$P(x_1, x_2 \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid parents(X_i)),$$

we can prove two types of independence:

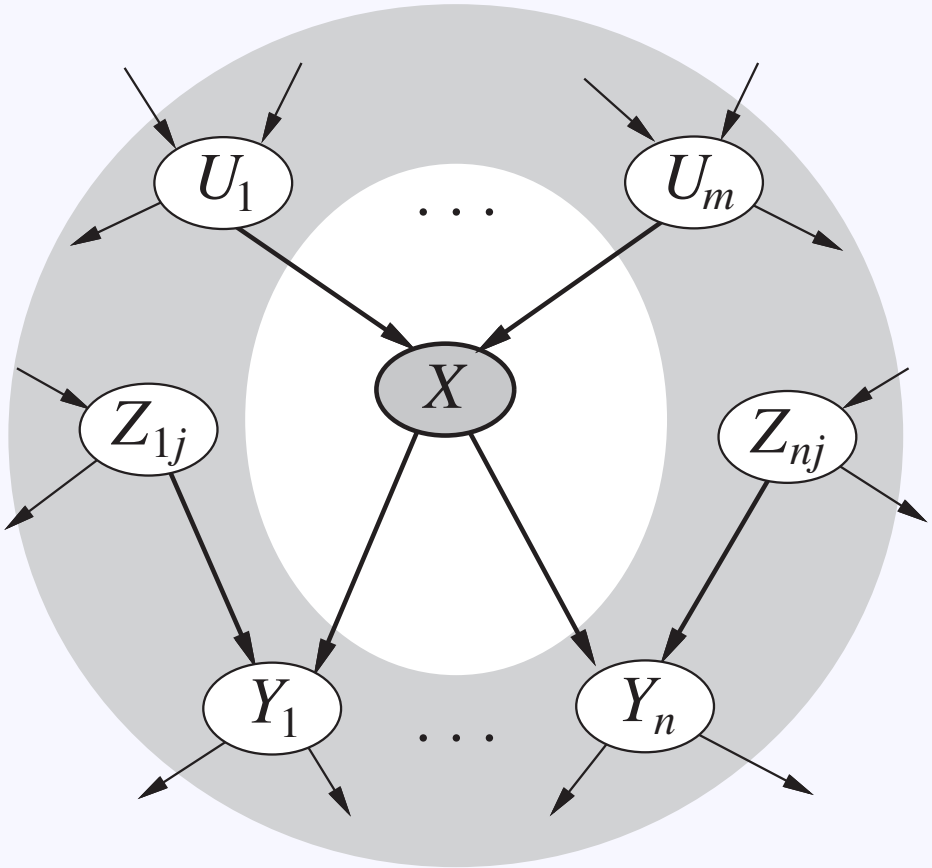
1. Each variable is conditionally independent of its non-descendants, given its parents.
2. A variable is conditionally independent of all other variables in the network, given its parents, children, and children's parents (Markov blanket).

Conditional Independence relations in Bayesian Networks



Each variable is conditionally independent of its non-descendants, given its parents.

Markov Blanket



A variable is conditionally independent of all other variables in the network, given its parents, children, and children's parents.

Inference in Bayesian Networks

- Inference in Bayesian Networks: find the probabilities of certain variables (**query variables**) given the values of some other variables (**evidence variables**, or **observed events**).
- Example: In the burglary network, we observe the events *JohnCalls* = *true* and *MaryCalls* = *true*. We ask then for the probability that a burglary has occurred:

$$P(Burglary \mid JohnCalls = true, MaryCalls = true) =< 0.284, 0.716 > .$$

- The remaining variables (nonevidence and nonquery variables) are called **hidden variables**.
- The hidden variables in the previous example are: *Alarm* and *Earthquake*.

We use the following notations

- X denotes the query variable.
- $\mathbf{E}$ denotes the set of evidence variables, and e is a particular observed event.
- $\mathbf{Y}$ denotes the set of hidden variables.
- $\mathbf{X} = \{X\} \cup \mathbf{E} \cup \mathbf{Y}$ denotes the complete set of variables.

A typical query asks for the posterior probability distribution $P(X \mid \mathbf{e})$.

Inference in Bayesian Networks

- Exact inference methods:
  - Inference by enumeration
  - Variable elimination algorithm
- Approximate inference methods:
  - Direct sampling
  - Rejection sampling
  - Likelihood weighting

Inference by enumeration

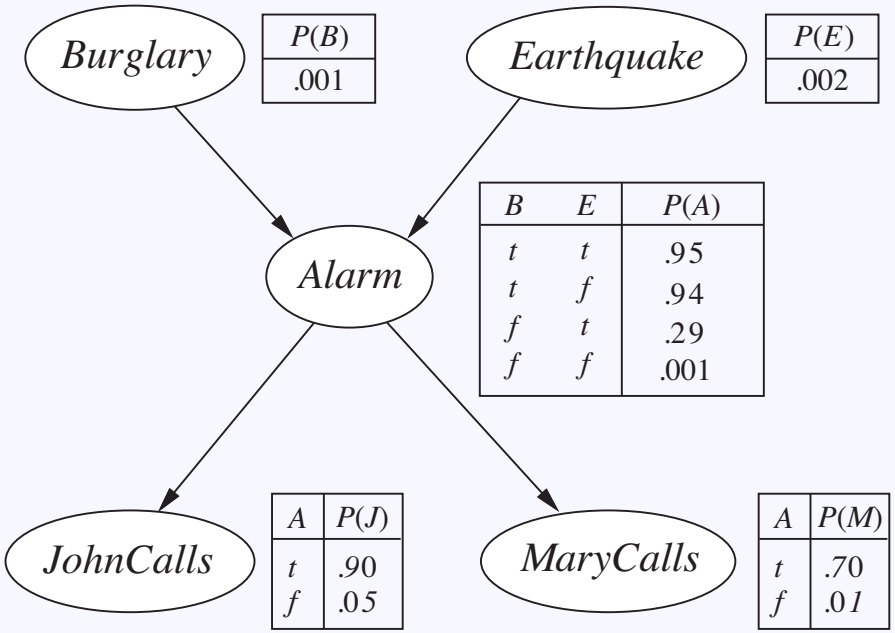
We use the joint probability distribution of all variables to answer the query

$$P(X \mid e) = \alpha P(X, e) = \alpha \sum_{y} P(X, e, y),$$

where α is a normalization constant (here, $\alpha = \frac{1}{\sum_x P(x,e)}$)

The full joint probability distribution is computed from the CPTs and topology of the Bayesian Network, by enumerating all the possible values of the hidden variables.

Inference by enumeration



Example: query $P(Burglary \mid JohnCalls = true, MaryCalls = true)$

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

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

Inference by enumeration

Example: query $P(Burglary \mid JohnCalls = true, MaryCalls = true)$

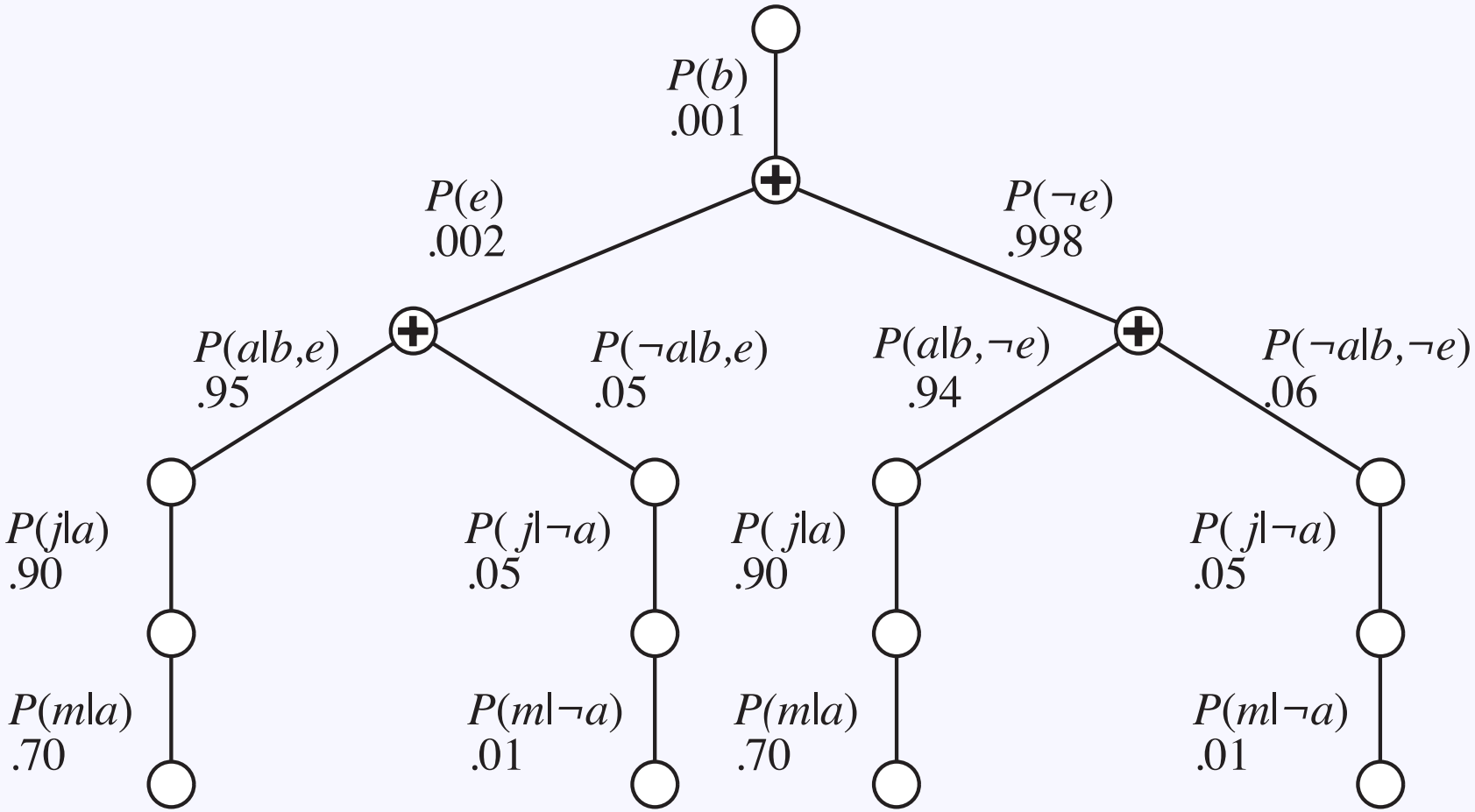
$$P(b \mid j, m) = \alpha \sum_e \sum_a P(b)P(e)P(a \mid b, e)P(j \mid a)P(m \mid a).$$

- This expression is computed by adding four terms, each computed by multiplying five numbers. Total $= 20$ operations
- Worst-case complexity: $\mathcal{O}(n2^n)$, where n is the number of variables.
- A simple re-arrangement of the terms can reduce the number of operations to 15.

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

Inference by enumeration

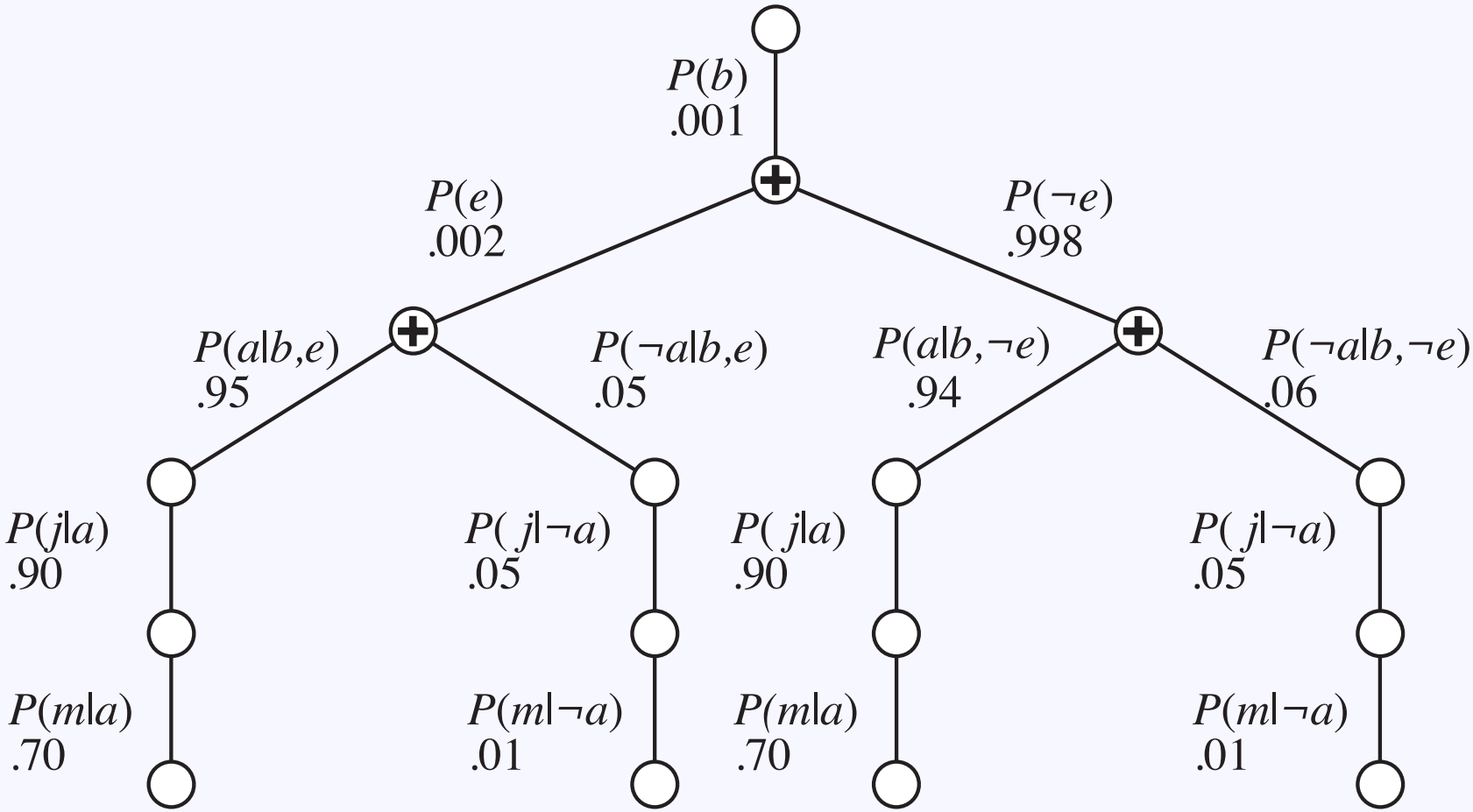
$$P(b \mid j, m) = \alpha P(b) \sum_e P(e) \sum_a P(a \mid b, e) P(j \mid a) P(m \mid a).$$



Enumeration Tree

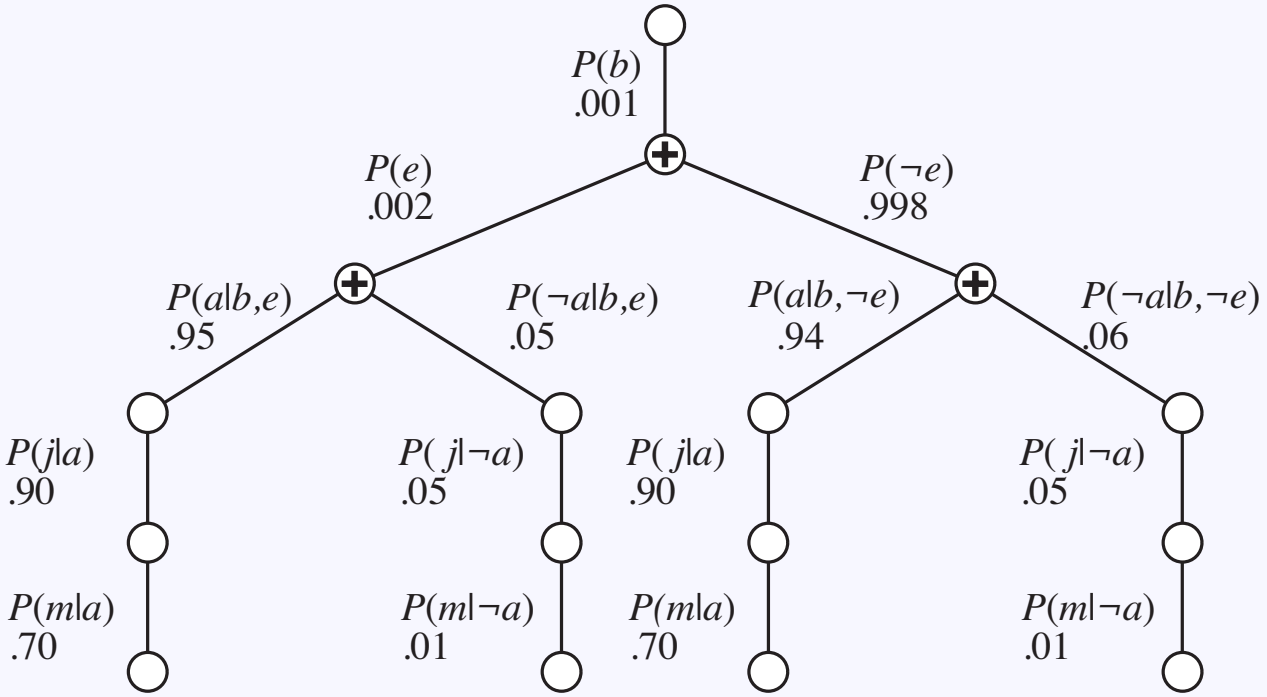
Inference by enumeration

$$P(B \mid j, m) = \alpha < 0.00059224, 0.0014919 > \approx < 0.284, 0.716 >$$



Enumeration Tree

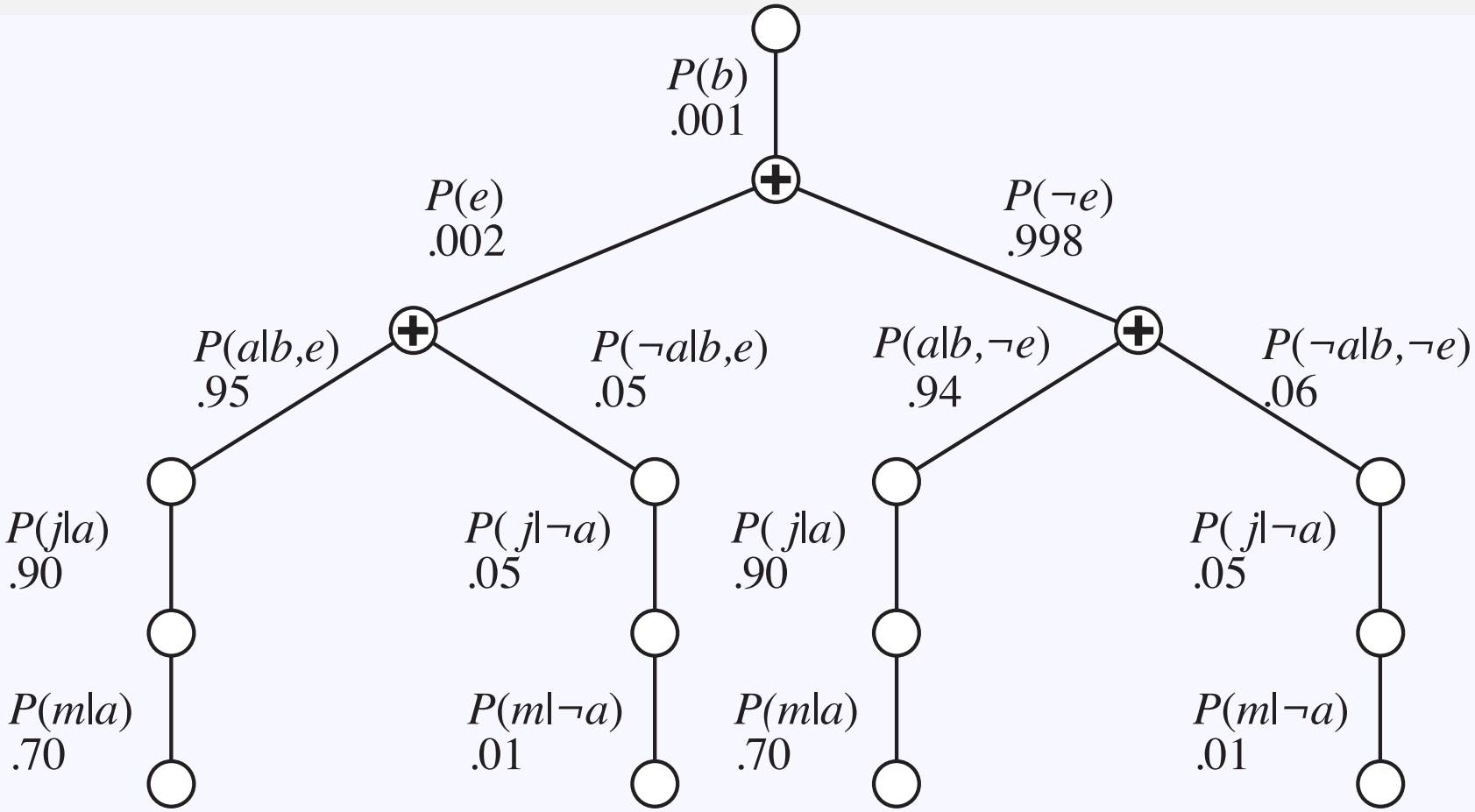
Inference by enumeration



Enumeration Tree

- Depth-First Search (DFS) can be used to parse the tree.
- Space complexity: $\mathcal{O}(n)$
- Time complexity: $\mathcal{O}(2^n)$, better than $\mathcal{O}(n2^n)$, but still high.

Variable elimination algorithm



- Notice how the products $P(j \mid a)P(m \mid a)$ and $P(j \mid \neg a)P(m \mid \neg a)$ are computed twice.
- Idea: do the calculation once and save the results for later use.

Variable elimination algorithm

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

- Each factor f_i is a matrix (tensor) indexed by the values of its argument variables.
- Example: Factors $f_4(A)$ and $f_5(A)$ corresponding to $P(j \mid a)$ and $P(m \mid a)$ depend just on A because J and M are fixed by the query.

$$f_4(A) = \begin{bmatrix} P(j \mid a) \\ P(j \mid \neg a) \end{bmatrix} = \begin{bmatrix} 0.90 \\ 0.05 \end{bmatrix} \qquad f_5(A) = \begin{bmatrix} P(m \mid a) \\ P(m \mid \neg a) \end{bmatrix} = \begin{bmatrix} 0.70 \\ 0.01 \end{bmatrix}$$

$$f_1(B) = \begin{bmatrix} P(b) \\ P(\neg b) \end{bmatrix} = \begin{bmatrix} 0.001 \\ 0.999 \end{bmatrix} \qquad f_2(E) = \begin{bmatrix} P(e) \\ P(\neg e) \end{bmatrix} = \begin{bmatrix} 0.002 \\ 0.998 \end{bmatrix}$$

Variable elimination algorithm

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

- Factor $f_3(A, B, E)$ depends on three variables, it corresponds to a $2 \times 2 \times 2$ matrix.

$$f_3(A, B, e) = \begin{bmatrix} P(a \mid b, e) & P(a \mid \neg b, e) \\ P(\neg a \mid b, e) & P(\neg a \mid \neg b, e) \end{bmatrix}$$

$$f_3(A, B, \neg e) = \begin{bmatrix} P(a \mid b, \neg e) & P(a \mid \neg b, \neg e) \\ P(\neg a \mid b, \neg e) & P(\neg a \mid \neg b, \neg e) \end{bmatrix}$$

Operations on factors

$$
\begin{aligned}
P(B \mid j, m) &= \alpha \underbrace{P(B)}_{f_1(B)} \sum_e \underbrace{P(e)}_{f_2(E)} \sum_a \underbrace{P(a \mid B, e)}_{f_3(A,B,E)} \underbrace{P(j \mid a)}_{f_4(A)} \underbrace{P(m \mid a)}_{f_5(A)}. \\
&= \alpha f_1(B) \times \sum_e f_2(E) \times \sum_a f_3(A, B, E) \times f_4(A) \times f_5(A).
\end{aligned}
$$

Pointwise Product $\times$

The pointwise product of two factors f_1 and f_2 yields a new factor f whose variables are the union of the variables in f_1 and f_2 and whose elements are given by the product of the corresponding elements in the two factors.

Operations on factors

A	B	$\mathbf{f}_1(A,B)$
T	T	.3
T	F	.7
F	T	.9
F	F	.1

B	C	$\mathbf{f}_2(B,C)$
T	T	.2
T	F	.8
F	T	.6
F	F	.4

A	B	C	$\mathbf{f}_3(A,B,C)$
T	T	T	$.3 \times .2 = .06$
T	T	F	$.3 \times .8 = .24$
T	F	T	$.7 \times .6 = .42$
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	$.1 \times .4 = .04$

$$f_3(A,B,C) = f_1(A,B) \times f_2(B,C)$$

Summing out

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

$$= \alpha f_1(B) \times \sum_e f_2(E) \times \sum_a f_3(A,B,E) \times f_4(A) \times f_5(A).$$

Summing out

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 A from $f_3(A,B,E)$, we write

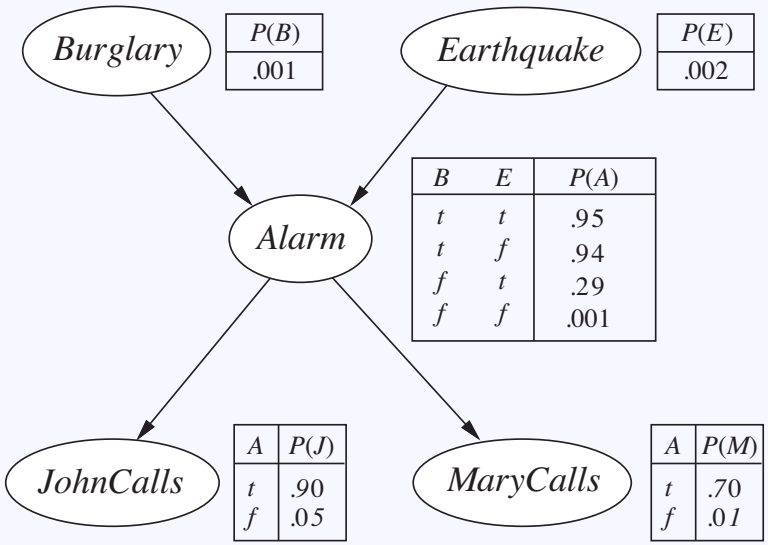
$$f(B,E) = \sum_a f_3(A,B,E) = f_3(a,B,E) + f_3(\neg a,B,E)$$

$$= \begin{bmatrix} .06 & .24 \\ .42 & .28 \end{bmatrix} + \begin{bmatrix} .18 & .72 \\ .06 & .04 \end{bmatrix} = \begin{bmatrix} .24 & .96 \\ .48 & .32 \end{bmatrix}$$

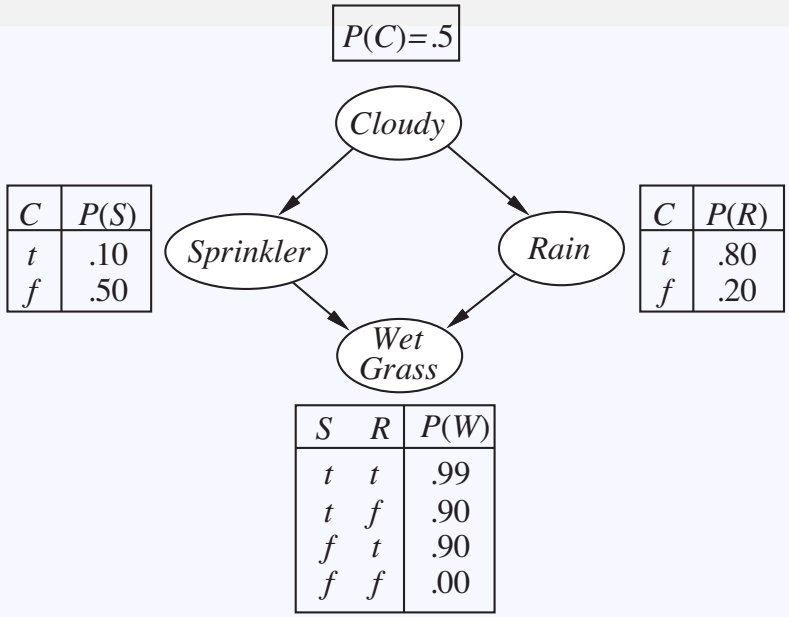
Variable elimination algorithm

The process of evaluation is a process of summing out variables (right to left) from pointwise products of factors to produce new factors, eventually yielding a factor that is the solution, i.e., the posterior distribution over the query variable.

Complexity of Exact Inference



Singly connected network



Multiply connected network

- Singly connected networks (or polytrees) are networks where there is at most one undirected path between any two nodes in the networks.
- The time and space complexity of exact inference in polytrees is **linear** in the number of CPT entries.
- For multiply connected networks, variable elimination can have **exponential** time and space complexity in the worst case.

Approximate Inference Methods

Since exact inference is intractable in large networks, we consider approximate inference methods that are much faster.

- Direct sampling
- Rejection sampling
- Likelihood weighting

What is sampling?

Sampling consists in generating a finite number of samples (values) from a known probability distribution.

Example

Sampling from a Bernoulli distribution $P(Coin) =< 0.5, 0.5 >$, where $Coin \in \{heads, tails\}$, consists in flipping a coin a number of times and observing the results, e.g. $\{heads, tails, tails, heads, tails, \dots\}$.

Why do we use sampling methods?

Sampling is often used to compute $\mathbb{E}[f(x)]$, where x is a random variable and $\mathbb{E}[f(x)]$ cannot be computed in a closed form (or efficiently).

Example

To compute $\mathbb{E}[\sqrt{|x|}]$ where $x \sim \mathcal{N}(0, 1)$ (standard normal distribution), we generate samples $\{0.0591, 1.7971, 0.2641, 0.8717, -1.4462\}$, and get $\mathbb{E}[\sqrt{x}] \approx \frac{\sqrt{|0.0591|}+\sqrt{|1.7971|}+\sqrt{|0.2641|}+\sqrt{|0.8717|}+\sqrt{|-1.4462|}}{5} \approx 0.85$

Direct sampling methods

Sample events from a network that has no evidence associated with it.

Each variable is sampled in turn, in topological order.

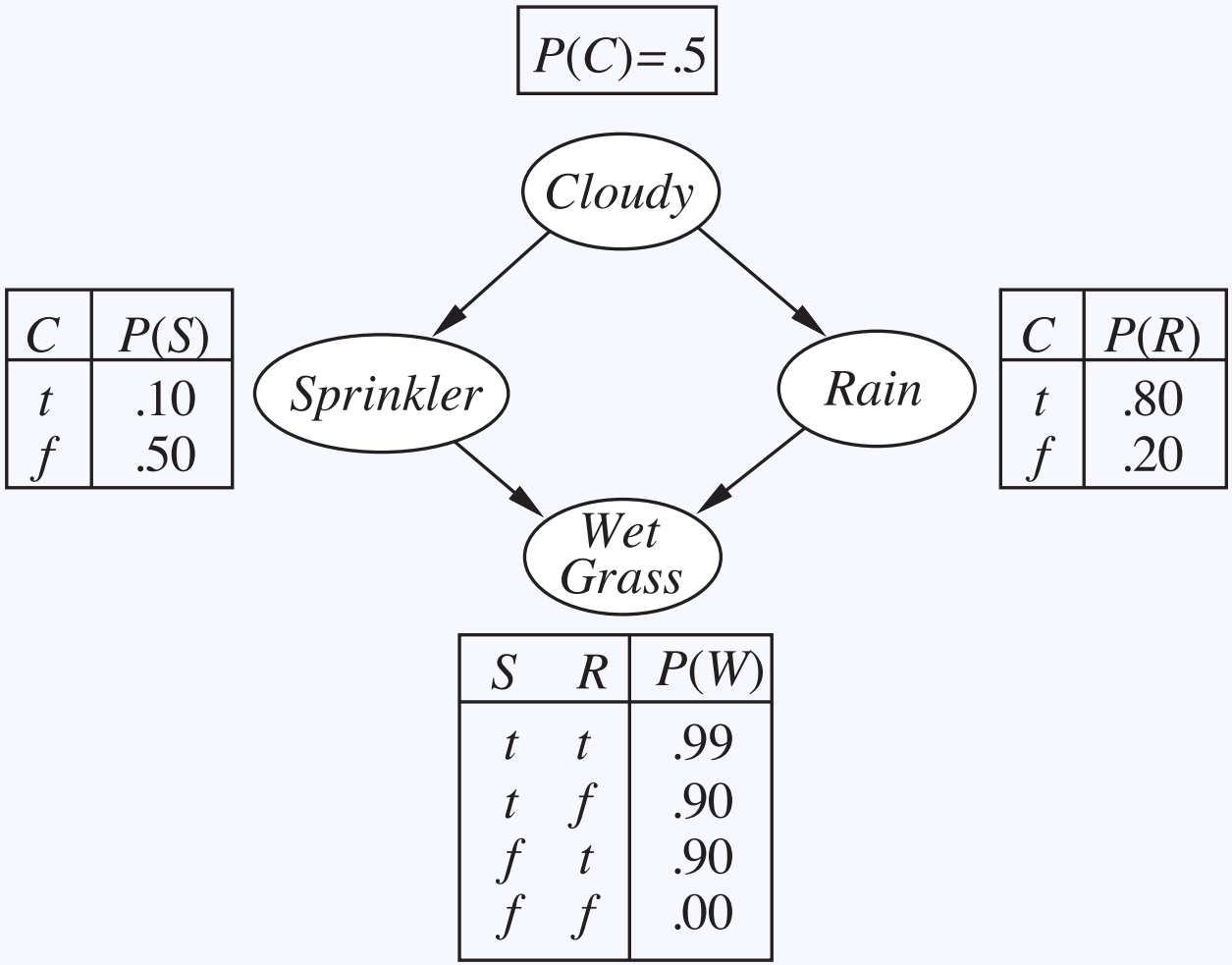
The probability distribution from which the value is sampled is conditioned on the values already assigned to the variable's parents.

function PRIOR-SAMPLE(bn) **returns** an event sampled from the prior specified by bn
inputs: bn, a Bayesian network specifying joint distribution $\mathbf{P}(X_1, \ldots, X_n)$

$\mathbf{x} \leftarrow$ an event with n elements
foreach variable X_i **in** $X_1, \ldots, X_n$ **do**
  $\mathbf{x}[i] \leftarrow$ a random sample from $\mathbf{P}(X_i \mid parents(X_i))$
return $\mathbf{x}$

Generate several samples $\mathbf{x}$ and calculate the frequency of each instance.

Example


P(C)=.5
Cloudy
Sprinkler
Rain
Wet Grass
C P(S)
t .10
f .50
C P(R)
t .80
f .20
S R P(W)
t t .99
t f .90
f t .90
f f .00

Example

1. Sample from $P(Cloudy) =< 0.5, 0.5 >$, value is true.
2. Sample from $P(Sprinkler \mid Cloudy = true) =< 0.1, 0.9 >$, value is false.
3. Sample from $P(Rain \mid Cloudy = true) =< 0.8, 0.2 >$, value is true.
4. Sample from $P(WetGrass \mid Sprinkler = false, Rain = true) =< 0.9, 0.1 >$, value is true.

Rejection sampling

Direct sampling is useful for estimating a joint probability $P(x_1, x_2, \ldots, x_n)$ when there is no evidence (no known value for any variable).

The same idea can be used to estimate $P(\mathbf{X} \mid \mathbf{e})$, where $\mathbf{X}$ is any variable and $\mathbf{e}$ is an evidence (the value(s) of certain variable(s)).

1. Generate samples from the prior distribution specified by the network.
2. Reject all those that do not match the evidence.
3. Estimate $\hat{P}(x| \mathbf{e})$ is obtained by counting the number of samples where $\mathbf{X} = x$.

$$\hat{P}(x| \mathbf{e}) = \frac{\text{number of samples } (x, \mathbf{e})}{\text{number of samples } (\mathbf{e})}$$

Example of rejection sampling

- We wish to estimate $P(Rain \mid Sprinkler = true)$, using 100 samples.
- Of the 100 samples,
  - 73 have $Sprinkler = false$ and are rejected,
  - 27 have $Sprinkler = true$. Of the 27,
    - 8 have $Rain = true$,
    - and 19 have $Rain = false$.
- Thus,

$$P(Rain|Sprinkler = true) \approx normalize(< 8, 19 >) =< 0.296, 0.704 > .$$

Rejection sampling

function REJECTION-SAMPLING(X, **e**, bn, N) **returns** an estimate of $\mathbf{P}(X|\mathbf{e})$
  inputs: X, the query variable
    e, observed values for variables **E**
    bn, a Bayesian network
    N, the total number of samples to be generated
  local variables: **N**, a vector of counts for each value of X, initially zero

  for $j = 1$ to N **do**
    x $\leftarrow$ PRIOR-SAMPLE(bn)
    if **x** is consistent with **e** **then**
      N[x] $\leftarrow$ **N**[x]+1 where x is the value of X in **x**
  return NORMALIZE(**N**)

Likelihood weighting

Rejection sampling is not efficient because it wastes a lot of samples (all the samples that do not agree with the provided evidence).

Can we simply force the evidence variables to agree with the provided values, and sample only the non-evidence variables?

Likelihood weighting

- Assume we have a simple Bayesian Network: $\boxed{Smoking} \rightarrow \boxed{Cancer}$.
- Assume $P(Smoking) = 0.3$, $P(Cancer \mid Smoking) = 0.2$ and $P(Cancer \mid \neg Smoking) = 0.01$.

- Evidence: $Cancer = true$
- Inquiry: $P(Smoking = true \mid Cancer = true)$

- If we use Rejection Sampling while forcing $Cancer = true$, we will sample only $Smoking$.
- Notice that $Smoking$ has no parent, it will be sampled as true with probability 0.3.
- Therefore: $\frac{\text{number of samples } (Smoking, Cancer)}{\text{number of samples } (Cancer)} \approx 0.3$

Likelihood weighting

Whereas:

$$P(Smoking \mid Cancer) = \frac{P(Cancer \mid Smoking)P(Smoking)}{P(Cancer)}$$
$$= \frac{P(Cancer \mid Smoking)P(Smoking)}{P(Cancer \mid Smoking)P(Smoking) + P(Cancer \mid \neg Smoking)P(\neg Smoking)}$$
$$= \frac{0.2 \times 0.3}{0.2 \times 0.3 + 0.01 \times 0.7} \approx 0.9$$

Likelihood weighting is a special case of methods known as *Importance Sampling*. It consists in weighting each sample by its likelihood.
$\hat{P}(Smoking \mid Cancer) = \frac{\hat{P}(Cancer, Smoking)}{\hat{P}(Cancer)}$, where both $\hat{P}(Cancer, Smoking)$ and $\hat{P}(Cancer)$ are estimated from the samples.

Likelihood weighting

function LIKELIHOOD-WEIGHTING(X, **e**, bn, N) **returns** an estimate of **P**($X|$**e**)

inputs: X, the query variable
e, observed values for variables **E**
bn, a Bayesian network specifying joint distribution **P**($X_1, \ldots, X_n$)
N, the total number of samples to be generated

local variables: **W**, a vector of weighted counts for each value of X, initially zero

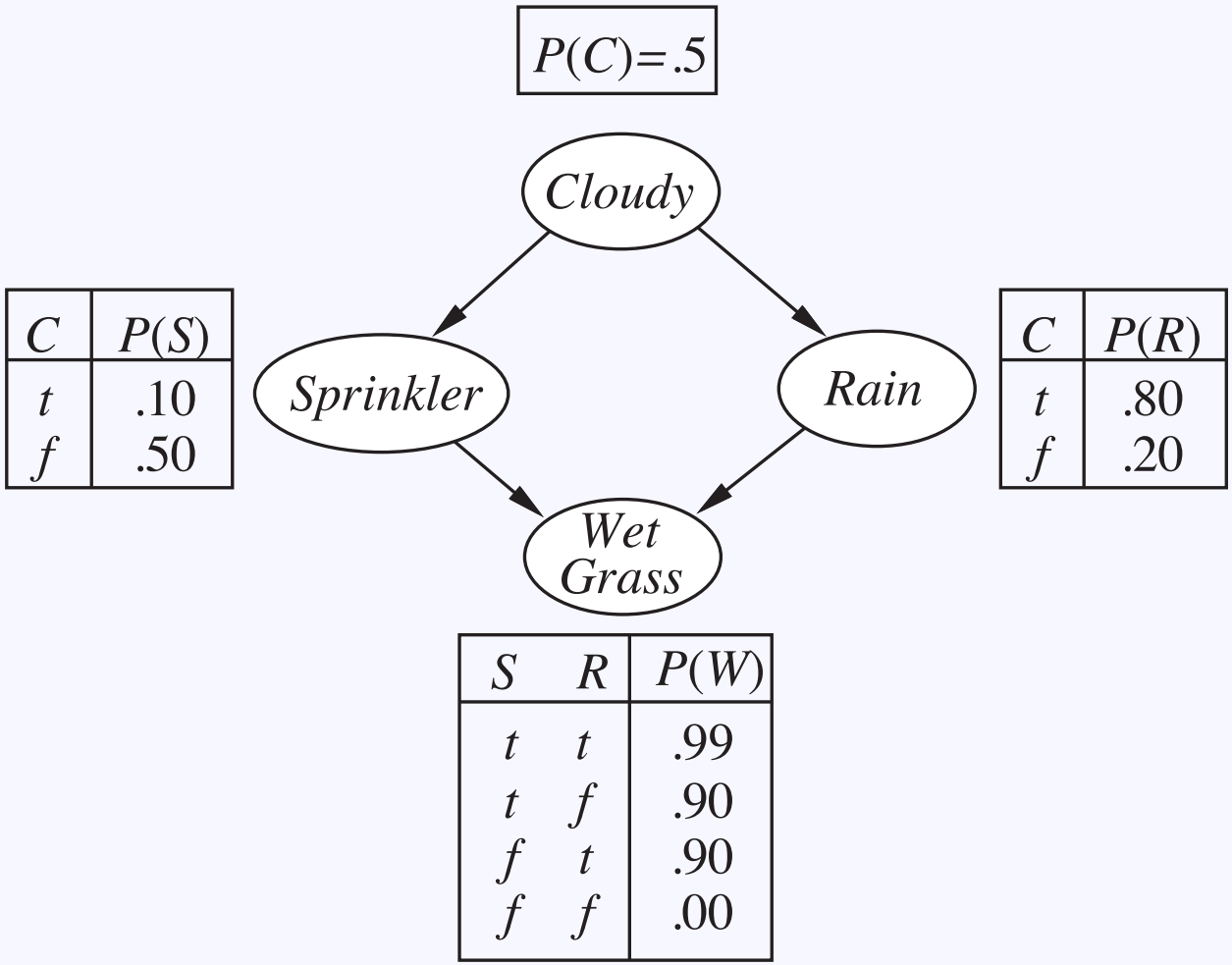
for j = 1 to N **do**
x, $w \leftarrow$ WEIGHTED-SAMPLE(bn, **e**)
W[x] $\leftarrow$ **W**[x] + w where x is the value of X in **x**
return NORMALIZE(**W**)

---

function WEIGHTED-SAMPLE(bn, **e**) **returns** an event and a weight

$w \leftarrow 1$; **x** $\leftarrow$ an event with n elements initialized from **e**
foreach variable X_i **in** $X_1, \ldots, X_n$ **do**
if X_i is an evidence variable with value x_i in **e**
then $w \leftarrow w \times P(X_i = x_i \mid parents(X_i))$
else **x**[i] $\leftarrow$ a random sample from **P**($X_i \mid parents(X_i)$)
return **x**, w

Example


P(C)=.5
Cloudy
Sprinkler
Rain
Wet Grass
C P(S)
t .10
f .50
C P(R)
t .80
f .20
S R P(W)
t t .99
t f .90
f t .90
f f .00

Example of likelihood weighting

Computing $P(Rain \mid Cloudy = true, WetGrass = true)$

Initially, the weight w is set to 1.

1. *Cloudy* is an evidence variable, with value *true*. Therefore, we set $w \leftarrow w \times P(Cloudy = true) = 1 \times 0.5$
2. *Sprinkler* is not an evidence variable, so sample from $P(Sprinkler \mid Cloudy = true) = [0.1, 0.9]$. Suppose the sampled value is *false*.
3. *Rain* is not an evidence variable, so sample from $P(Rain \mid Cloudy = true) = [0.8, 0.2]$. Suppose the sampled value is *true*.
4. *WetGrass* is an evidence variable, with value *true*. Therefore, we set

$$w \leftarrow w \times P(WetGrass = true \mid Sprinkler = false, Rain = true) = 0.5 \times 0.9$$

WEIGHTED-SAMPLE returns the event $[Cloudy = true, Sprinkler = false, Rain = true, WetGrass = true]$ with weight 0.45

Example of likelihood weighting

Computing $P(Rain \mid Cloudy = true, WetGrass = true)$

- Vector $\mathbf{W}$ is indexed by the values of the inquiry variable ($Rain \in \{true, false\}$).
- Therefore, vector $\mathbf{W}$ has two entries: $W[true]$ for $Rain = true$ and $W[false]$ for $Rain = false$.
- $\mathbf{W}$ is initialized to $[0, 0]$.
- Since the sampled event is $[Cloudy = true, Sprinkler = false, \mathbf{Rain = true}, WetGrass = true]$ with weight 0.45, then $\mathbf{W}[true] \leftarrow W[true] + 0.45$.

Suppose after several samples and updates of $\mathbf{W}$, we have $\mathbf{W} = [580, 1420]$.
Then, $P(Rain \mid Cloudy = true, WetGrass = true) \approx [\frac{\mathbf{W}[true]}{\mathbf{W}[true]+\mathbf{W}[false]}, \frac{\mathbf{W}[false]}{\mathbf{W}[true]+\mathbf{W}[false]}] = [\frac{580}{580+1420}, \frac{1420}{580+1420}] = [0.29, 0.71]$

Why does likelihood weighting work?

- Let $[z, e]$ be a sample returned by the WEIGHTED-SAMPLE procedure, where e is a vector of evidence values, and z is a vector of sampled values (for non-evidence variables).
- Let $S(z, e)$ denote the probability of a sample $[z, e]$, we have then

$$S(z, e) = \prod_{i=1}^{l} P(z_i \mid parents(Z_i))$$

- Weight $w(z, e)$ of sample $[z, e]$ is given by

$$w(z, e) = \prod_{i=1}^{m} P(e_i \mid parents(E_i))$$

Why does likelihood weighting work?

The LIKELIHOOD-WEIGHTING procedure returns an estimated probability $\hat{P}(x \mid e)$ given as

$$\hat{P}(x \mid e) = \frac{\sum_y N(x, y, e) w(x, y, e)}{\sum_x \sum_y N(x, y, e) w(x, y, e)},$$

where y are non-evidence non-inquiry variables, and $N(x, y, e)$ is the number of times $[x, y, e]$ has been sampled.

For a large number of samples, we have

$$\hat{P}(x \mid e) \approx \frac{\sum_y S(x, y, e) w(x, y, e)}{\sum_x \sum_y S(x, y, e) w(x, y, e)}$$

Why does likelihood weighting work?

For a large number of samples, we have

$$\begin{aligned}\hat{P}(x \mid e) &\approx \frac{\sum_y S(x,y,e)w(x,y,e)}{\sum_x \sum_y S(x,y,e)w(x,y,e)} \\ &= \frac{\sum_y \prod_{i=1}^{l} P(z_i \mid parents(Z_i)) \prod_{i=1}^{m} P(e_i \mid parents(E_i))}{\sum_x \sum_y \prod_{i=1}^{l} P(z_i \mid parents(Z_i)) \prod_{i=1}^{m} P(e_i \mid parents(E_i))},\end{aligned}$$

where Z_i are non-evidence variables ($\in X \cup Y$). $parents(Z_i)$ and $parents(E_i)$ may contain X or Y.

Therefore,

$$\begin{aligned}\hat{P}(x \mid e) &\approx \frac{\sum_y P(x,y,e)}{\sum_x \sum_y P(x,y,e)} \\ \hat{P}(x \mid e) &\approx \frac{P(x,e)}{P(e)} = P(x \mid e)\end{aligned}$$

Markov Chain Monte Carlo (MCMC)

- Instead of generating each sample from scratch, MCMC algorithms generate each sample by making a random change to the preceding sample.
- An assignment of values to all variables is a *state*.
- MCMC jumps from the current state to a next state.
- After a large number of moves, the frequency of each value of the inquiry variable X converges to its true distribution $P(X \mid e)$.

Markov Chain Monte Carlo (MCMC)

Fix the evidence variables. Initialize all the remaining variables with random values.

1. For each non-evidence variable Z_i, sample a new value from $P(Z_i \mid mb(Z_i))$ where $mb(Z_i)$ is the Markov blanket of Z_i (parents, children, and children's parents).
2. Add the current state to the collection of samples, and repeat step 1.

Gibbs sampling

function GIBBS-ASK(X, **e**, bn, N) **returns** an estimate of $\mathbf{P}(X|\mathbf{e})$
  local variables: **N**, a vector of counts for each value of X, initially zero
    Z, the nonevidence variables in bn
    x, the current state of the network, initially copied from **e**

  initialize **x** with random values for the variables in **Z**
  for $j = 1$ to N **do**
    for each Z_i in **Z do**
      set the value of Z_i in **x** by sampling from $\mathbf{P}(Z_i|mb(Z_i))$
      $\mathbf{N}[x] \leftarrow \mathbf{N}[x] + 1$ where x is the value of X in **x**
  return NORMALIZE(**N**)

Example

Computing $P(Rain \mid Sprinkler = true, WetGrass = true)$

Initialization:

1. We fix $Sprinkler = true$ and $WetGrass = true$.
2. We assign random values to $Cloudy$ and $Rain$. Suppose these random values are $true$ and $false$ respectively. The initial state is then $[Cloudy = true, Sprinkler = true, Rain = false, WetGrass = true]$.

Example

Computing $P(Rain \mid Sprinkler = true, WetGrass = true)$

First iteration:

1. We fix $Sprinkler = true$ and $WetGrass = true$.
2. $Cloudy$ is sampled, given the current values of its Markov blanket variables. We sample from $P(Cloudy \mid Sprinkler = true, Rain = false)$. Suppose we get $Cloudy = false$, then the current state is $[Cloudy = false, Sprinkler = true, Rain = false, WetGrass = true]$.
3. $Rain$ is sampled, given the current values of its Markov blanket variables. We sample from $P(Rain \mid Cloudy = false, Sprinkler = true, WetGrass = true)$. Suppose we get $Rain = true$, then the current state is $[Cloudy = false, Sprinkler = true, Rain = true, WetGrass = true]$.

Suppose we repeat this process for 100 iterations, and $Rain = true$ in 20 visited states and $Rain = false$ in 80 states, then $P(Rain \mid Sprinkler = true, WetGrass = true) = [0.2, 0.8]$.