Probabilistic Reasoning

LESSON 12

Reading

Chapter 14

Outline

- Syntax of Bayesian networks
- Semantics of Bayesian networks
- Efficient representation of conditional distributions
- Exact inference by enumeration
- Exact inference by variable elimination
- Approximate inference by stochastic simulation*
- Approximate inference by Markov Chain Monte Carlo*

Motivations

- Full joint probability distribution can answer any question but can become intractably large as number of variable increases
- Specifying probabilities for atomic events can be difficult, e.g., large set of data, statistical estimates, etc.
- Independence and conditional independence reduce the probabilities needed for full joint probability distribution.

Bayesian networks

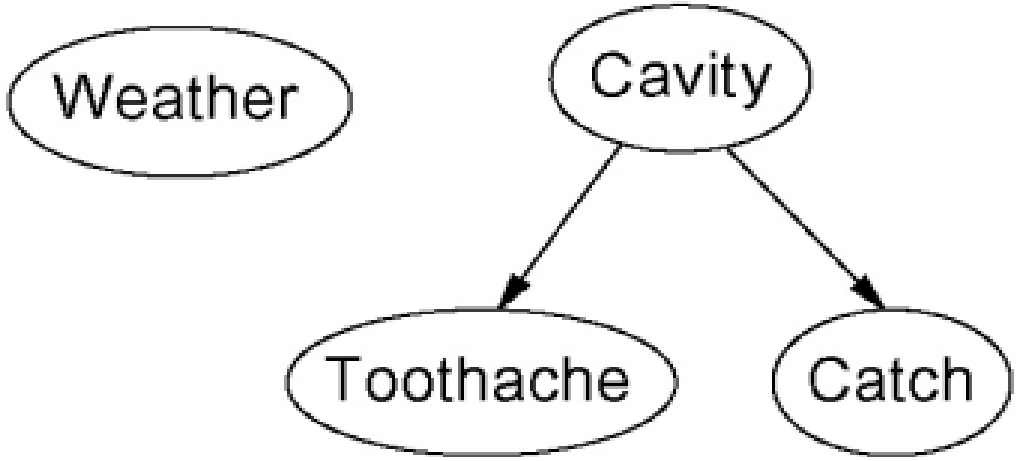
- A directed, acyclic graph (DAG)
- A set of nodes, one per variable (discrete or continuous)
- A set of directed links (arrows) connects pairs of nodes. *X* is a parent of *Y* if there is an arrow (direct influence) from node *X* to node *Y*.
- Each node X_i has a conditional probability distribution that quantifies the effect of the parents on the node.
- Combinations of the topology and the conditional distributions specify (implicitly) the full joint distribution for all the variables.

$$P\left(X_i \mid Parents(X_i)\right)$$

Bayesian networks

Example 1： The Teeth Disease Bayesian

Topology of network encodes conditional independence assertions:



Weather is independent of the other variables

Toothache and *Catch* are conditionally independent given *Cavity*

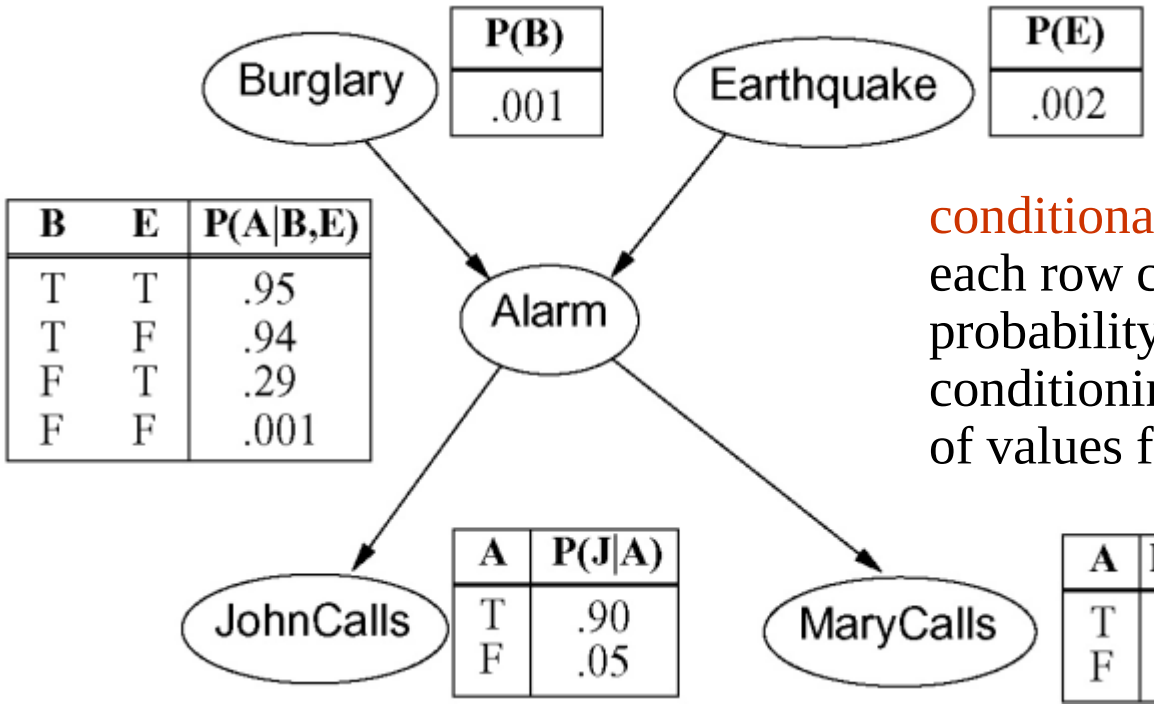
Example: Burglar alarm system

- I have a burglar alarm installed at home
  - It is fairly reliable at detecting a burglary, but also responds on occasion to minor earth quakes.
- I also have two neighbors, John and Mary
  - They have promised to call me at work when they hear the alarm
  - John always calls when he hears the alarm, but sometimes confuses the telephone ringing with the alarm and calls then, too.
  - Mary likes rather loud music and sometimes misses the alarm altogether.

- **Bayesian networks variables:**
  - ***Burglar, Earthquake, Alarm, JohnCalls, MaryCalls***

Example: Burglar alarm system

- Network topology reflects "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



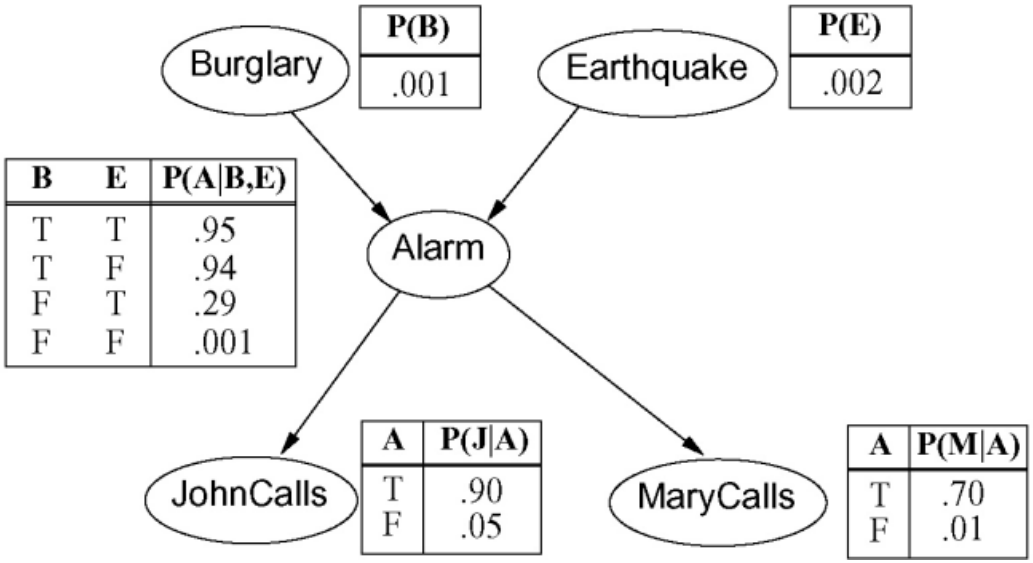
conditional probability table (CPT): each row contains the conditional probability of each node value for a conditioning case (a possible combination of values for the parent nodes).

Compactness of Bayesian networks

A CPT for Boolean X_i with k Boolean parents has *2^k* rows for the combinations of parent values. Each row requires one number *p* for X_i =*true* (the number for X_i =*false* is just *1-p*)

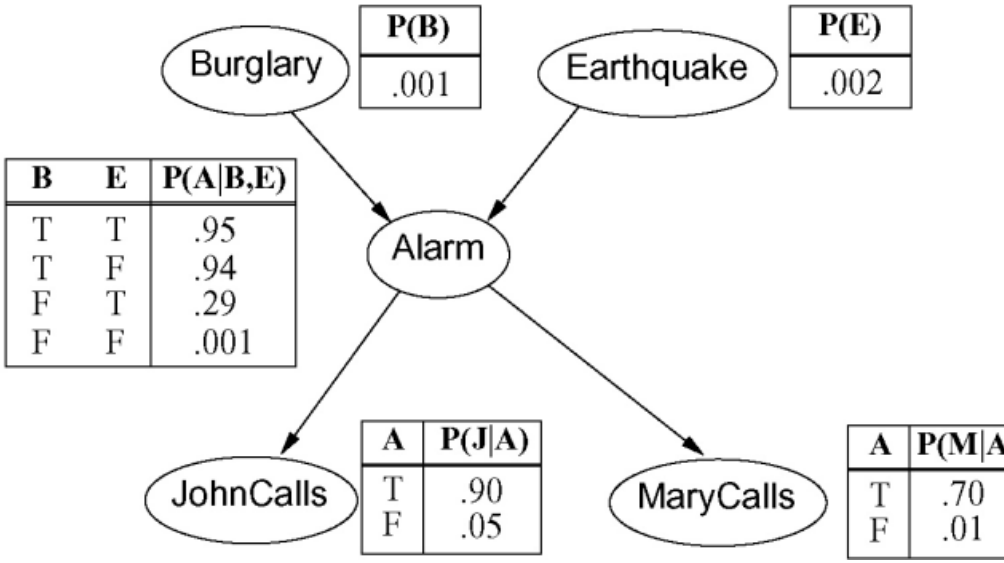
If each variable has no more than k parents, The complete network requires *O(n. 2^k)* numbers i.e., grows linearly with *n*, vs. *O(2^k)* for the full joint distribution

For Burglary net,
1+1+4+2+2=10 numbers
(vs. *2^5-1=31*)



Global semantics of Bayesian networks

Global semantics defines the full joint distribution as the product of the local conditional distributions



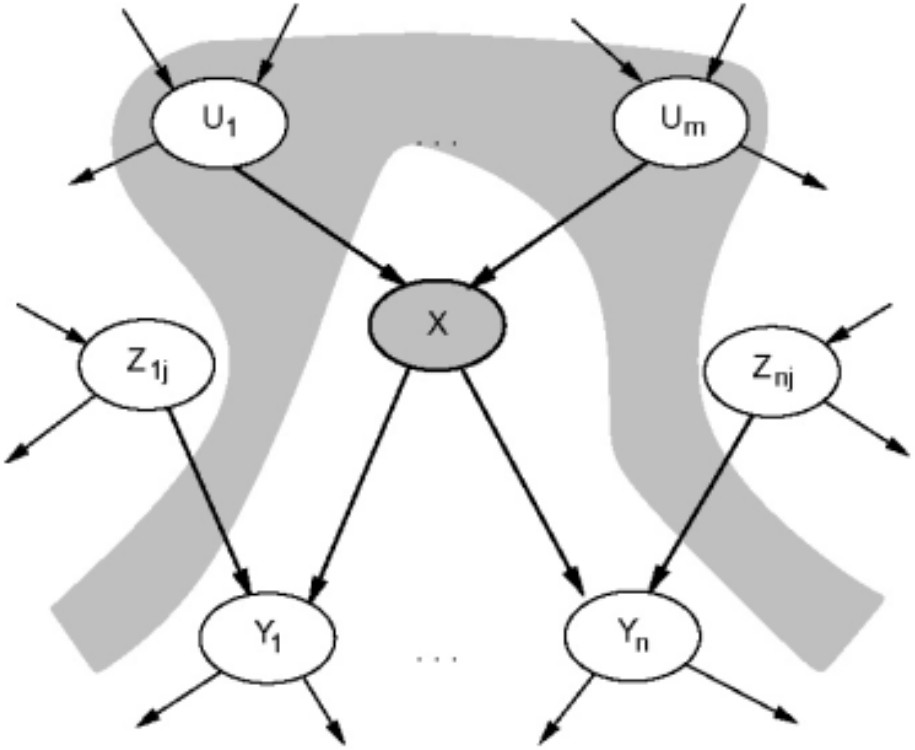
$$p(X_1,\cdots,X_n)=\prod_{i=1}^{n} p(X_i \mid Parents(X_i))$$

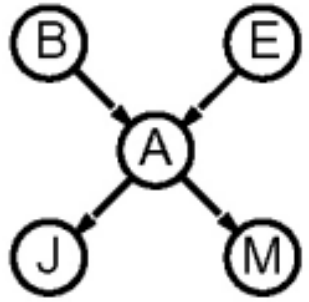
e.g.

$$p(j\wedge m\wedge a\wedge\neg b\wedge\neg e)$$
$$= p(j\mid a)p(m\mid \mathrm{a})p(a\mid\neg b,\neg e)p(\neg b)p(\neg e)$$
$$= 0.90\times 0.70\times 0.001\times 0.999\times 0.998$$
$$= 0.00062$$

Local semantics of Bayesian network

Local semantics: each node is conditionally independent of its nondescendants given its parents

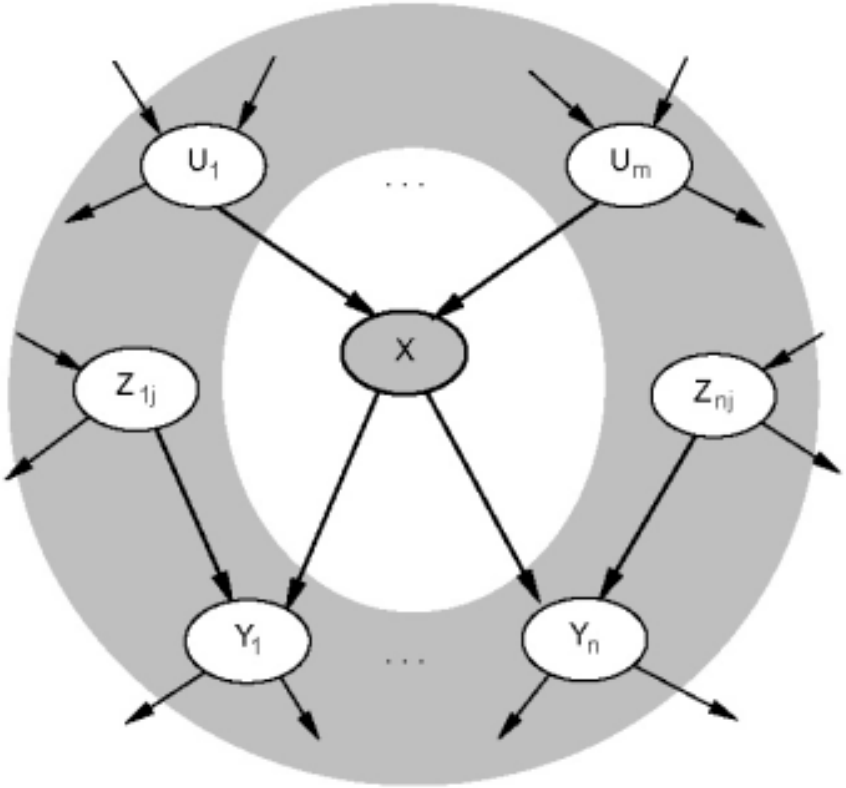


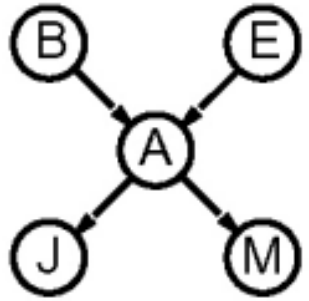


e.g., *JohnCalls* is independent of *Burglary* and *Earthquake*, given the value of *Alarm*.

Markov blanket

Each node is conditionally independent of all others given its Markov blanket: parents + children + children's parents





e.g., *Burglary* is independent of *JohnCalls* and *MaryCalls* , given the value of *Alarm* and *Earthquake*.

Constructing Bayesian networks

Need a method such that a series of locally testable assertions of conditional independence guarantees the required global semantics.

---

1. Choose an ordering of variables $X_1, \ldots, X_n$
2. For $i = 1$ to n
   add X_i to the network
   select parents from $X_1, \ldots, X_{i-1}$ such that
   $$\mathbf{P}(X_i|Parents(X_i)) = \mathbf{P}(X_i|X_1, \ldots, X_{i-1})$$

This choice of parents guarantees the global semantics:

$$\begin{aligned}\mathbf{P}(X_1, \ldots, X_n) &= \prod_{i=1}^{n} \mathbf{P}(X_i|X_1, \ldots, X_{i-1}) \quad \text{(chain rule)} \\ &= \prod_{i=1}^{n} \mathbf{P}(X_i|Parents(X_i)) \quad \text{(by construction)}\end{aligned}$$

The correct order in which to add nodes is to add the "root causes" first, then the variables they influence, and so on.

What happens if we choose the wrong order?

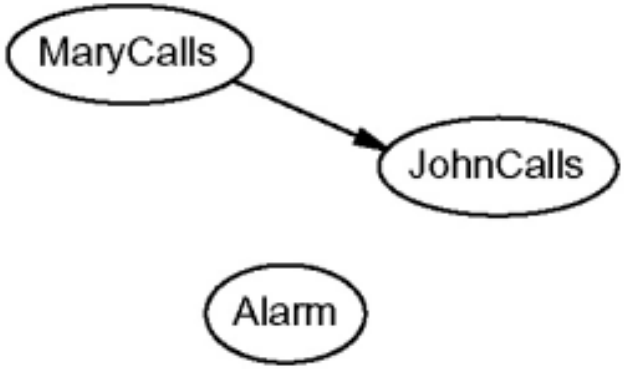
Example

Suppose we choose the ordering M, J, A, B, E

$P(J|M) = P(J)$?

Example

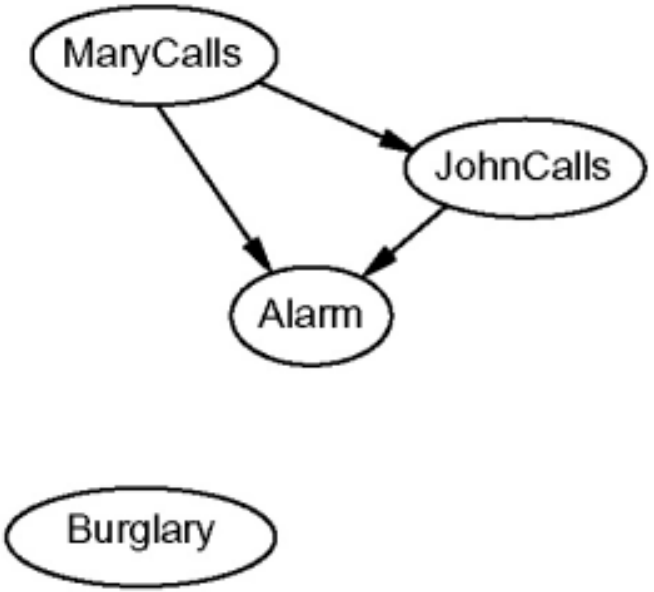
Suppose we choose the ordering M, J, A, B, E



$P(J|M) = P(J)$? No
$P(A|J, M) = P(A|J)$? $P(A|J, M) = P(A)$?

Example

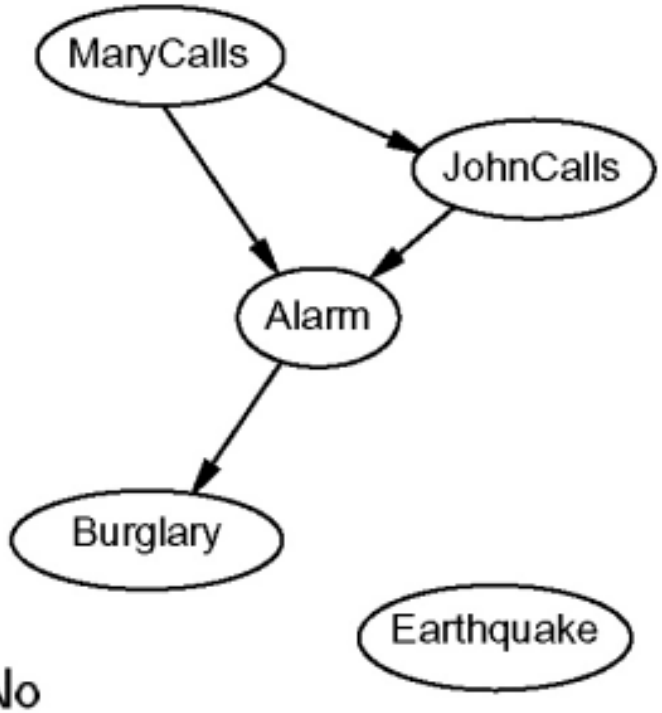
Suppose we choose the ordering M, J, A, B, E



$P(J|M) = P(J)$? No
$P(A|J, M) = P(A|J)$? $P(A|J, M) = P(A)$? No
$P(B|A, J, M) = P(B|A)$?
$P(B|A, J, M) = P(B)$?

Example

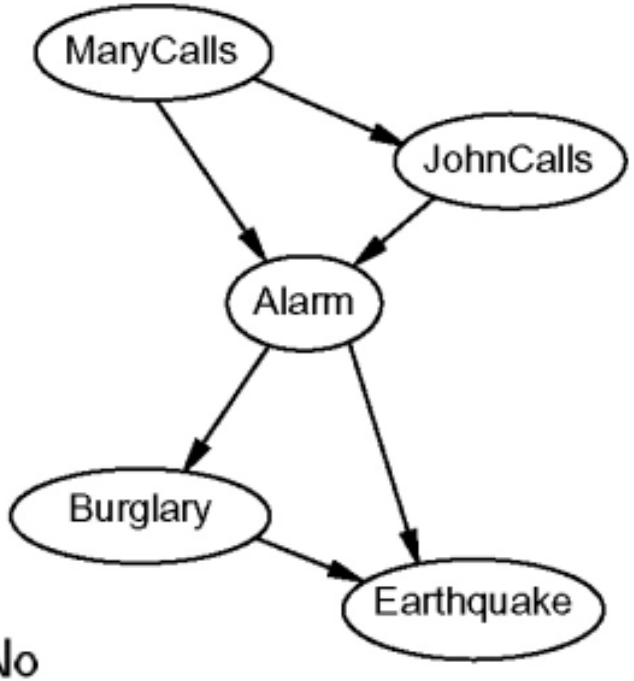
Suppose we choose the ordering M, J, A, B, E



$P(J|M) = P(J)$? No
$P(A|J, M) = P(A|J)$? $P(A|J, M) = P(A)$? No
$P(B|A, J, M) = P(B|A)$? Yes
$P(B|A, J, M) = P(B)$? No
$P(E|B, A, J, M) = P(E|A)$?
$P(E|B, A, J, M) = P(E|A, B)$?

Example

Suppose we choose the ordering M, J, A, B, E



$P(J|M) = P(J)$? No

$P(A|J, M) = P(A|J)$? $P(A|J, M) = P(A)$? No

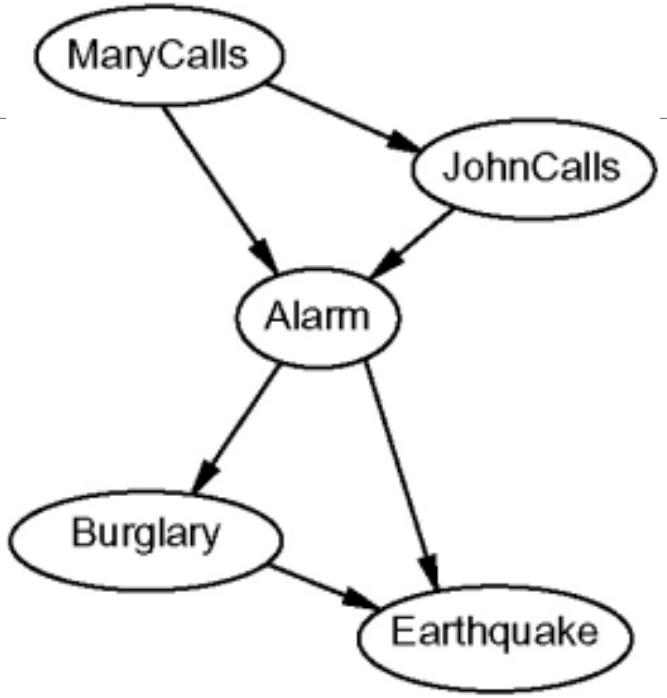
$P(B|A, J, M) = P(B|A)$? Yes

$P(B|A, J, M) = P(B)$? No

$P(E|B, A, J, M) = P(E|A)$? No

$P(E|B, A, J, M) = P(E|A, B)$? Yes

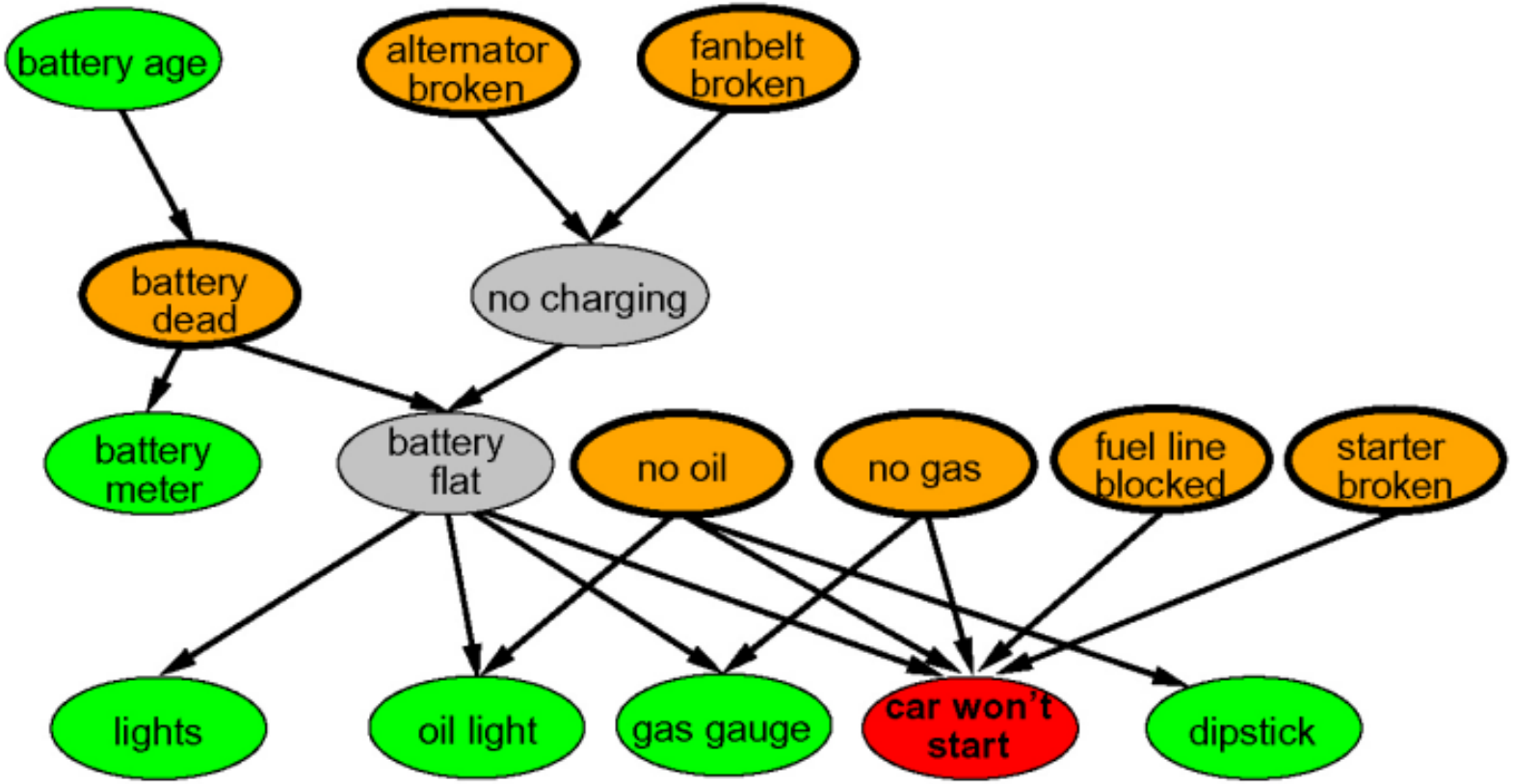
Example



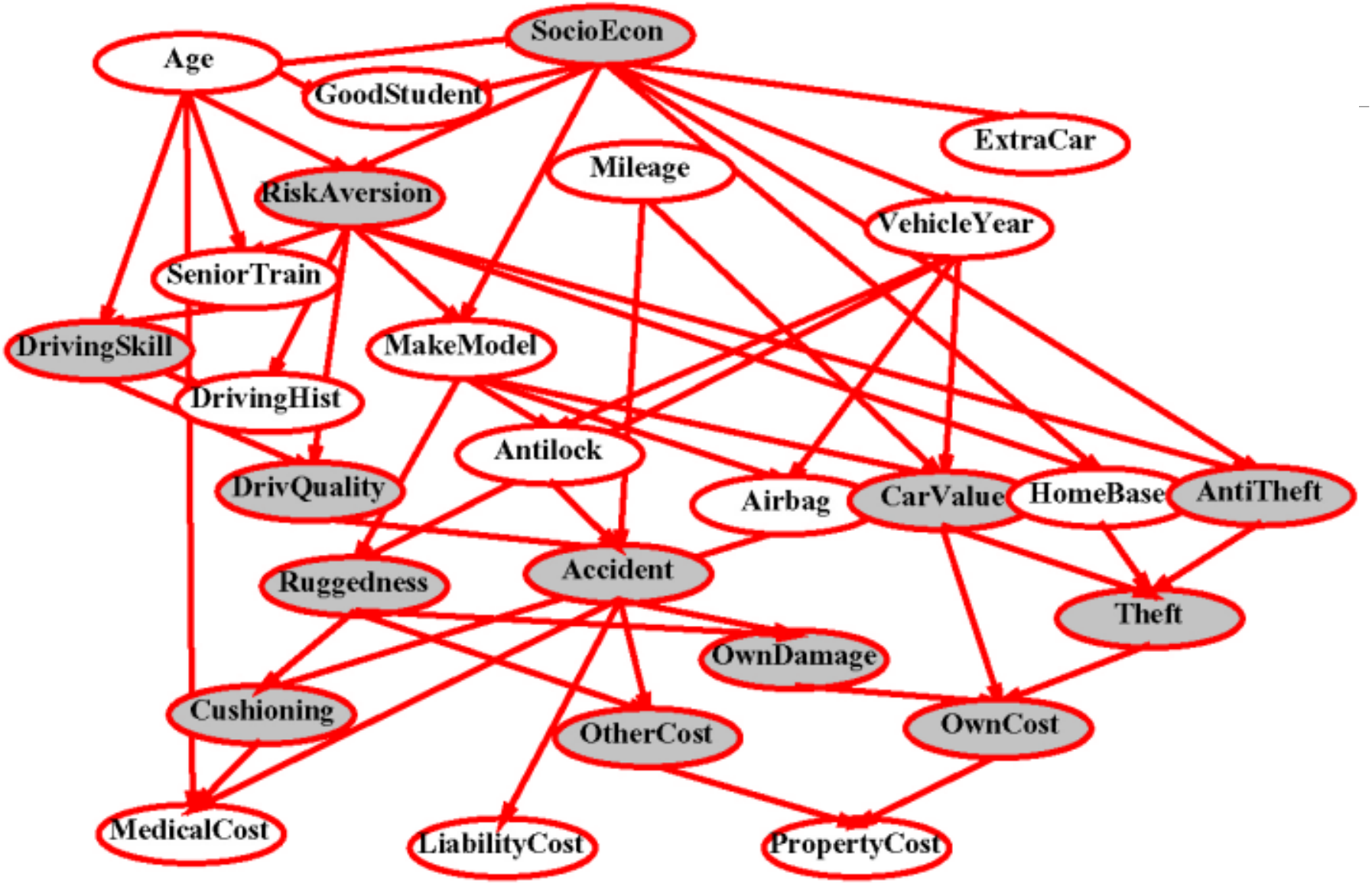
- Deciding conditional independence is hard in noncausal directions.
- Assessing conditional probabilities is hard in noncausal directions
- Network is less compact: 1 + 2 + 4 + 2 + 4 = 13 numbers needed

Example: Car diagnosis

- Initial evidence: car won't start
- Testable variables (green), "broken, so fix it" variables (orange)
- Hidden variables (gray) ensure sparse structure, reduce parameters



Example: Car insurance

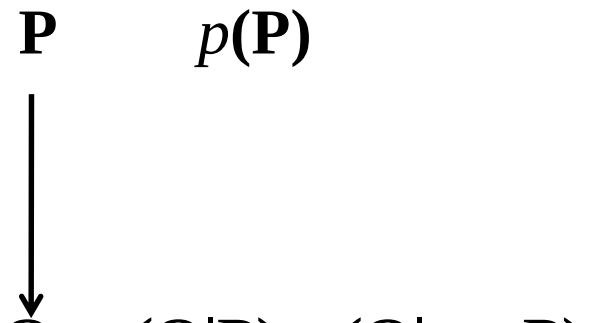


Class Exercise

1. Using the axioms of probability, prove
2. Given Bayesian Network

$$p(P \mid \mathrm{P} \wedge Q) = 1$$

- Calculate
- In what condition,



$$p(P \Rightarrow Q)$$

$$p(P \Rightarrow Q) = p(Q \mid P)$$

Efficient representation of conditional distributions

- CPT grows exponentially with number of parents $O(2^k)$
- CPT becomes infinite with continuous-valued parent or child

- Solution: canonical distribution that can be specified by a few parameters
- Simplest example: deterministic node whose value specified exactly by the values of its parents, with no uncertainty

$$X = f(Parents(X)) \text{ for some function } f$$

E.g., Boolean functions

$$NorthAmerican \Leftrightarrow Canadian \vee US \vee Mexican$$

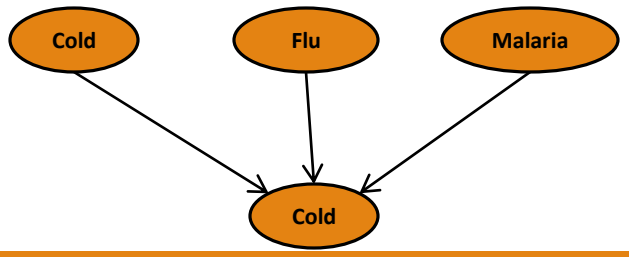
E.g., numerical relationships among continuous variables

$$\frac{\partial Level}{\partial t} = \text{inflow} + \text{precipitation} - \text{outflow} - \text{evaporation}$$

Efficient representation of conditional distributions

- "Noisy" logic relationships: uncertain relationships
- noisy-OR model allows for uncertainty about the ability of each parent to cause the child to be true, but the causal relationship between parent and child maybe inhibited.
  - E.g.: *Fever* is caused by *Cold*, *Flu*, or *Malaria*, but a patient could have a cold, but not exhibit a fever.
- Two assumptions of noisy-OR
  - parents include all the possible causes (can add leak node that covers "miscellaneous causes.")
  - inhibition of each parent is independent of inhibition of any other parents, e.g., whatever inhibits *Malaria* from causing a fever is independent of whatever inhibits *Flu* from causing a fever

$$P(\neg fever \mid cold, \neg flu, \neg malaria) = 0.6$$
$$P(\neg fever \mid \neg cold, flu, \neg malaria) = 0.2$$
$$P(\neg fever \mid \neg cold, \neg flu, malaria) = 0.1$$



Efficient representation of conditional distributions

- Other probabilities can be calculated from the product of the inhibition probabilities for each parent

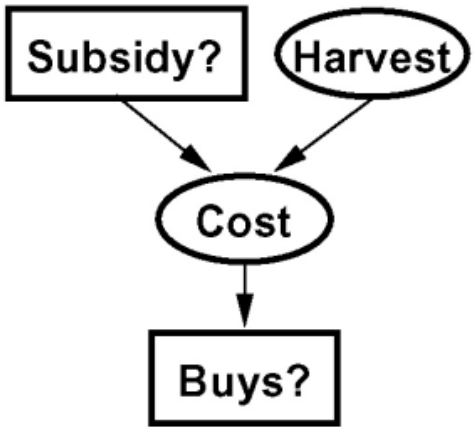
Cold	*Flu*	*Malaria*	$P(Fever)$	$P(\neg Fever)$
F	F	F	**0.0**	1.0
F	F	T	0.9	**0.1**
F	T	F	0.8	**0.2**
F	T	T	0.98	$0.02 = 0.2 \times 0.1$
T	F	F	0.4	**0.6**
T	F	T	0.94	$0.06 = 0.6 \times 0.1$
T	T	F	0.88	$0.12 = 0.6 \times 0.2$
T	T	T	0.988	$0.012 = 0.6 \times 0.2 \times 0.1$

- Number of parameters linear in number of parents

$O(2^k) \rightarrow O(k)$

Bayesian nets with continuous variables

- Hybrid Bayesian network: discrete variables + continuous variables
- Discrete (*Subsidy*? and *Buys*?); continuous (*Harvest* and *Cost*)



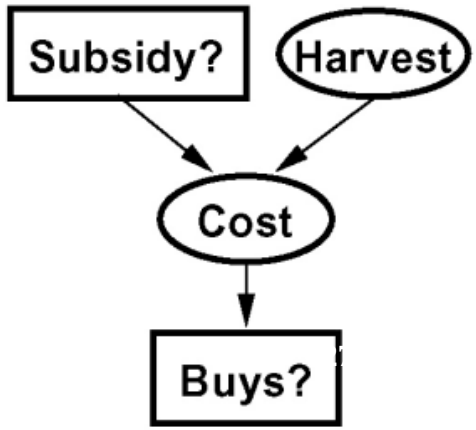
- Two options
  - discretization – possibly large errors, large CPTs
  - finitely parameterized canonical families(E.g.: Gaussian Distribution)
- Two kinds of conditional distributions for hybrid Bayesian network
  - continuous variable given discrete or continuous parents (e.g., *Cost*)
  - discrete variable given continuous parents (e.g., *Buys*?)

Continuous child variables

- Need one conditional density function for child variable given continuous parents, for each possible assignment to discrete parents
- Most common is the linear Gaussian model, e.g.,:

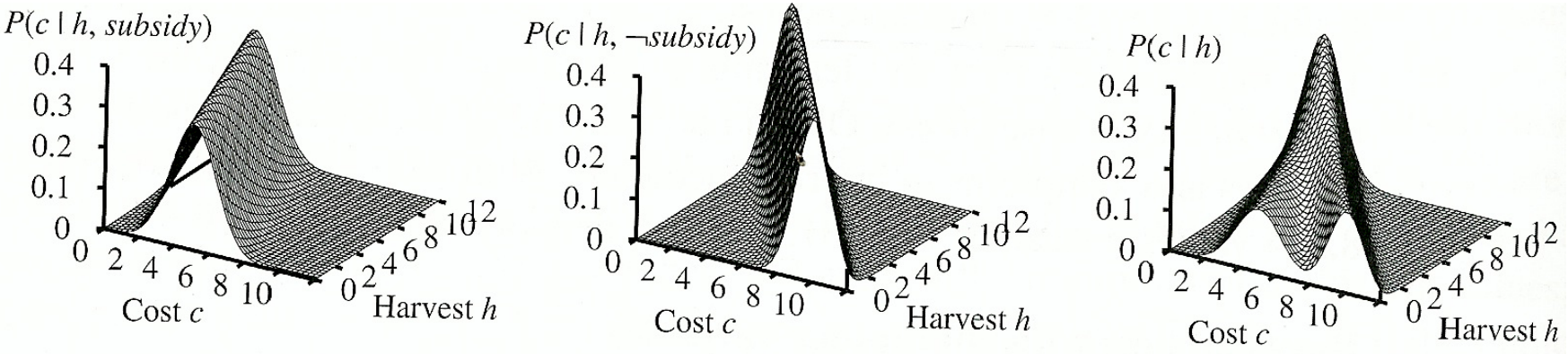
$$
\begin{aligned}
&P(Cost = c | Harvest = h, Subsidy? = true) \\
&= N(a_t h + b_t, \sigma_t)(c) \\
&= \frac{1}{\sigma_t \sqrt{2\pi}} exp\left(-\frac{1}{2}\left(\frac{c - (a_t h + b_t)}{\sigma_t}\right)^2\right)
\end{aligned}
$$

- Mean *Cost* varies linearly with *Harvest*, variance is fixed
- the linear model is reasonable only if the harvest size is limited to a narrow range



Continuous child variables

- Discrete + continuous linear Gaussian network is a conditional Gaussian network, i.e., a multivariate Gaussian distribution over all continuous variables for each combination of discrete variable values.

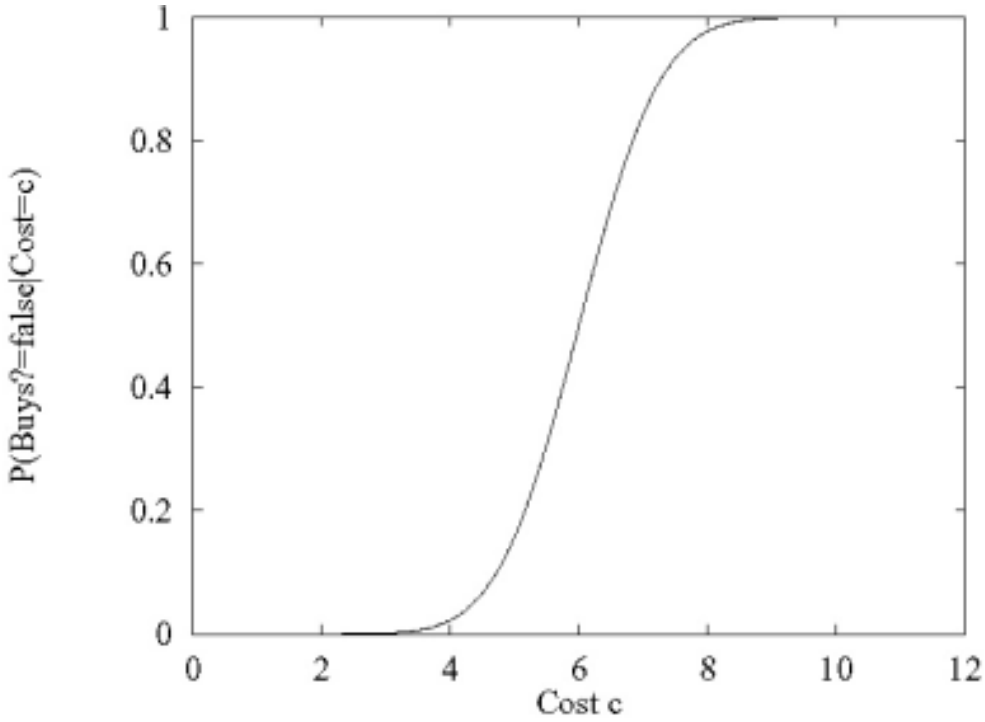


- A multivariate Gaussian distribution is a surface in more than one dimension that has a peak at the mean and drops off on all sides

Discrete variable with continuous parents

- Probability of *Buys*?
- given *Cost* should

be a "soft" threshold"



- Probit distribution uses

integral of Gaussian:

$$\Phi(x) = \int_{-\infty}^{x} N(0,1)(x)dx$$

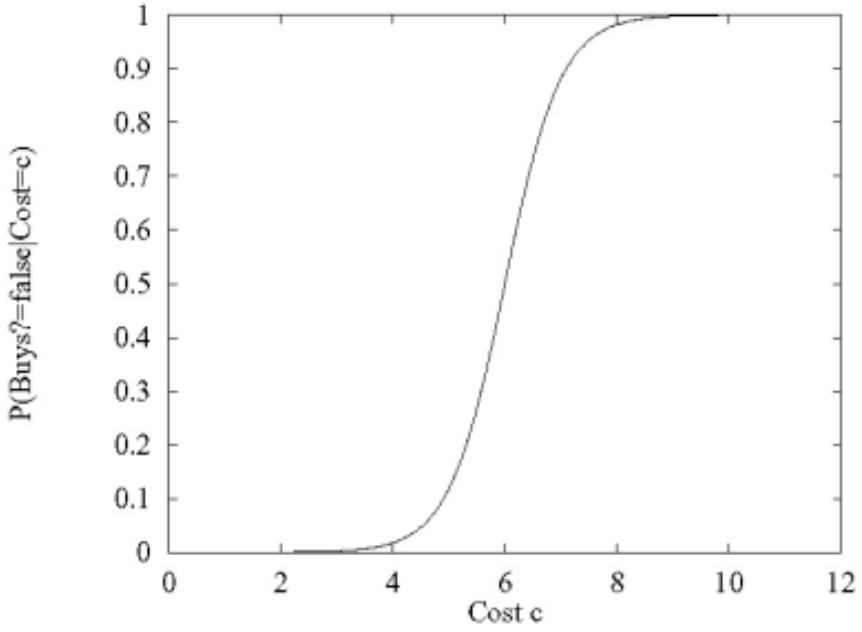
$$P(Buys? = true \mid Cost = c) = \Phi((-c + \mu)/\sigma)$$

Discrete variable with continuous parents

- Sigmoid (or logit) distribution also used in neural networks:

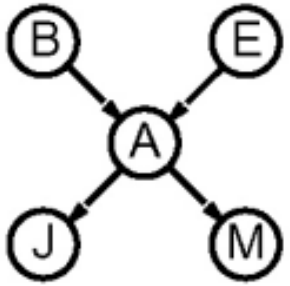
$$P(Buys? = true \mid Cost = c) = \frac{1}{1 + exp(-2\frac{-c+\mu}{\sigma})}$$

- Sigmoid has similar shape to probit but much longer tails



Exact inference by enumeration

- A query can be answered using a Bayesian network by computing sums of products of conditional probabilities from the network.



Simple query on the burglary network:

$$\begin{aligned}&\mathbf{P}(B|j,m)\\&=\mathbf{P}(B,j,m)/P(j,m)\\&=\alpha\mathbf{P}(B,j,m)\\&=\alpha\Sigma_e\Sigma_a\mathbf{P}(B,e,a,j,m)\end{aligned}$$

sum over hidden variables: *earthquake* and *alarm*

Rewrite full joint entries using product of CPT entries:

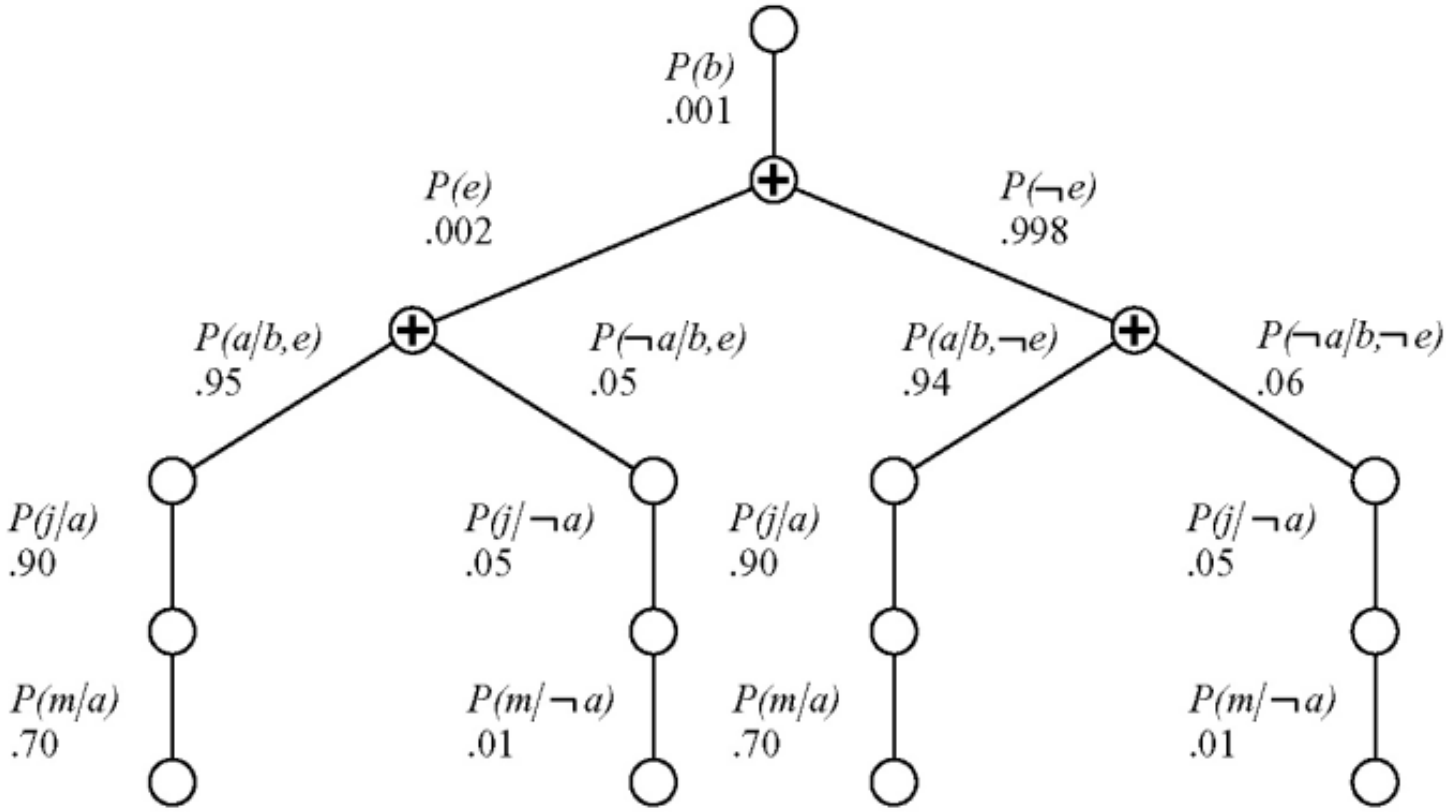
$$\begin{aligned}&\mathbf{P}(B|j,m)\\&=\alpha\Sigma_e\Sigma_a\mathbf{P}(B)P(e)\mathbf{P}(a|B,e)P(j|a)P(m|a)\\&=\alpha\mathbf{P}(B)\Sigma_eP(e)\Sigma_a\mathbf{P}(a|B,e)P(j|a)P(m|a)\end{aligned}$$

Recursive depth-first enumeration: $O(n)$ space, $O(d^n)$ time

d = 2 when we have n Boolean variables

Evaluation tree

Enumeration is inefficient: repeated computation
e.g., computes $P(j|a)P(m|a)$ for each value of e



Exact inference by variable elimination

- Do the calculation once and save the results for later use – idea of dynamic programming
- Variable elimination: carry out summations right-to-left, storing intermediate results (factors) to avoid re-computation

$$
\begin{aligned}
&\mathbf{P}(B|j,m) \\
&= \alpha \underbrace{\mathbf{P}(B)}_{B} \Sigma_e \underbrace{P(e)}_{E} \Sigma_a \underbrace{\mathbf{P}(a|B,e)}_{A} \underbrace{P(j|a)}_{J} \underbrace{P(m|a)}_{M} \\
&= \alpha \mathbf{P}(B) \Sigma_e P(e) \Sigma_a \mathbf{P}(a|B,e) P(j|a) f_M(a) \\
&= \alpha \mathbf{P}(B) \Sigma_e P(e) \Sigma_a \mathbf{P}(a|B,e) f_J(a) f_M(a) \\
&= \alpha \mathbf{P}(B) \Sigma_e P(e) \Sigma_a f_A(a,b,e) f_J(a) f_M(a) \\
&= \alpha \mathbf{P}(B) \Sigma_e P(e) f_{\bar{A}JM}(b,e) \text{ (sum out } A\text{)} \\
&= \alpha \mathbf{P}(B) f_{\bar{E}\bar{A}JM}(b) \text{ (sum out } E\text{)} \\
&= \alpha f_B(b) \times f_{\bar{E}\bar{A}JM}(b)
\end{aligned}
$$

Variable elimination – basic operations

Summing out a variable from a product of factors:

move any constant factors outside the summation

add up submatrices in pointwise product of remaining factors

$$\sum_x f_1 \times \cdots \times f_k = f_1 \times \cdots \times f_i \sum_x f_{i+1} \times \cdots \times f_k = f_1 \times \cdots \times f_i \times f_{\bar{X}}$$

assuming $f_1, \ldots, f_i$ do not depend on X

Pointwise product of factors f_1 and f_2:

$$f_1(x_1, \ldots, x_j, y_1, \ldots, y_k) \times f_2(y_1, \ldots, y_k, z_1, \ldots, z_l)$$
$$= f(x_1, \ldots, x_j, y_1, \ldots, y_k, z_1, \ldots, z_l)$$

E.g., $f_1(a, b) \times f_2(b, c) = f(a, b, c)$

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

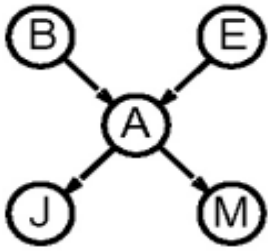
A	B	C	$\mathbf{f}_3(A,B,C)$
T	T	T	$.3 \times .2$
T	T	F	$.3 \times .8$
T	F	T	$.7 \times .6$
T	F	F	$.7 \times .4$
F	T	T	$.9 \times .2$
F	T	F	$.9 \times .8$
F	F	T	$.1 \times .6$
F	F	F	$.1 \times .4$

Variable elimination – irrelevant variables

Consider the query $P(JohnCalls|Burglary = true)$

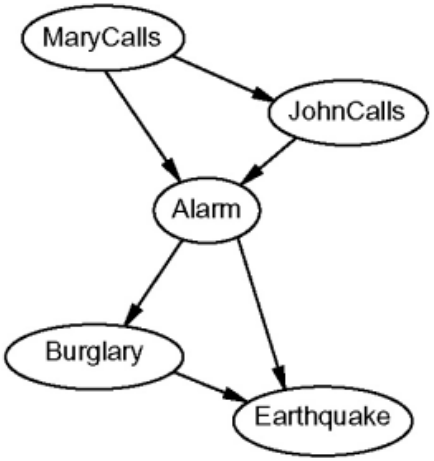
$$P(J|b) = \alpha P(b) \sum_e P(e) \sum_a P(a|b,e) P(J|a) \sum_m P(m|a)$$

Sum over m is identically 1; M is **irrelevant** to the query



Y is irrelevant unless $Y \in Ancestors(\{X\} \cup \mathbf{E})$

Here, $X = JohnCalls$, $\mathbf{E} = \{Burglary\}$, and
$Ancestors(\{X\} \cup \mathbf{E}) = \{Alarm, Earthquake\}$
so M is irrelevant



The complexity of variable elimination

- Single connected networks (or polytrees)
  - any two nodes are connected by at most one (undirected) path
  - time and space cost of variable elimination are $O(d^k n)$, $d = 2$ (n Boolean variables)

    i.e., linear in the number of variables (nodes) if the number of parents of each node is bounded by a constant
- Multiply connected network
  - variable elimination can have exponential time and space complexity even the number of parents per node is bounded.

Approximate inference by stochastic simulation*

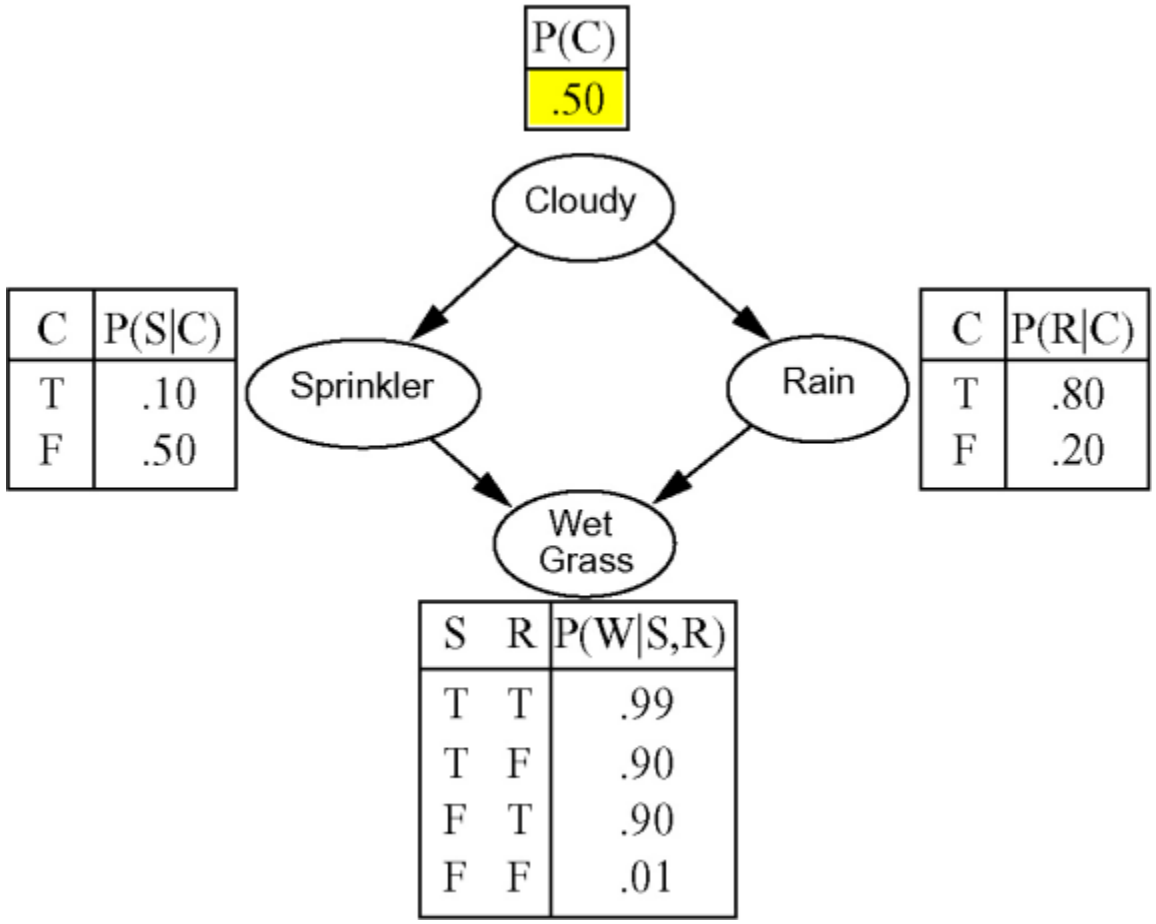
Basic idea:

1) Draw N samples from a sampling distribution S
2) Compute an approximate posterior probability $\hat{P}$
3) Show this converges to the true probability P

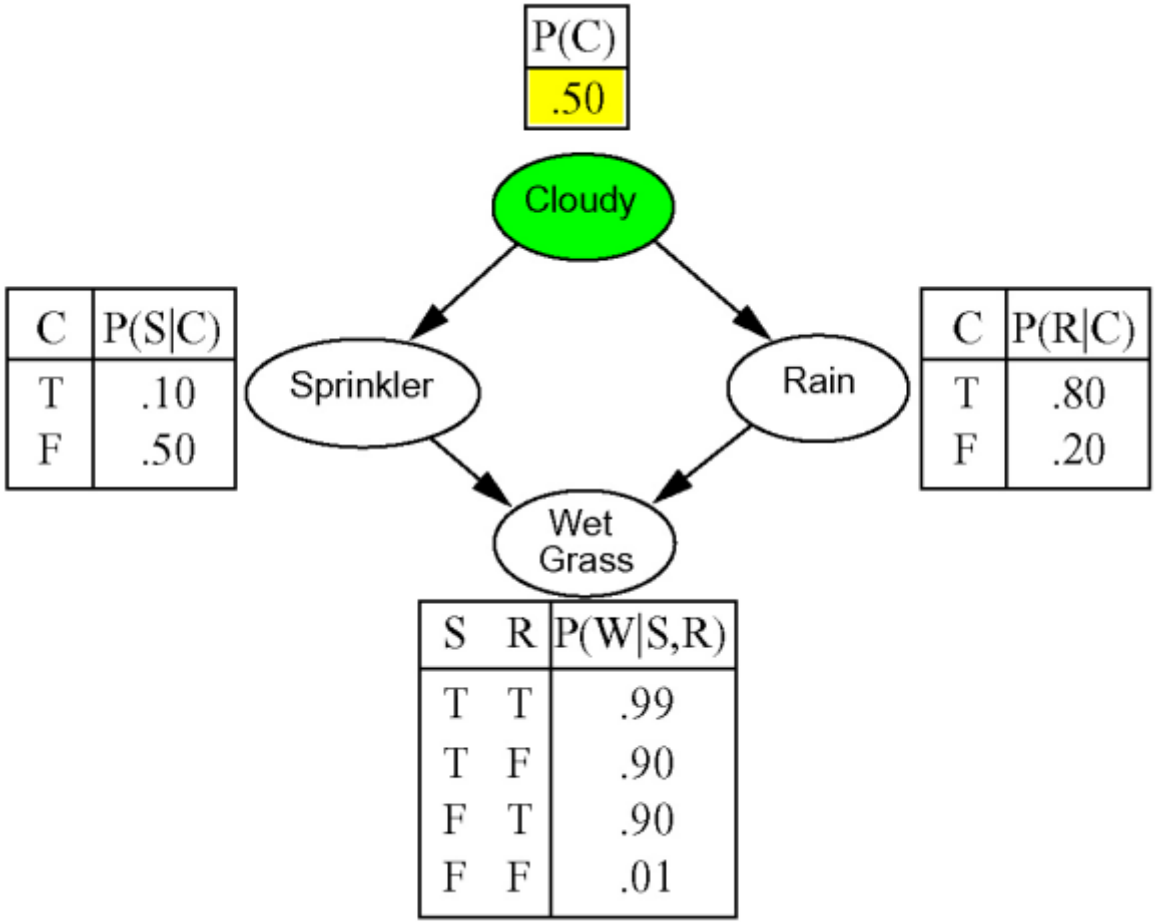
- Direct sampling
  - Generate events from (an empty) network that has no associated evidence
  - Rejection sampling: reject samples disagreeing with evidence
  - Likelihood weighting: use evidence to weight samples

- Markov chain Monte Carlo (MCMC)*
  - sample from a stochastic process whose stationary distribution is the true posterior

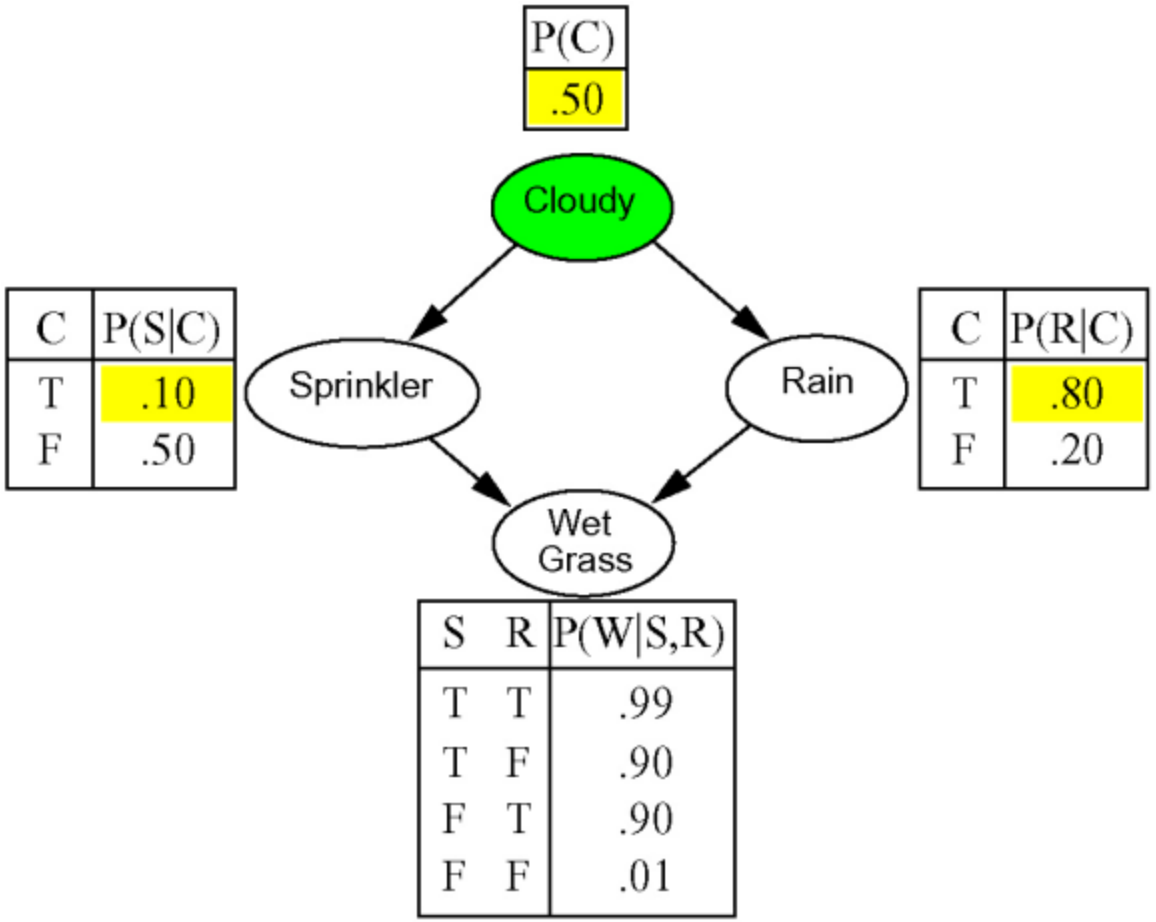
Example of sampling from an empty network



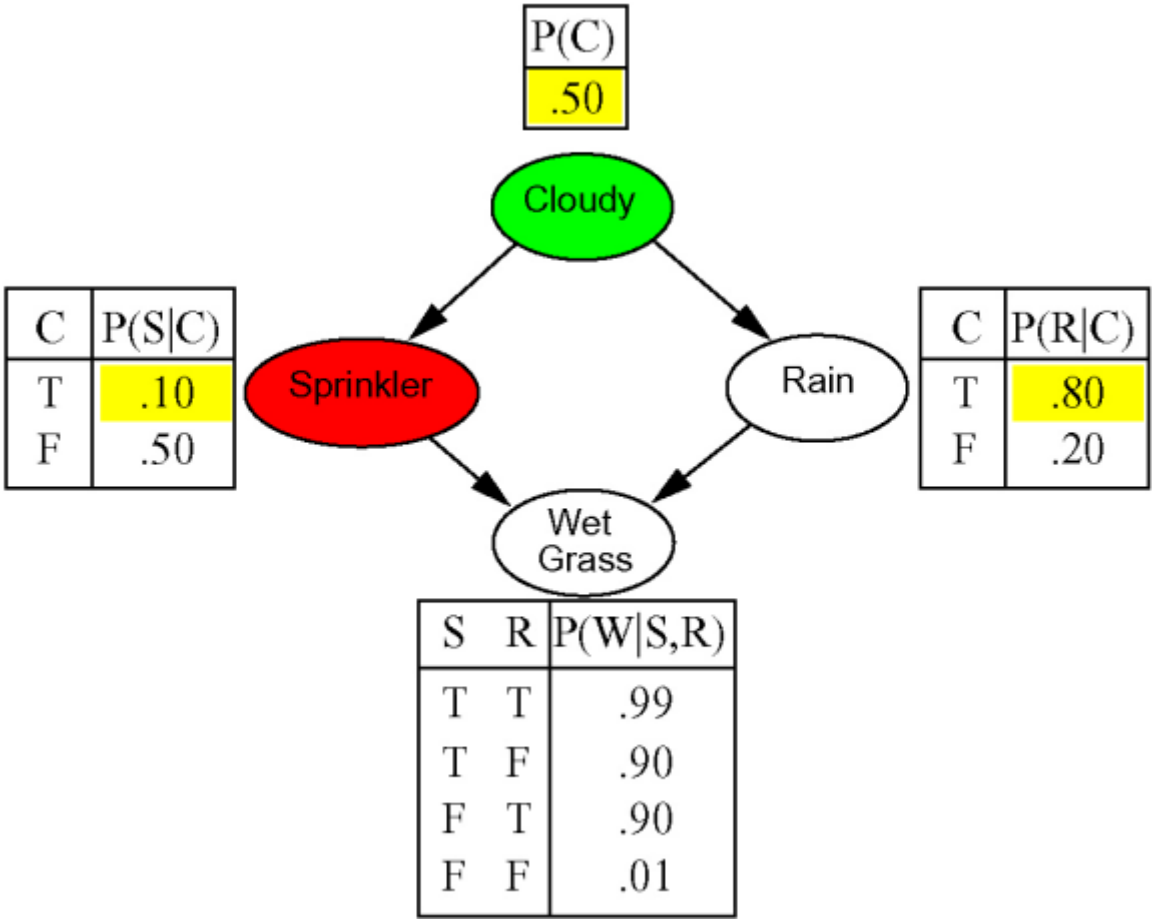
Example of sampling from an empty network



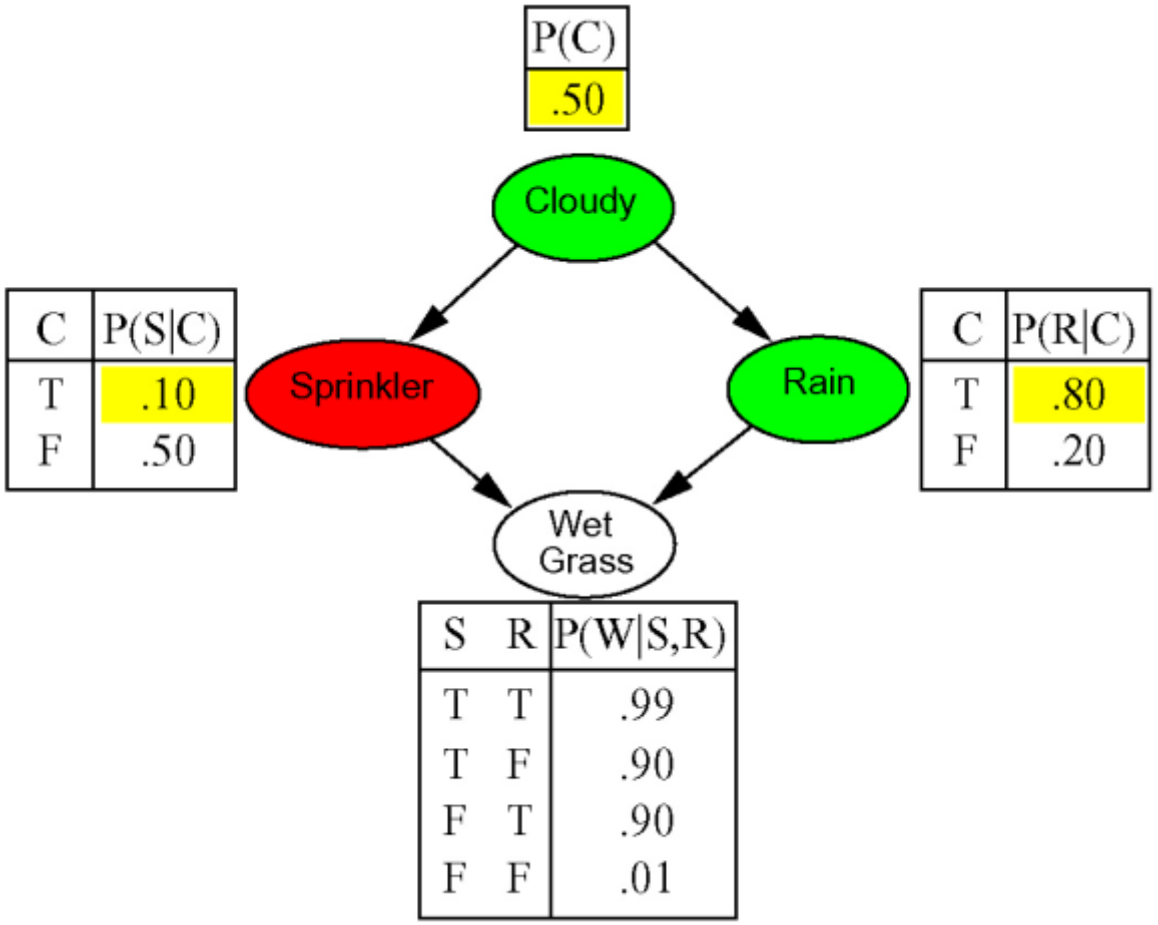
Example of sampling from an empty network



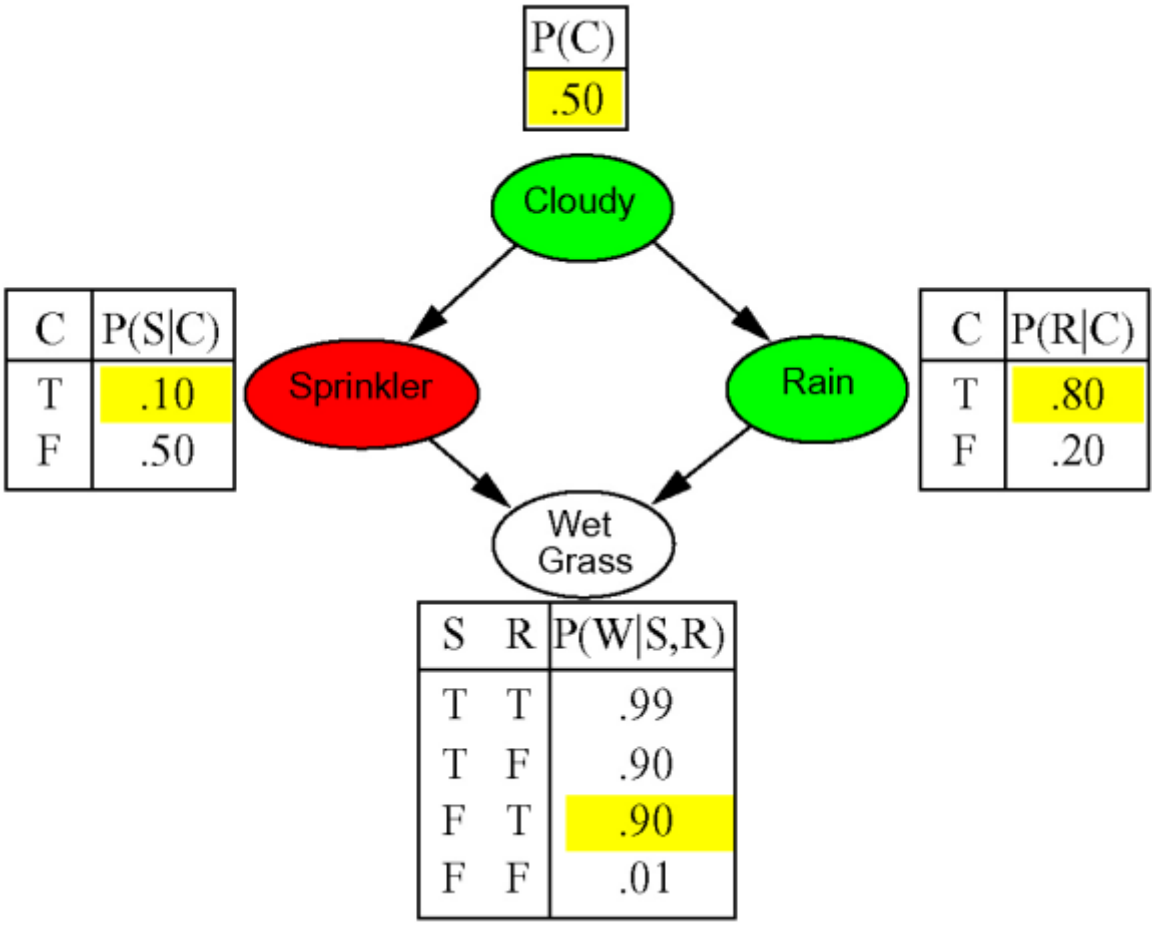
Example of sampling from an empty network



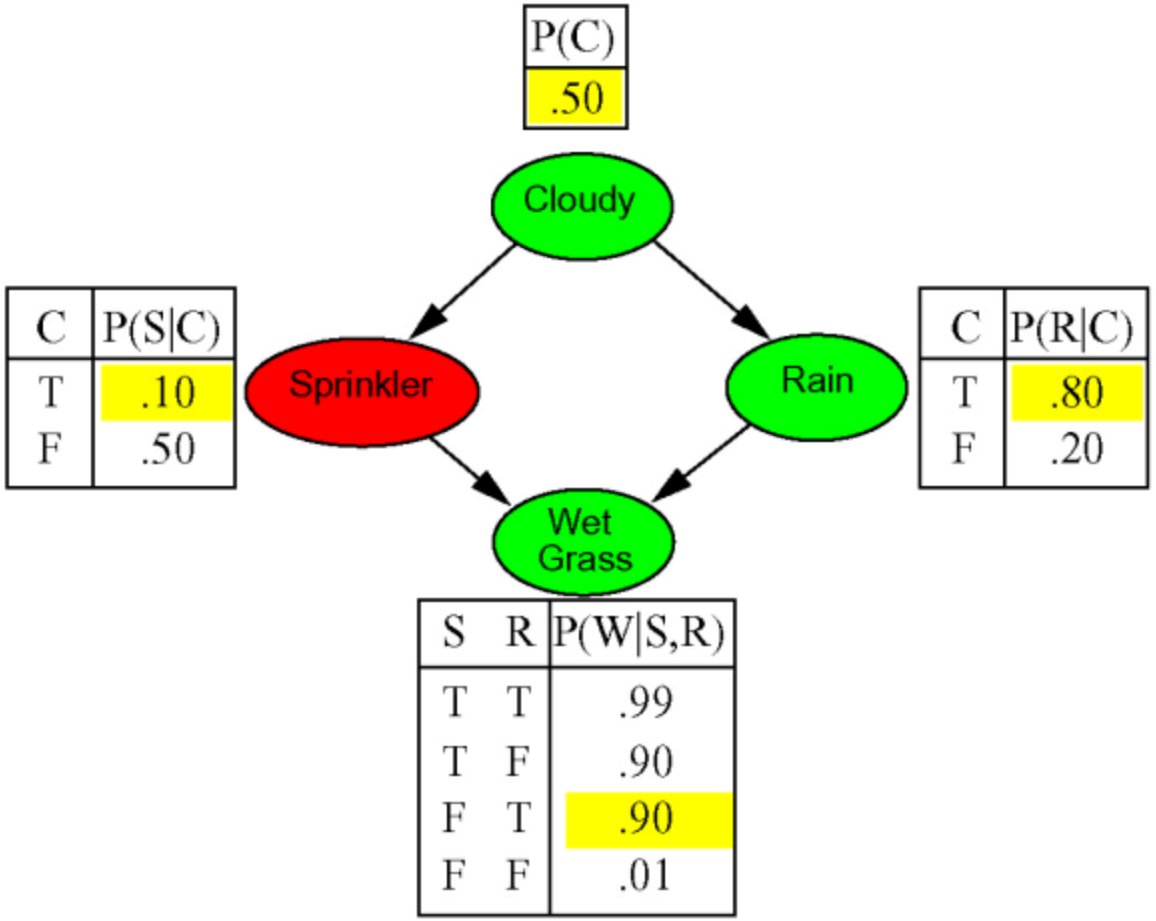
Example of sampling from an empty network



Example of sampling from an empty network



Example of sampling from an empty network



Example of sampling from an empty network

Probability that PRIORSAMPLE generates a particular event

$$S_{PS}(x_1 \ldots x_n) = \Pi_{i=1}^{n} P(x_i | Parents(X_i)) = P(x_1 \ldots x_n)$$

i.e., the true prior probability

E.g., $S_{PS}(t, f, t, t) = 0.5 \times 0.9 \times 0.8 \times 0.9 = 0.324 = P(t, f, t, t)$

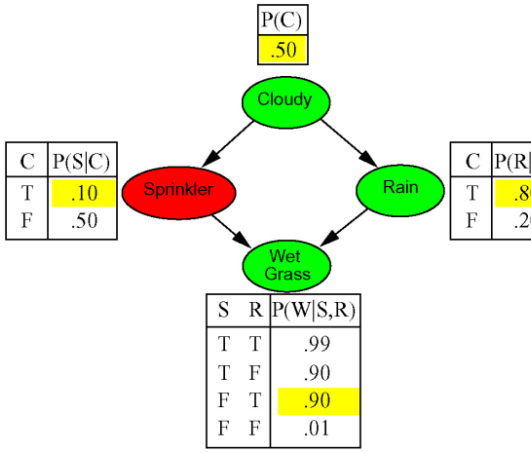
Let $N_{PS}(x_1 \ldots x_n)$ be the number of samples generated for event $x_1, \ldots, x_n$

Then we have

$$\begin{aligned} \lim_{N \to \infty} \hat{P}(x_1, \ldots, x_n) &= \lim_{N \to \infty} N_{PS}(x_1, \ldots, x_n)/N \\ &= S_{PS}(x_1, \ldots, x_n) \\ &= P(x_1 \ldots x_n) \end{aligned}$$

That is, estimates derived from PRIORSAMPLE are consistent

Shorthand: $\hat{P}(x_1, \ldots, x_n) \approx P(x_1 \ldots x_n)$



Rejection sampling

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

E.g., estimate $\mathbf{P}(Rain|Sprinkler = true)$ using 100 samples
27 samples have $Sprinkler = true$
Of these, 8 have $Rain = true$ and 19 have $Rain = false$.

$\hat{\mathbf{P}}(Rain|Sprinkler = true) = \text{NORMALIZE}(\langle 8, 19\rangle) = \langle 0.296, 0.704\rangle$

$$\begin{aligned}
\hat{\mathbf{P}}(X|\mathbf{e}) &= \alpha \mathbf{N}_{PS}(X, \mathbf{e}) && \text{(algorithm defn.)} \\
&= \mathbf{N}_{PS}(X, \mathbf{e})/N_{PS}(\mathbf{e}) && \text{(normalized by } N_{PS}(\mathbf{e})\text{)} \\
&\approx \mathbf{P}(X, \mathbf{e})/P(\mathbf{e}) && \text{(property of PRIORSAMPLE)} \\
&= \mathbf{P}(X|\mathbf{e}) && \text{(defn. of conditional probability)}
\end{aligned}$$

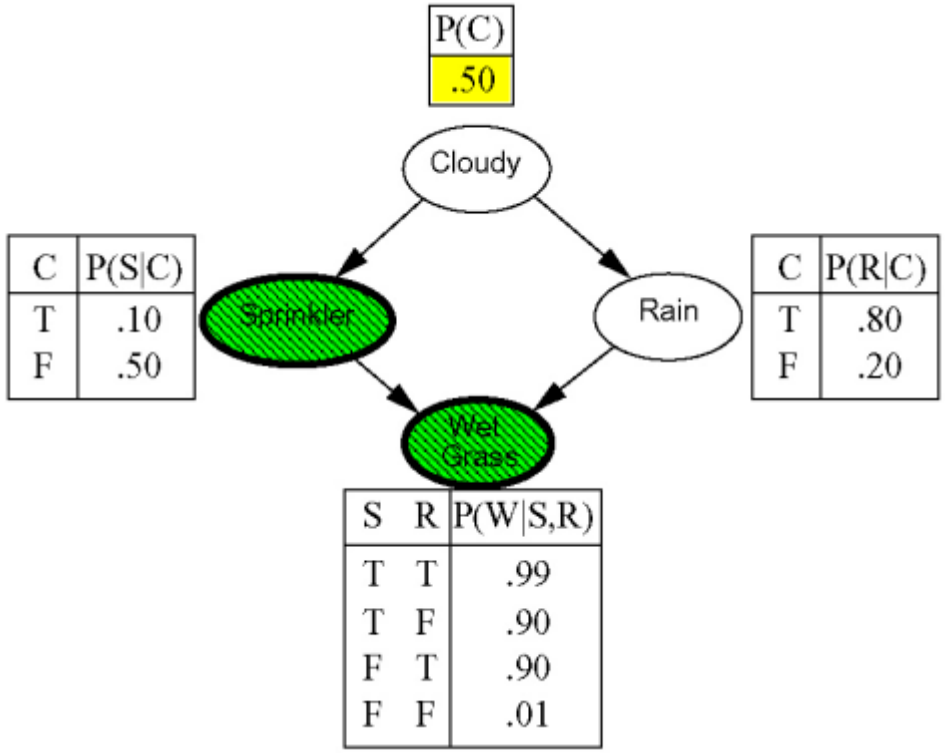
Hence rejection sampling returns consistent posterior estimates

Problem: hopelessly expensive if $P(\mathbf{e})$ is small

$P(\mathbf{e})$ drops off exponentially with number of evidence variables!

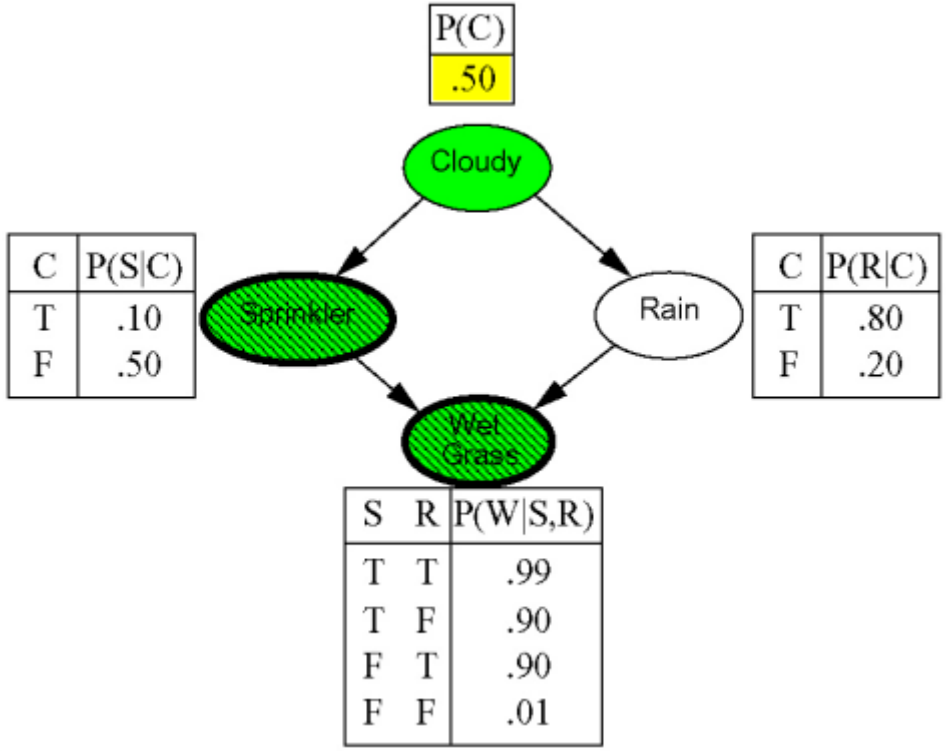
Likelihood weighting

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



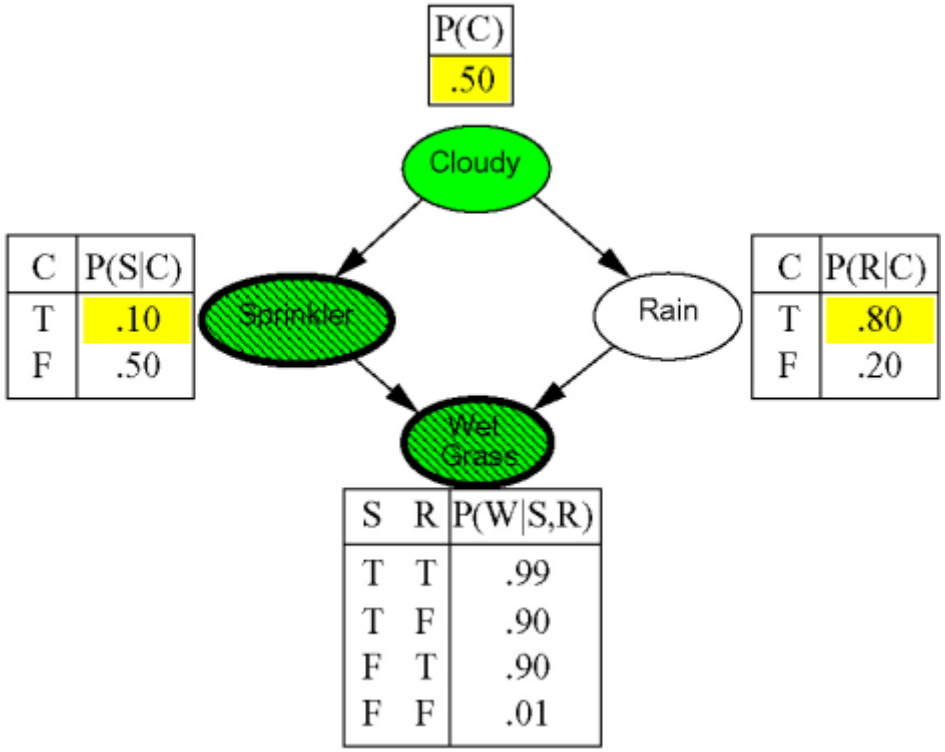
$w = 1.0$

Likelihood weighting



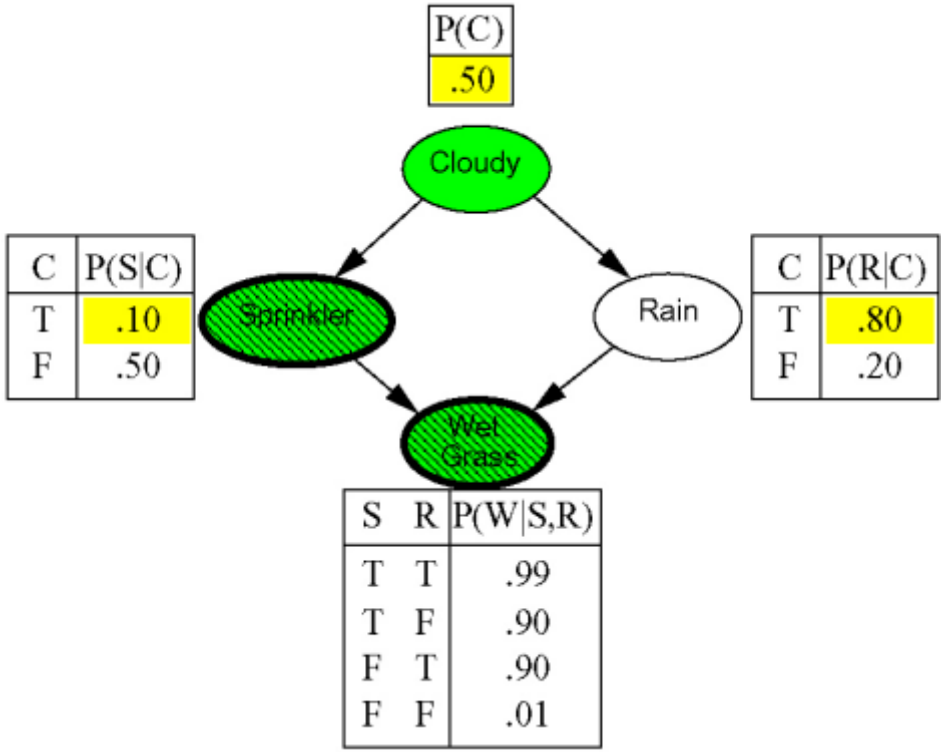
$w = 1.0$

Likelihood weighting



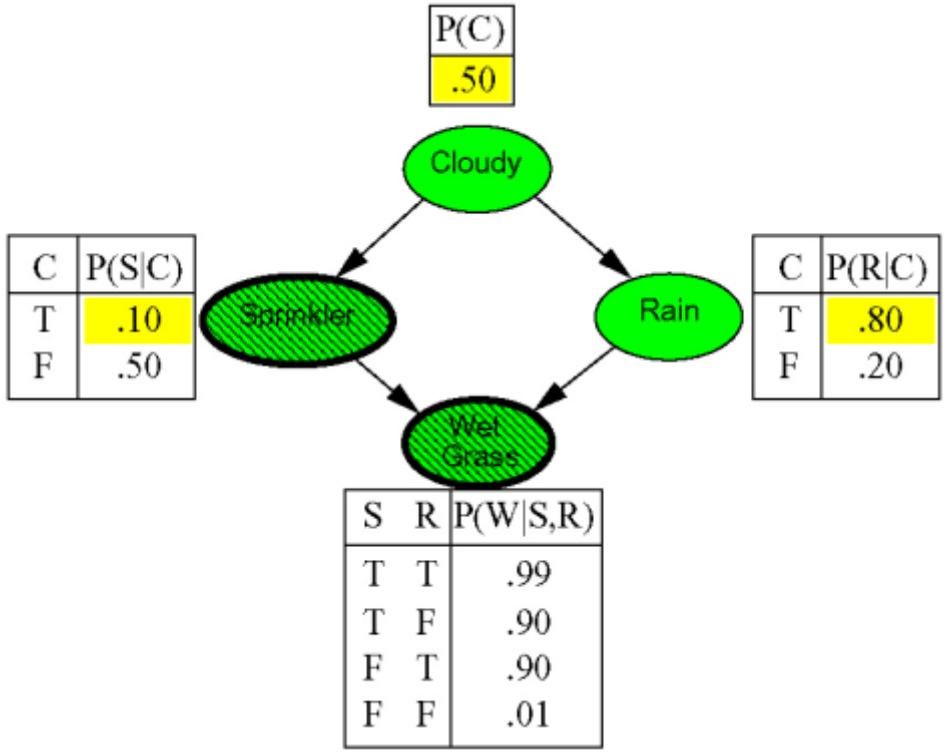
$w = 1.0$

Likelihood weighting



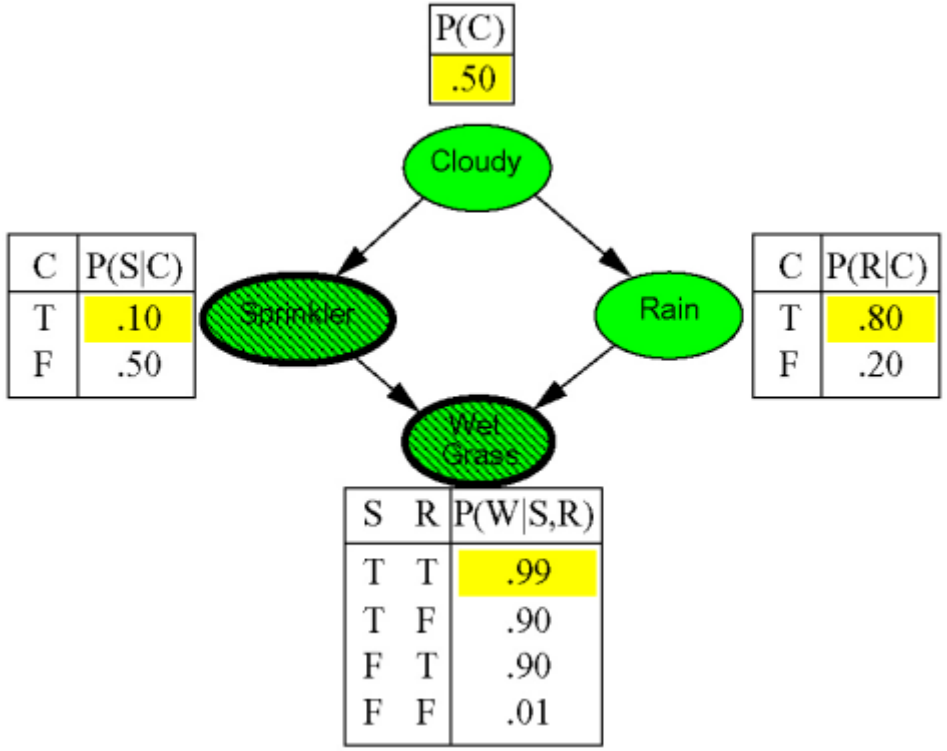
$$w = 1.0 \times 0.1$$

Likelihood weighting



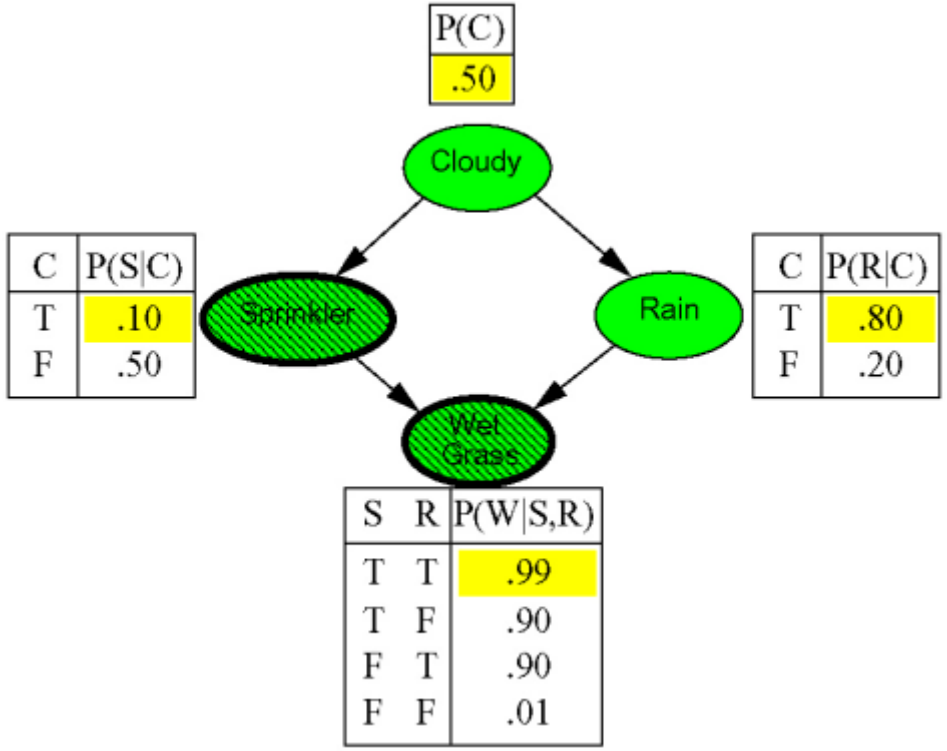
$$w = 1.0 \times 0.1$$

Likelihood weighting



$$w = 1.0 \times 0.1$$

Likelihood weighting



$$w = 1.0 \times 0.1 \times 0.99 = 0.099$$

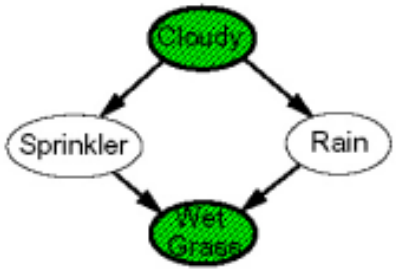
Likelihood weighting analysis

Sampling probability for WEIGHTEDSAMPLE is

$$S_{WS}(\mathbf{z}, \mathbf{e}) = \Pi_{i=1}^{l} P(z_i | Parents(Z_i))$$

Note: pays attention to evidence in **ancestors** only

$\Rightarrow$ somewhere "in between" prior and posterior distribution



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

$$w(\mathbf{z}, \mathbf{e}) = \Pi_{i=1}^{m} P(e_i | Parents(E_i))$$

Weighted sampling probability is

$$S_{WS}(\mathbf{z}, \mathbf{e}) w(\mathbf{z}, \mathbf{e})$$
$$= \Pi_{i=1}^{l} P(z_i | Parents(Z_i)) \ \Pi_{i=1}^{m} P(e_i | Parents(E_i))$$
$$= P(\mathbf{z}, \mathbf{e})$$ (by standard global semantics of network)

Hence likelihood weighting returns consistent estimates but performance still degrades with many evidence variables because a few samples have nearly all the total weight

Approximate inference using MCMC*

"State" of network = current assignment to all variables.

Generate next state by sampling one variable given Markov blanket
Sample each variable in turn, keeping evidence fixed

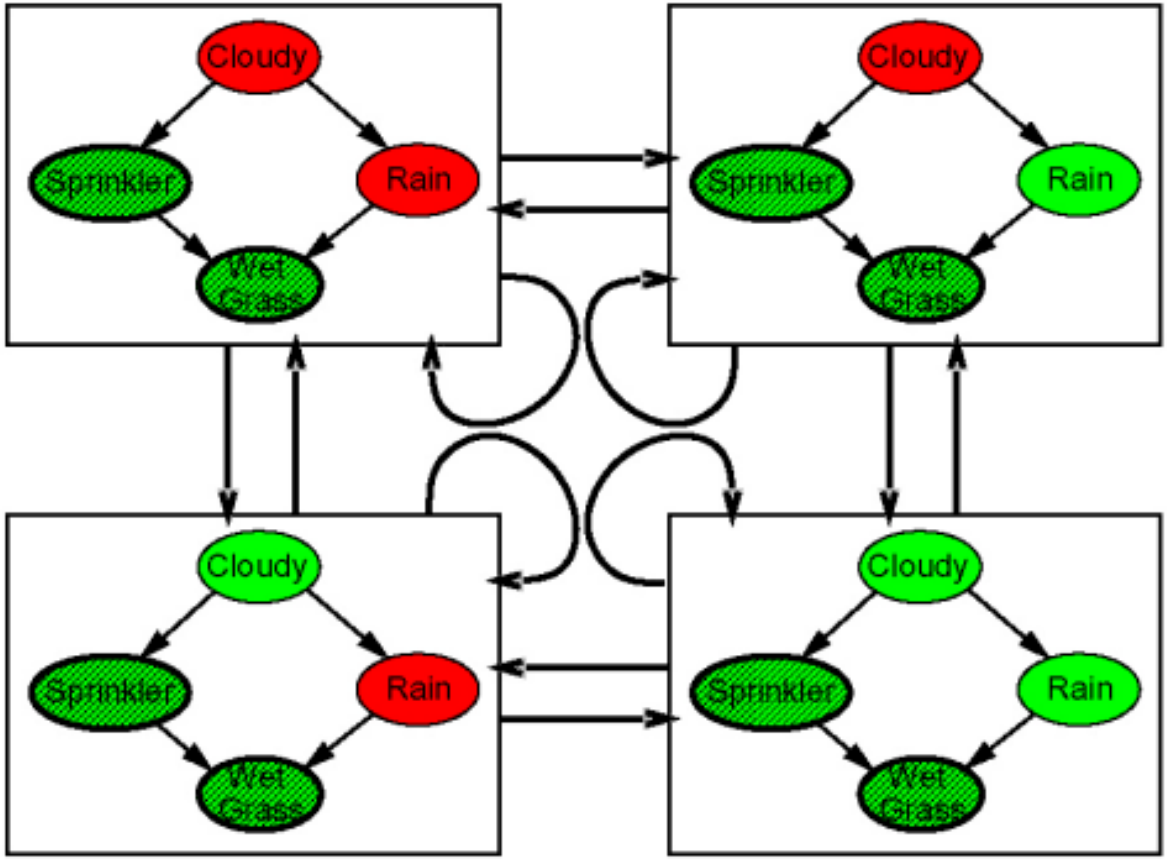
function MCMC-ASK($X, \mathbf{e}, bn, N$) **returns** an estimate of $P(X|\mathbf{e})$
  local variables: $\mathbf{N}[X]$, a vector of counts over X, initially zero
    $\mathbf{Z}$, the nonevidence variables in bn
    $\mathbf{x}$, the current state of the network, initially copied from $\mathbf{e}$

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

Can also choose a variable to sample at random each time

The Markov chain

With $Sprinkler = true, WetGrass = true$, there are four states:



Wander about for a while, average what you see

MCMC example

Estimate $\mathbf{P}(Rain|Sprinkler = true, WetGrass = true)$

Sample $Cloudy$ or $Rain$ given its Markov blanket, repeat.
Count number of times $Rain$ is true and false in the samples.

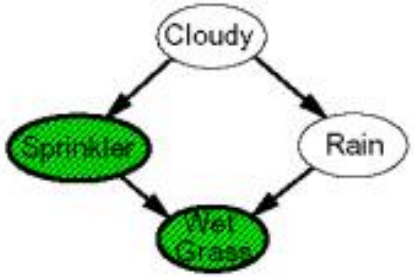
E.g., visit 100 states
31 have $Rain = true$, 69 have $Rain = false$

$\hat{\mathbf{P}}(Rain|Sprinkler = true, WetGrass = true)$
$= \text{NORMALIZE}(\langle 31, 69 \rangle) = \langle 0.31, 0.69 \rangle$

Theorem: chain approaches stationary distribution:
long-run fraction of time spent in each state is exactly
proportional to its posterior probability

Markov blanket sampling

Markov blanket of $Cloudy$ is
$Sprinkler$ and $Rain$
Markov blanket of $Rain$ is
$Cloudy$, $Sprinkler$, and $WetGrass$



Probability given the Markov blanket is calculated as follows:

$$P(x_i'|MB(X_i)) = P(x_i'|Parents(X_i))\Pi_{Z_j \in Children(X_i)} P(z_j|Parents(Z_j))$$

Easily implemented in message-passing parallel systems, brains

Main computational problems:

1) Difficult to tell if convergence has been achieved
2) Can be wasteful if Markov blanket is large:
   $P(X_i|MB(X_i))$ won't change much (law of large numbers)