

Outline

- Introduction
- Bayesian Probability Theory
- Sequence Prediction and Data Compression
- Decision Trees and Ensemble Learning
- Bayesian Networks
 - Syntax and semantics
 - Inference algorithms

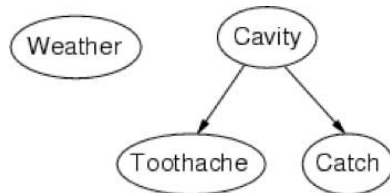
Bayesian Networks

- A graphical notation for capturing large distributions
- Exploits (conditional) independence between variables to achieve compact representations
- Syntax
 - a set of nodes, one per variable
 - a directed, acyclic graph (link \approx influences)
 - a conditional distribution for each node x_i given its parents

$$\Pr(x_i | \text{parents}(x_i))$$

Conditional Independence

- Topology of network encodes conditional independence assertions.



E.g.

- $\Pr(\text{Weather} | \text{Cavity}, \text{Toothache}, \text{Catch}) = \Pr(\text{Weather})$
- $\Pr(\text{Toothache} | \text{Cavity}, \text{Catch}) = \Pr(\text{Toothache} | \text{Cavity})$
- $\Pr(\text{Catch} | \text{Cavity}, \text{Toothache}, \text{Weather}) = \Pr(\text{Catch} | \text{Cavity})$

Semantics of Bayesian Networks

A Bayesian network with nodes $V = \{x_1, x_2, \dots, x_n\}$ defines a distribution over V as follows:

$$\Pr(x_1, x_2, \dots, x_n) = \prod_{i=1}^n \Pr(x_i | \text{parents}(x_i)).$$

In general, the specification of $\Pr(x_1, x_2, \dots, x_n)$ for n boolean variables requires 2^n numbers.

Taking independence into account can yield more compact descriptions.

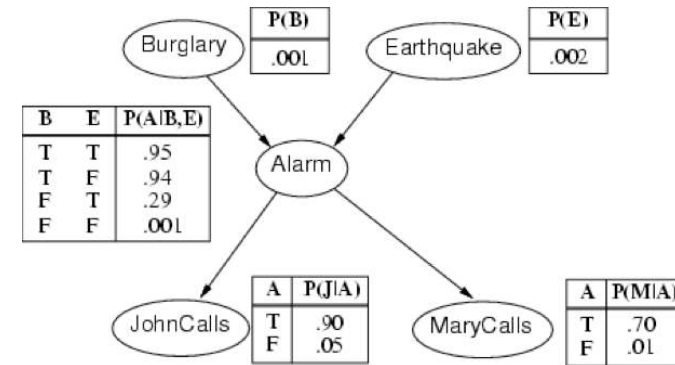
For example, if $\text{parents}(x_i) \leq k$ for each i , the specification of

$\Pr(x_1, x_2, \dots, x_n)$ can be done using $O(n2^k)$ numbers with a Bayes net.

An Example Problem

- I'm at work, neighbour John calls to say my alarm is ringing, but neighbour Mary doesn't call. Sometimes alarm is set off by minor earthquakes. Is there a burglar?
- Five variables: Burglary, Earthquake, Alarm, JohnCalls, MaryCalls
- We have these "causal" knowledge
 - A burglar can set the alarm off
 - An earthquake can set the alarm off
 - The alarm can cause Mary to call
 - The alarm can cause John to call

A Bayes Net



$$\Pr(j \wedge \neg m \wedge a \wedge b \wedge \neg e) = \Pr(j|a) \Pr(\neg m|a) \Pr(a|b, \neg e) \Pr(b) \Pr(\neg e)$$

$$= 0.9 \times 0.01 \times 0.94 \times 0.001 \times 0.998 = 8.44 \times 10^{-6}$$

Inference Tasks

- Simple queries: compute $\Pr(X_i | E = e)$
 - e.g. $\Pr(\text{Burglar} | \text{JohnCalls})$
- Conjunctive queries: $\Pr(X_i, X_j | E = e) = \Pr(X_i | E = e) \Pr(X_j | X_i, E = e)$
- Optimal decisions: attach utility nodes to form decision networks
 - $U(\text{outcome}) \Pr(\text{outcome} | \text{action}, \text{evidence})$

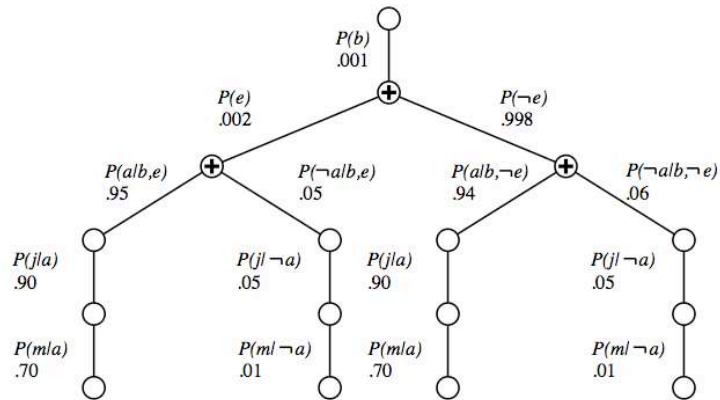
Inference Algorithm

Simple query on the burglary network: $\Pr(B|j, m)$

$$\begin{aligned} \Pr(B|j, m) &= \Pr(B, j, m) / \Pr(j, m) \\ &= \alpha \Pr(B, j, m) \\ &= \alpha \sum_e \sum_a \Pr(B, e, a, j, m) \\ &= \alpha \sum_e \sum_a \Pr(B) \Pr(e) \Pr(a|B, e) \Pr(j|a) \Pr(m|a) \\ &= \alpha \Pr(B) \sum_e \Pr(e) \sum_a \Pr(a|B, e) \Pr(j|a) \Pr(m|a) \end{aligned}$$

The order in which expressions are evaluated is important.

Left-to-Right Evaluation



Inefficient: repeated computation of $P(j|a)P(m|a)$ for each value of e

Variable Elimination Algorithm

Carry out summations right-to-left, storing intermediate results to avoid recomputation.

$$\begin{aligned}
 \Pr(B|j, m) &= \alpha \Pr(B) \sum_e \Pr(e) \sum_a \Pr(a|B, e) \Pr(j|a) \Pr(m|a) \\
 &= \alpha \Pr(B) \sum_e \Pr(e) \sum_a f_A(a, b, e) f_J(a) f_M(a) \\
 &= \alpha \Pr(B) \sum_e \Pr(e) f_{\bar{A}JM}(b, e) \quad (\text{sum out } A) \\
 &= \alpha \Pr(B) f_{\bar{E}\bar{A}JM}(b) \quad (\text{sum out } E) \\
 &= \alpha f_B(b) \times f_{\bar{E}\bar{A}JM}(b)
 \end{aligned}$$

Variable Elimination Algorithm

```

function ELIMINATION-ASK( $X, e, bn$ ) returns a distribution over  $X$ 
  inputs:  $X$ , the query variable
          $e$ , evidence specified as an event
          $bn$ , a belief network specifying joint distribution  $P(X_1, \dots, X_n)$ 
  factors  $\leftarrow []$ ; vars  $\leftarrow$  REVERSE(VARS[ $bn$ ])
  for each var in vars do
    factors  $\leftarrow$  [MAKE-FACTOR( $var, e$ )|factors]
    if var is a hidden variable then factors  $\leftarrow$  SUM-OUT( $var, factors$ )
  return NORMALIZE(POINTWISE-PRODUCT(factors))
    
```

Complexity of exact probabilistic inference: NP-hard in general.

Approximation Inference using Sampling

Basic idea:

1. Draw N samples from a sampling distribution S
2. Compute an approximate posterior probability \hat{P}
3. Show \hat{P} converges to true probability P as N increases

Three algorithms:

1. Sampling from the whole network
2. Rejection sampling: reject samples disagreeing with evidence
3. Likelihood weighting: use evidence to weight samples

Sampling from Whole Network

```
function PRIOR-SAMPLE(bn) returns an event sampled from bn
  inputs: bn, a belief network specifying joint distribution  $P(X_1, \dots, X_n)$ 
   $x \leftarrow$  an event with  $n$  elements
  for  $i = 1$  to  $n$  do
     $x_i \leftarrow$  a random sample from  $P(X_i \mid \text{parents}(X_i))$ 
    given the values of  $\text{Parents}(X_i)$  in  $x$ 
  return  $x$ 
```

PriorSample Analysis

Probability that PriorSample generates a particular event

$$S_{PS}(x_1, \dots, x_n) = \prod_{i=1}^n \Pr(x_i \mid \text{parents}(x_i)) = \Pr(x_1, \dots, x_n).$$

Let $N_{PS}(x_1, \dots, x_n)$ be the number of samples generated.

Then we have

$$\begin{aligned} \lim_{N \rightarrow \infty} \hat{\Pr}(x_1, \dots, x_n) &= \lim_{N \rightarrow \infty} N_{PS}(x_1, \dots, x_n) / N \\ &= \lim_{N \rightarrow \infty} S_{PS}(x_1, \dots, x_n) \\ &= \Pr(x_1, \dots, x_n) \end{aligned}$$

That is, estimates derived from PriorSample are consistent.

Rejection Sampling

$\hat{\Pr}(X|e)$ estimated from samples agreeing with e

```
function REJECTION-SAMPLING( $X, e, bn, N$ ) returns an estimate of  $P(X|e)$ 
  local variables:  $\mathbf{N}$ , a vector of counts over  $X$ , initially zero
  for  $j = 1$  to  $N$  do
     $x \leftarrow$  PRIOR-SAMPLE(bn)
    if  $x$  is consistent with  $e$  then
       $\mathbf{N}[x] \leftarrow \mathbf{N}[x] + 1$  where  $x$  is the value of  $X$  in  $x$ 
  return NORMALIZE( $\mathbf{N}[X]$ )
```

Easy to show that $\lim_{N \rightarrow \infty} \hat{\Pr}(X|e) = \Pr(X|e)$.

Problem: hopelessly expensive if $\Pr(e)$ is small

Likelihood Weighting

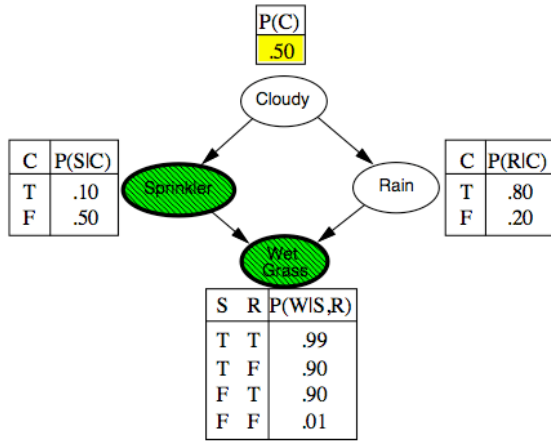
Idea: fix evidence variables, sample only non-evidence variables, and weight each sample by the likelihood it accords the evidence.

```
function LIKELIHOOD-WEIGHTING( $X, e, bn, N$ ) returns an estimate of  $P(X|e)$ 
  local variables:  $\mathbf{W}$ , a vector of weighted counts over  $X$ , initially zero
  for  $j = 1$  to  $N$  do
     $x, w \leftarrow$  WEIGHTED-SAMPLE(bn)
     $\mathbf{W}[x] \leftarrow \mathbf{W}[x] + w$  where  $x$  is the value of  $X$  in  $x$ 
  return NORMALIZE( $\mathbf{W}[X]$ )
```

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

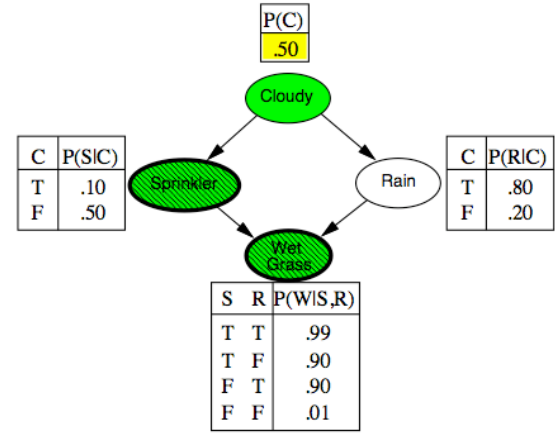
```
 $x \leftarrow$  an event with  $n$  elements;  $w \leftarrow 1$ 
for  $i = 1$  to  $n$  do
  if  $X_i$  has a value  $x_i$  in  $e$ 
    then  $w \leftarrow w \times P(X_i = x_i \mid \text{parents}(X_i))$ 
    else  $x_i \leftarrow$  a random sample from  $P(X_i \mid \text{parents}(X_i))$ 
return  $x, w$ 
```

Likelihood Weighting Example



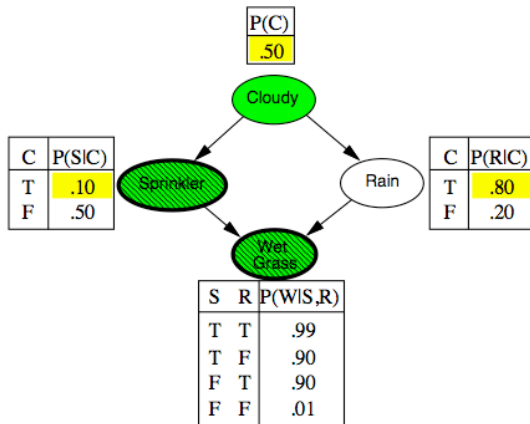
weight = 1.0

Likelihood Weighting Example



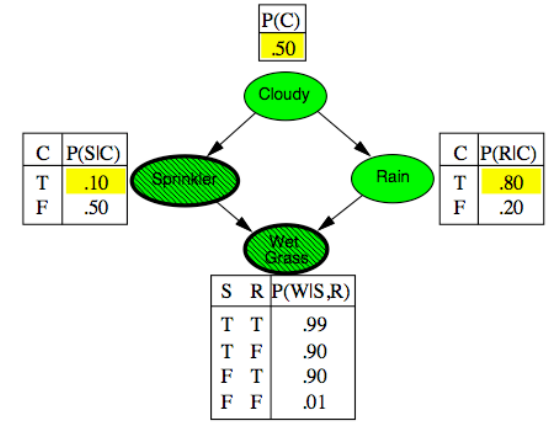
weight = 1.0

Likelihood Weighting Example



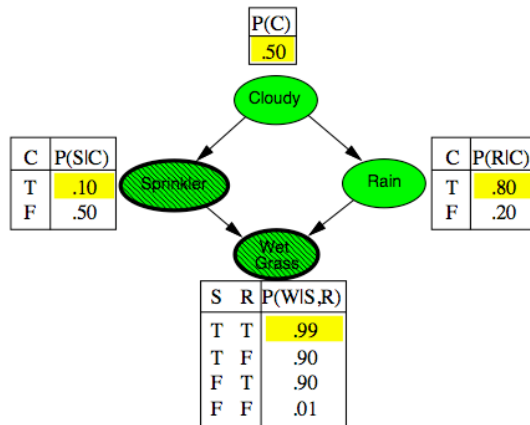
weight = 1.0×0.1

Likelihood Weighting Example



weight = 1.0×0.1

Likelihood Weighting Example



$$\text{weight} = 1.0 \times 0.1 \times 0.99 = 0.099$$

Likelihood Weighting Analysis

Sampling probability for WeightedSample is

$$S_{WS}(\mathbf{z}, \mathbf{e}) = \prod_{i=1}^{|\mathbf{z}|} \Pr(z_i | \text{parents}(Z_i))$$

Weight for a given sample (\mathbf{z}, \mathbf{e}) is

$$w(\mathbf{z}, \mathbf{e}) = \prod_{i=1}^{|\mathbf{e}|} \Pr(e_i | \text{parents}(E_i))$$

Weighted sampling probability is

$$\begin{aligned} S_{WS}(\mathbf{z}, \mathbf{e})w(\mathbf{z}, \mathbf{e}) &= \prod_{i=1}^{|\mathbf{z}|} \Pr(z_i | \text{parents}(Z_i)) \times \prod_{i=1}^{|\mathbf{e}|} \Pr(e_i | \text{parents}(E_i)) \\ &= \Pr(\mathbf{z}, \mathbf{e}) \end{aligned}$$