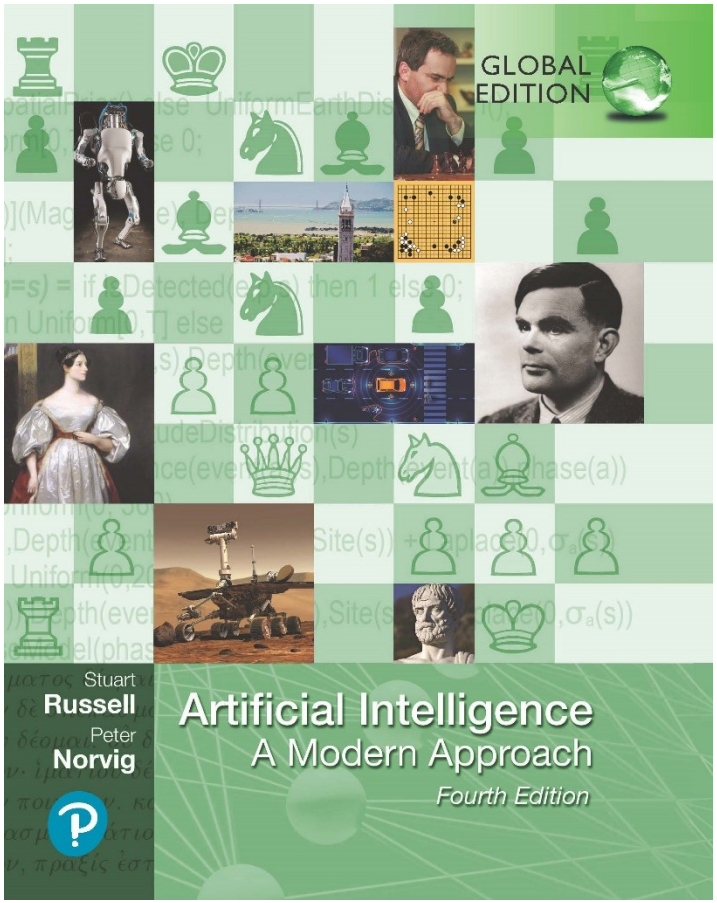
Artificial Intelligence: A Modern Approach

Fourth Edition, Global Edition



Chapter 12

Quantifying Uncertainty

Lecture Presentations: Artificial Intelligence

Adapted from:
"Artificial Intelligence: A Modern Approach, Global Edition", 4th Edition by Stuart Russell and Peter Norvig © 2021 Pearson Education.

Adapted for educational use at ACE Engineering College. Some slides customized by Mr. Shafakhatullah Khan Mohammed, Assistant Professor @ ACE Engineering College. For instructional use only. Not for commercial distribution.

Outline

- Acting Under Uncertainty
- Basic Probability Notation
- Inference Using Full Joint Distributions
- Independence
- Bayes' Rule and Its Use
- Naive Bayes Models
- The Wumpus World Revisited

Acting Under Uncertainty

- Real world problems contain **uncertainties due to:**
  - partial observability,
  - nondeterminism, or
  - adversaries.

- Example of dental diagnosis using propositional logic

$$Toothache \Rightarrow Cavity.$$

- However inaccurate, not all patients with toothaches have cavities

$$Toothache \Rightarrow Cavity \vee GumProblem \vee Abscess...$$

- In order to make the rule true, we have to add an almost unlimited list of possible problems.

- The only way to fix the rule is to make it logically exhaustive

Acting Under Uncertainty

- An agent strives to choose the right thing to do—the rational decision—depends on both the relative importance of various goals and the likelihood that, and degree to which, they will be achieved.

- Large domains such as medical diagnosis fail to three main reasons:
  - **Laziness:** It is too much work to list the complete set of antecedents or consequents needed to ensure an exceptionless rule
  - **Theoretical ignorance:** Medical science has no complete theory for the domain
  - **Practical ignorance:** Even if we know all the rules, we might be uncertain about a particular patient because not all the necessary tests have been or can be run.

- An agent only has a degree of belief in the relevant sentences.

Acting Under Uncertainty

- **Probability theory**
  - tool to deal with degrees of belief of relevant sentences.
  - summarizes the uncertainty that comes from our laziness and ignorance

- **Uncertainty and rational decisions**
  - An requires **preference** among **different possible outcomes** of various plans
  - **Utility Theory**: the quality of the outcome being useful
    - Every state has a degree of usefulness/utility
    - Higher utility is preferred
  - **Decision Theory:** Preferences (Utility Theory) combined with probabilities
    - *Decision theory = probability theory + utility theory.*
    - agent is **rational** if and only if it chooses the action that **yields the highest expected utility**, averaged over all the **possible outcomes** of the action.
    - principle of maximum expected utility (MEU).

Acting Under Uncertainty

- Function of a decision-theoretic agent that selects rational actions.

function DT-AGENT(*percept*) **returns** an *action*
persistent: *belief state*, probabilistic beliefs about the current state of the world
    action, the agent's action
update *belief state* based on *action* and *percept*
calculate outcome probabilities for actions,
    given action descriptions and current *belief state*
select *action* with highest expected utility
    given probabilities of outcomes and utility information
return *action*

Basic Probability Notation

- For our agent to represent and use probabilistic information, we need a formal language.

- **Sample space**: the set of all possible worlds
  - The possible worlds are mutually exclusive and exhaustive

- A fully specified probability model associates a numerical probability $P(\omega)$ with each possible world.
- The basic axioms of probability theory say that every possible world has a probability between 0 and 1 and that the total probability of the set of possible worlds is 1:

$$0 \leq P(\omega) \leq 1 \text{ for every } \omega \text{ and } \omega \in \Omega$$

- **Unconditional or prior probability**: degrees of belief in propositions in the absence of any other information

Basic Probability Notation

- **Conditional or posterior probability:** given evidence that has happened, degree of belief of new event
  - Make use of unconditional probabilities

- Probability of a given b:

$$P(a|b) = \frac{P(a \wedge b)}{P(b)}$$

- Can also written as:

$$P(a \wedge b) = P(a \mid b)P(b) \ .$$

- Example of rolling fair dice, rolling doubles when the first dice is 5

$$P(doubles|Die_1 = 5) = \frac{P(doubles \wedge Die_1 = 5)}{P(Die_1 = 5)} \ .$$

Basic Probability Notation

- **Factored representation:** possible world is represented by a set of variable/value pairs.
  - Variables in probability theory are called random variables, and their names begin with an uppercase letter. ($Total$ and Die_1)

- Sometimes we will want to talk about the probabilities of all the possible values of a random variable. We could write:

$$P(Weather = sun) = 0.6$$
$$P(Weather = rain) = 0.1$$
$$P(Weather = cloud) = 0.29$$
$$P(Weather = snow) = 0.01\ ,$$

- Abbreviation of this will be:

$$\mathbf{P}(Weather) = (0.6, 0.1, 0.29, 0.01),$$

- **P** statement defines a **probability distribution** for the random variable *Weather*

Inference Using Full Joint Distributions

Start with the joint distribution:

	toothache		*toothache*	
	catch	*catch*	*catch*	*catch*
cavity	**.108**	**.012**	**.072**	**.008**
cavity	**.016**	**.064**	**.144**	**.576**

For any proposition φ, sum the atomic events where it is true:

$$P(\varphi) = \Sigma_{\omega : \omega \models \varphi} P(\omega)$$

Inference Using Full Joint Distributions

Start with the joint distribution:

	toothache		*toothache*	
	catch	*catch*	*catch*	*catch*
cavity	**.108**	**.012**	**.072**	**.008**
cavity	**.016**	**.064**	**.144**	**.576**

For any proposition φ, sum the atomic events where it is true:

$$P(\varphi) = \Sigma_{\omega : \omega \models \varphi} P(\omega)$$

$$P(toothache) = 0.108 + 0.012 + 0.016 + 0.064 = 0.2$$

Inference Using Full Joint Distributions

Start with the joint distribution:

	toothache		*toothache*	
	catch	*catch*	*catch*	*catch*
cavity	**.108**	**.012**	**.072**	**.008**
cavity	**.016**	**.064**	**.144**	**.576**

For any proposition φ, sum the atomic events where it is true:

$$P(\varphi) = \Sigma_{\omega:\omega \models \varphi} P(\omega)$$

$$P(cavity \vee toothache) = 0.108+0.012+0.072+0.008+0.016+0.064 = 0.28$$

Inference Using Full Joint Distributions

Start with the joint distribution:

	toothache		*toothache*	
	catch	*catch*	*catch*	*catch*
cavity	**.108**	**.012**	**.072**	**.008**
cavity	**.016**	**.064**	**.144**	**.576**

Can also compute conditional probabilities:

$$P(\neg cavity \mid toothache) = \frac{P(\neg cavity \wedge toothache)}{P(toothache)}$$

$$= \frac{0.016 + 0.064}{0.108 + 0.012 + 0.016 + 0.064} = 0.4$$

Normalization

	toothache		*toothache*	
	catch	*catch*	*catch*	*catch*
cavity	.108	.012	.072	.008
cavity	.016	.064	.144	.576

Denominator can be viewed as a normalization constant α

$$\begin{aligned}
\mathbf{P}(Cavity|toothache) &= \alpha\,\mathbf{P}(Cavity, toothache) \\
&= \alpha\,[\mathbf{P}(Cavity, toothache, catch) + \mathbf{P}(Cavity, toothache, \neg catch)] \\
&= \alpha\,[(0.108, 0.016) + (0.012, 0.064)] \\
&= \alpha\,(0.12, 0.08) = (0.6, 0.4)
\end{aligned}$$

General idea: compute distribution on query variable
by fixing evidence variables and summing over hidden variables

Inference Using Full Joint Distributions

Let $\mathbf{X}$ be all the variables. Typically, we want
the posterior joint distribution of the query variables $\mathbf{Y}$
given specific values $\mathbf{e}$ for the evidence variables $\mathbf{E}$

Let the hidden variables be $\mathbf{H} = \mathbf{X} - \mathbf{Y} - \mathbf{E}$

Then the required summation of joint entries is done by summing out the hidden variables:

$$\mathbf{P}(\mathbf{Y} \mid \mathbf{E} = \mathbf{e}) = \alpha \mathbf{P}(\mathbf{Y}, \mathbf{E} = \mathbf{e}) = \alpha \Sigma_{\mathbf{h}} \mathbf{P}(\mathbf{Y}, \mathbf{E} = \mathbf{e}, \mathbf{H} = \mathbf{h})$$

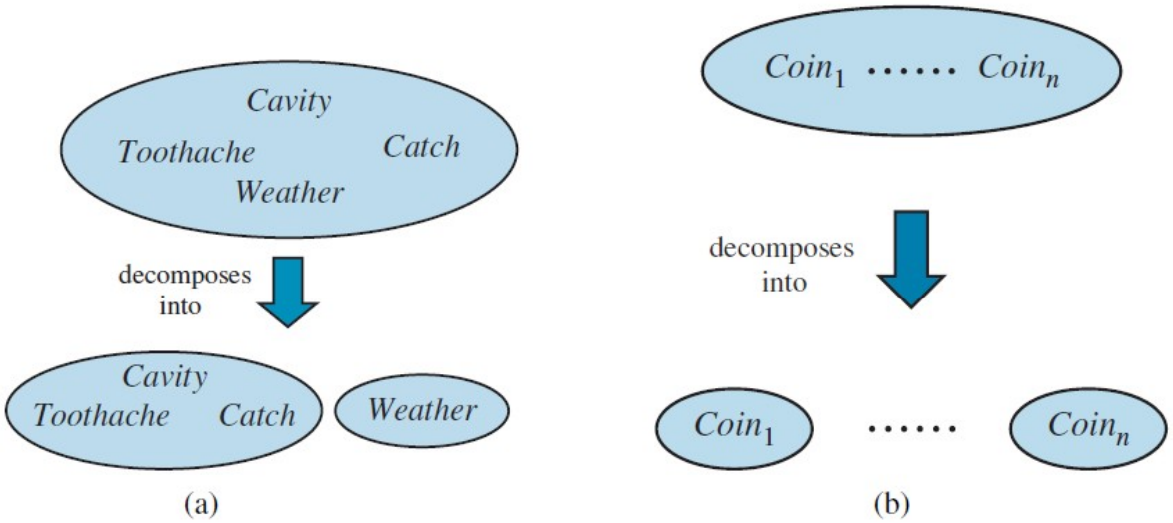
The terms in the summation are joint entries because $\mathbf{Y}$, $\mathbf{E}$, and $\mathbf{H}$ together exhaust the set of random variables

Obvious problems:

1) Worst-case time complexity $O(d^n)$ where d is the largest arity
2) Space complexity $O(d^n)$ to store the joint distribution
3) How to find the numbers for $O(d^n)$ entries???

Independence

- Two examples of factoring a large joint distribution into smaller distributions, using absolute independence. (a) Weather and dental problems are independent. (b) Coin flips are independent.



- $P(a \mid b) = P(a)$ or $P(b \mid a) = P(b)$ or $P(a \wedge b) = P(a)P(b)$.
- one's dental problems influence the weather thus:
- $P(toothache, catch, cavity, cloud) = P(cloud \mid toothache, catch\ cavity)\ P(toothache, catch, cavity)$.
- $P(cloud \mid toothache, catch, cavity) = P(cloud)$.
- $P(toothache, catch, cavity, cloud) = P(cloud)P(toothache, catch, cavity)$

Bayes' Rule and Its Use

- Bayes' rule is derived from the product rule
- $P(a \wedge b) = P(a \mid b)P(b)$ and $P(a \wedge b) = P(b \mid a)P(a)$.
- Equating the two right-hand sides and dividing by $P(a)$, we get

$$P(b|a) = \frac{P(a|b)P(b)}{P(a)} .$$

- Often, we perceive as evidence the effect of some unknown cause and we would like to determine that cause. In that case, Bayes' rule becomes

$$P(cause|effect) = \frac{P(effect|cause)P(cause)}{P(effect)}$$

- The conditional probability $P(effect|cause)$ quantifies the relationship in the **causal** direction, whereas $P(cause|effect)$ describes the **diagnostic** direction.

Bayes' Rule and Its Use

- For example, a doctor knows that the disease meningitis causes a patient to have a stiff neck, say, 70% of the time. The doctor also knows some unconditional facts: the prior probability that any patient has meningitis is 1/50,000, and the prior probability that any patient has a stiff neck is 1%. Letting *s* be the proposition that the patient has a stiff neck and *m* be the proposition that the patient has meningitis, we have

$$P(s|m) = 0.7$$
$$P(m) = 1/50000$$
$$P(s) = 0.01$$

$$P(m|s) = \frac{P(s|m)P(m)}{P(s)} = \frac{0.7 \times 1/50000}{0.01} = 0.0014$$

- That is, we expect only 0.14% of patients with a stiff neck to have meningitis. Notice that even though a stiff neck is quite strongly indicated by meningitis (with probability 0.7), the probability of meningitis in patients with stiff necks remains small. This is because the prior probability of stiff necks (from any cause) is much higher than the prior for meningitis.

Bayes' Rule and conditional independence

$$\mathrm{P}(Cavity|toothache \wedge catch)$$
$$= a\,\mathrm{P}(toothache \wedge catch|Cavity)\mathrm{P}(Cavity)$$
$$= a\,\mathrm{P}(toothache|Cavity)\mathrm{P}(catch|Cavity)\mathrm{P}(Cavity)$$

This is an example of a naive Bayes model:

$$\mathrm{P}(Cause, Effect_1, \ldots, Effect_n) = \mathrm{P}(Cause)\Pi_i\mathrm{P}(Effect_i|Cause)$$

Total number of parameters is linear in n

Naïve Bayes Models

- The full joint distribution can be written as

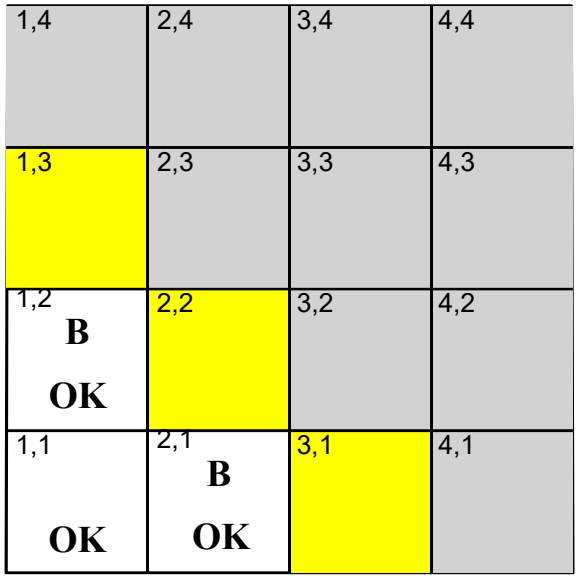
$$\mathbf{P}(Cause, Effect_1, \ldots, Effect_n) = \mathbf{P}(Cause) \prod_i \mathbf{P}(Effect_i \mid Cause)$$

- Such a probability distribution is called a naive Bayes model—"naive" because it is often used (as a simplifying assumption) in cases where the "effect" variables are not strictly independent given the cause variable.

- Call the observed effects **E**=**e**, while the remaining effect variables **Y** are unobserved

$$\begin{aligned}
\mathbf{P}(Cause \mid \mathbf{e}) &= \alpha \sum_{\mathbf{y}} \mathbf{P}(Cause)\mathbf{P}(\mathbf{y} \mid Cause)\left(\prod_j \mathbf{P}(e_j \mid Cause)\right) \\
&= \alpha \mathbf{P}(Cause)\left(\prod_j \mathbf{P}(e_j \mid Cause)\right)\sum_{\mathbf{y}} \mathbf{P}(\mathbf{y} \mid Cause) \\
&= \alpha \mathbf{P}(Cause)\prod_j \mathbf{P}(e_j \mid Cause)
\end{aligned}$$

The Wumpus World Revisited



$P_{ij} = true$ iff $[i, j]$ contains a pit

$B_{ij} = true$ iff $[i, j]$ is breezy

Include only $B_{1,1}, B_{1,2}, B_{2,1}$ in the probability model

Specifying the probability model

The full joint distribution is $\mathrm{P}(P_{1,1}, \ldots, P_{4,4}, B_{1,1}, B_{1,2}, B_{2,1})$

Apply product rule: $\mathrm{P}(B_{1,1}, B_{1,2}, B_{2,1} \mid P_{1,1}, \ldots, P_{4,4})\mathrm{P}(P_{1,1}, \ldots, P_{4,4})$

(Do it this way to get $P(Effect|Cause)$.)

First term: 1 if pits are adjacent to breezes, 0 otherwise

Second term: pits are placed randomly, probability 0.2 per square:

$$\mathrm{P}(P_{1,1}, \ldots, P_{4,4}) = \Pi_{i,j=1,1}^{4,4}\mathrm{P}(P_{i,j}) = 0.2^n \times 0.8^{16-n}$$

for n pits.

Observations and query

We know the following facts:

$$b = \neg b_{1,1} \wedge b_{1,2} \wedge b_{2,1}$$

$$known = \neg p_{1,1} \wedge \neg p_{1,2} \wedge \neg p_{2,1}$$

Query is $\mathrm{P}(P_{1,3} | known, b)$

Define $Unknown$ = P_{ij}s other than $P_{1,3}$ and $Known$

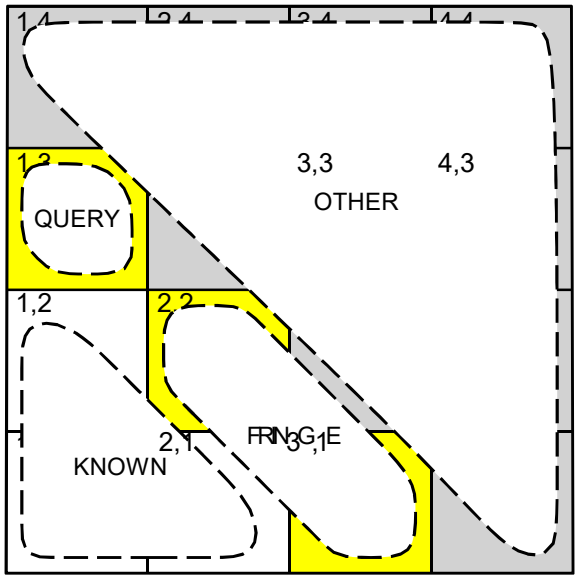
For inference by enumeration, we have

$$\mathrm{P}(P_{1,3} | known, b) = \alpha \Sigma_{unknown} \mathrm{P}(P_{1,3}, unknown, known, b)$$

Grows exponentially with number of squares!

Using conditional independence

Basic insight: observations are conditionally independent of other hidden squares given neighbouring hidden squares



Define $Unknown = Fringe \cup Other$

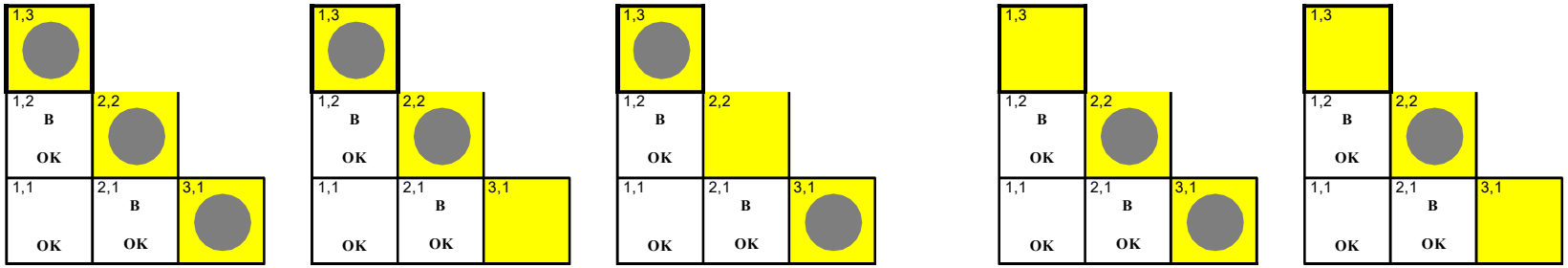
$\mathrm{P}(b|P_{1,3}, Known, Unknown) = \mathrm{P}(b|P_{1,3}, Known, Fringe)$

Manipulate query into a form where we can use this!

Using conditional independence contd.

$$
\begin{aligned}
\mathrm{P}(P_{1,3}|known, b) &= \alpha_{unknown} \mathrm{P}(P_{1,3}, unknown, known, b) \\
&= \alpha_{unknown} \mathrm{P}(b|P_{1,3}, known, unknown)\mathrm{P}(P_{1,3}, known, unknown) \\
&= \alpha_{fringe\,other} \mathrm{P}(b|known, P_{1,3}, fringe, other)\mathrm{P}(P_{1,3}, known, fringe, other) \\
&= \alpha_{fringe\,other} \mathrm{P}(b|known, P_{1,3}, fringe)\mathrm{P}(P_{1,3}, known, fringe, other) \\
&= \alpha_{fringe} \mathrm{P}(b|known, P_{1,3}, fringe) \sum_{other} \mathrm{P}(P_{1,3}, known, fringe, other) \\
&= \alpha_{fringe} \mathrm{P}(b|known, P_{1,3}, fringe) \sum_{other} \mathrm{P}(P_{1,3})P(known)P(fringe)P(other) \\
&= \alpha P(known)\mathrm{P}(P_{1,3}) \sum_{fringe} \mathrm{P}(b|known, P_{1,3}, fringe)P(fringe) \sum_{other} P(other) \\
&= \alpha\, \mathrm{P}(P_{1,3}) \sum_{fringe} \mathrm{P}(b|known, P_{1,3}, fringe)P(fringe)
\end{aligned}
$$

Using conditional independence contd.



$$\mathbf{P}(P_{1,3}|known, b) = \alpha\,(0.2(0.04 + 0.16 + 0.16),\ 0.8(0.04 + 0.16)) \approx (0.31, 0.69)$$

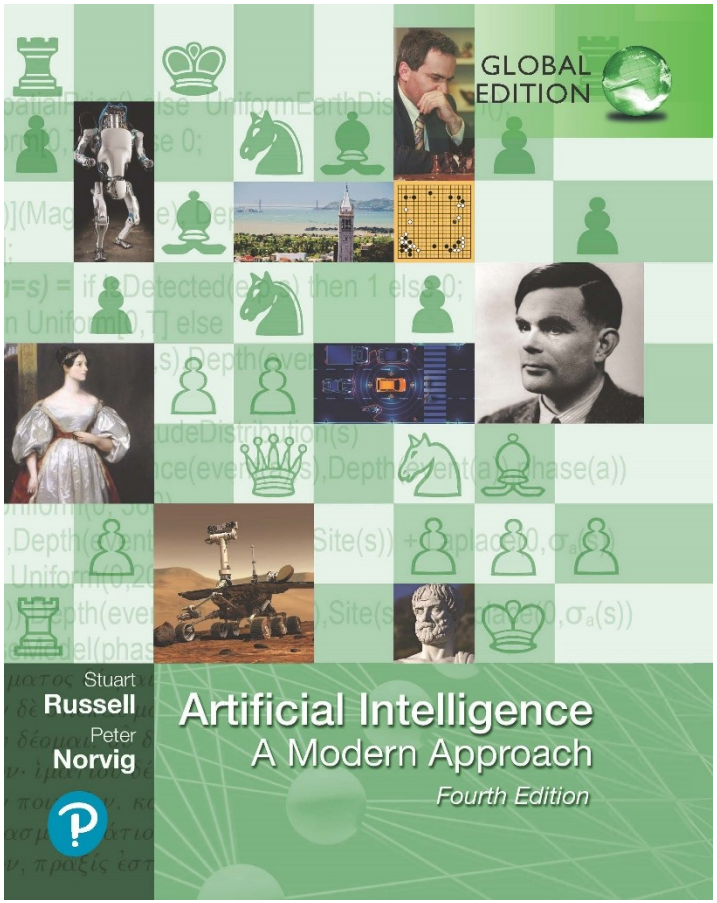
$$\mathbf{P}(P_{2,2}|known, b) \approx (0.86, 0.14)$$

Summary

- **Probabilities** express the agent's inability to reach a definite decision regarding the truth of a sentence.
- **Decision theory** combines the agent's beliefs and desires, defining the best action as the one that maximizes expected utility.
- Basic probability statements include **prior or unconditional probabilities** and **posterior or conditional probabilities** over simple and complex propositions.
- The axioms of probability constrain the probabilities of logically related propositions.
- The **full joint probability distribution** specifies the probability of each complete assignment of values to random variables
- **Absolute independence** between subsets of random variables allows the full joint distribution to be factored into smaller joint distributions, greatly reducing its complexity.
- **Bayes' rule** allows unknown probabilities to be computed from known conditional probabilities, usually in the causal direction.
- **Conditional independence** brought about by direct causal relationships in the domain allows the full joint distribution to be factored into smaller, conditional distributions.

Artificial Intelligence: A Modern Approach

Fourth Edition, Global Edition



Chapter 13

Probabilistic Reasoning

Lecture Presentation: Artificial Intelligence

Adapted from:
"Artificial Intelligence: A Modern Approach, Global Edition", 4th Edition by Stuart Russell and Peter Norvig © 2021 Pearson Education.

Adapted for educational use at ACE Engineering College. Some slides customized by Mr. Shafakhatullah Khan Mohammed, Assistant Professor @ ACE Engineering College. For instructional use only. Not for commercial distribution.

Outline

- Representing Knowledge in an Uncertain Domain
- Semantics of Bayesian Networks
- Exact Inference in Bayesian Networks
- Approximate Inference for Bayesian Networks
- Causal Networks

Representing Knowledge in an Uncertain Domain

Bayesian networks: represents dependencies among variables.

A simple, directed graph in which each node is annotated with quantitative probability information

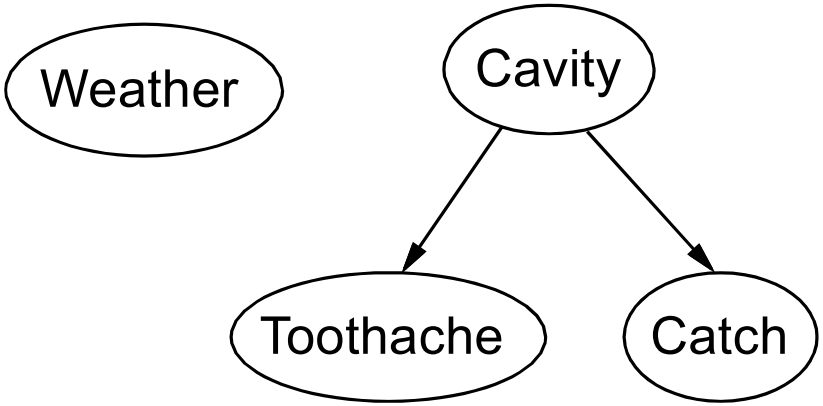
Syntax:

- a set of nodes, one per variable
- a directed, acyclic graph (link ≈ "directly influences")
- a conditional distribution for each node given its parents:
  $\mathrm{P}(X_i|Parents(X_i))$

In the simplest case, conditional distribution represented as a conditional probability table (CPT) giving the distribution over X_i for each combination of parent values

Example

Topology of network encodes conditional independence assertions:



Weather is independent of the other variables

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

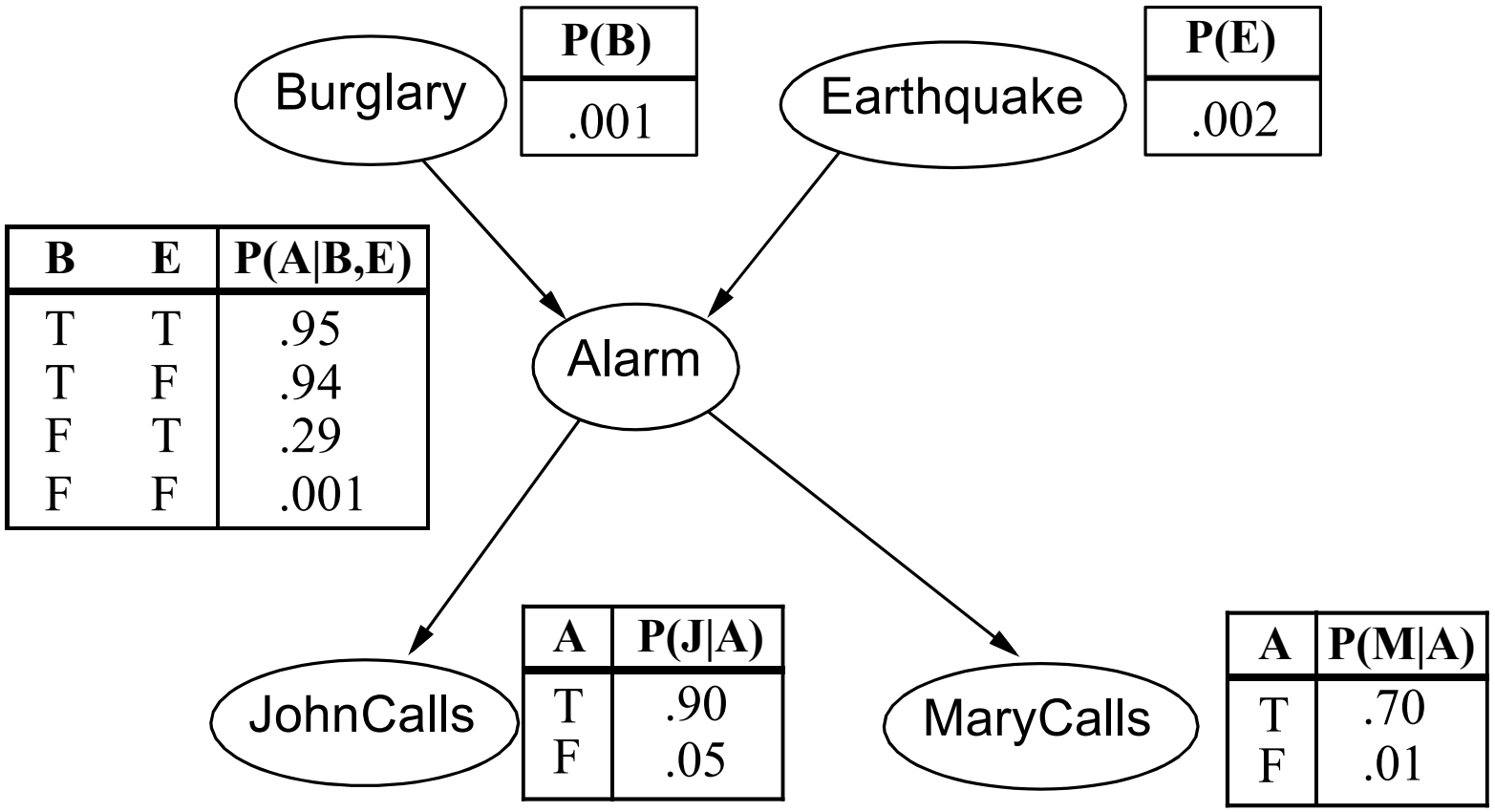
Example

I'm at work, neighbor John calls to say my alarm is ringing, but neighbor Mary doesn't call. Sometimes it's set off by minor earthquakes. Is there a burglar?

Variables: *Burglar*, *Earthquake*, *Alarm*, *JohnCalls*, *MaryCalls*
Network topology reflects "causal" knowledge:

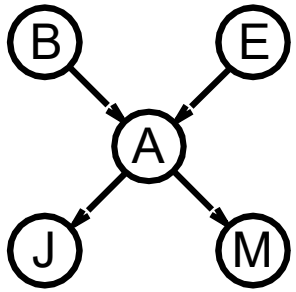
- A burglar can set the alarm off
- An earthquake can set the alarm off
- The alarm can cause Mary to call
- The alarm can cause John to call

Example contd.



Compactness

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 $O(n \cdot 2^k)$ numbers

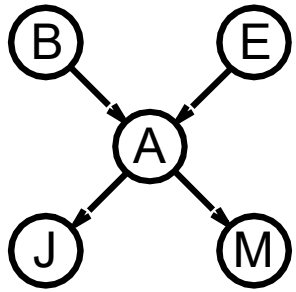
I.e., grows linearly with n, vs. $O(2^n)$ for the full joint distribution

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

The Semantics of Bayesian Networks

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

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



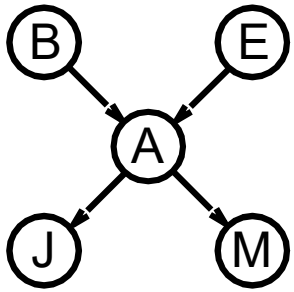
e.g., $P(j \wedge m \wedge a \wedge \neg b \wedge \neg e)$

$=$

Global semantics

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

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

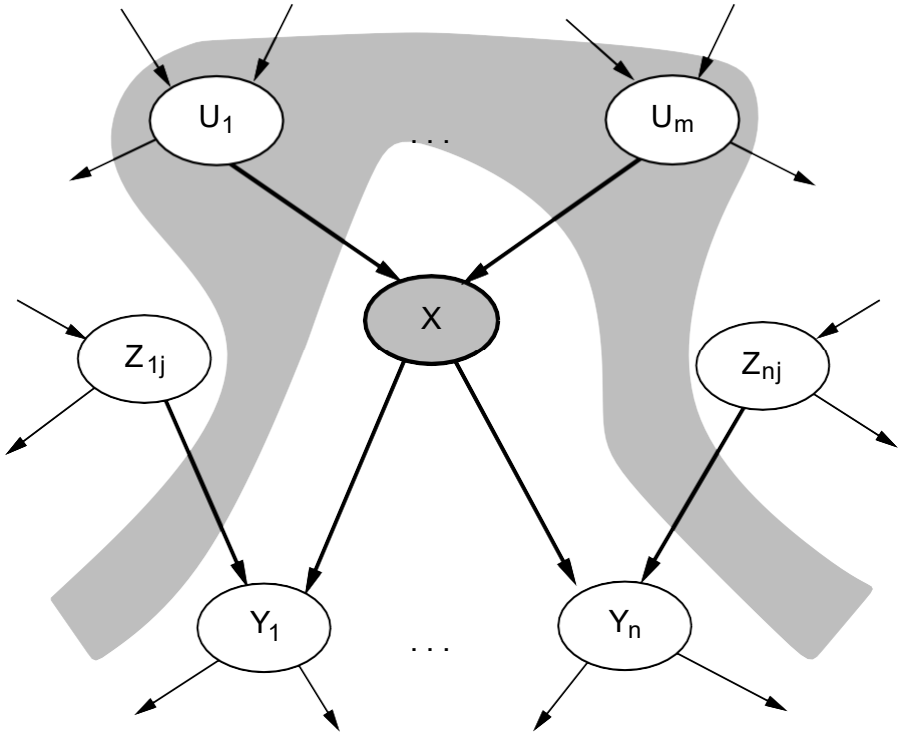


e.g., $P(j \wedge m \wedge a \wedge \neg b \wedge \neg e)$

$$= P(j|a)P(m|a)P(a|\neg b, \neg e)P(\neg b)P(\neg e)$$
$$= 0.9 \times 0.7 \times 0.001 \times 0.999 \times 0.998$$
$$\approx 0.00063$$

Local semantics

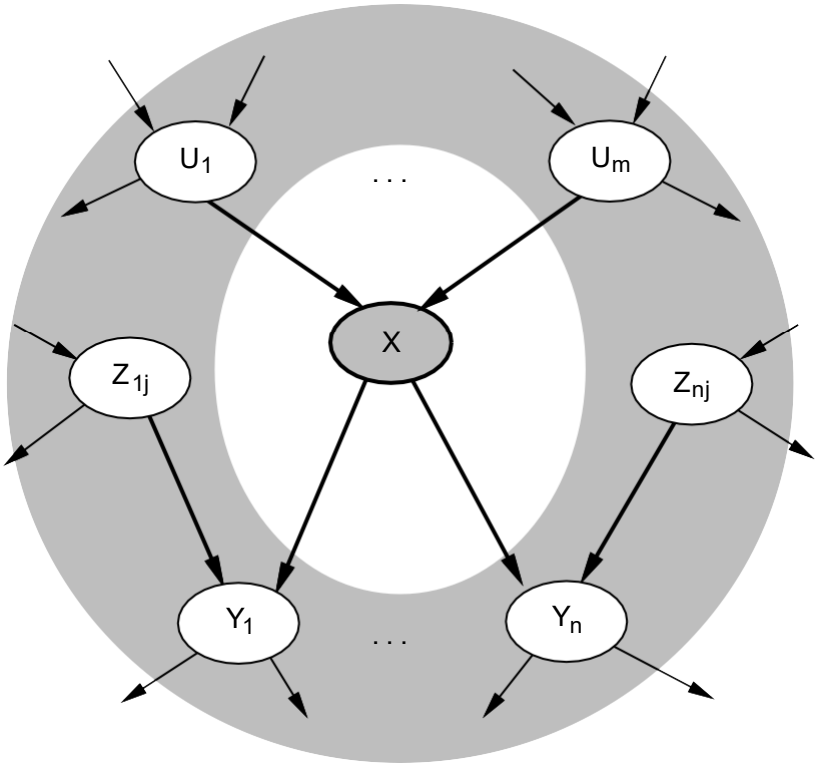
Local semantics: each node is conditionally independent of its nondescendants given its parents



Theorem: Local semantics ⇔ global semantics

Markov blanket

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



Constructing Bayesian networks

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

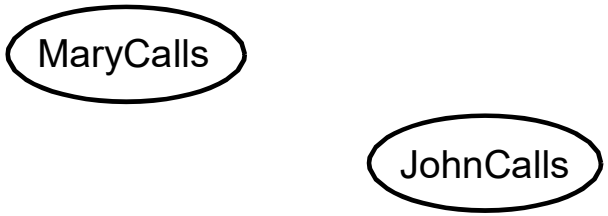
1. Choose an ordering of variables $X_1, \ldots, X_n$
2. For $i = 1$ to n
   add X_i to the network
   select parents from $X_1, \ldots, X_{i-1}$ such that
   $$\mathrm{P}(X_i | Parents(X_i)) = \mathrm{P}(X_i | X_1, \ldots, X_{i-1})$$

This choice of parents guarantees the global semantics:

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

Example

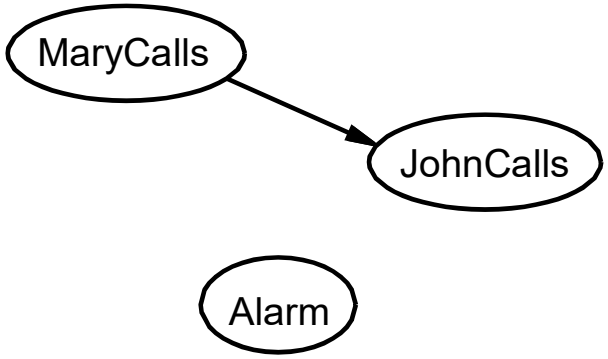
Suppose we choose the ordering M, J, A, B, E



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

Example

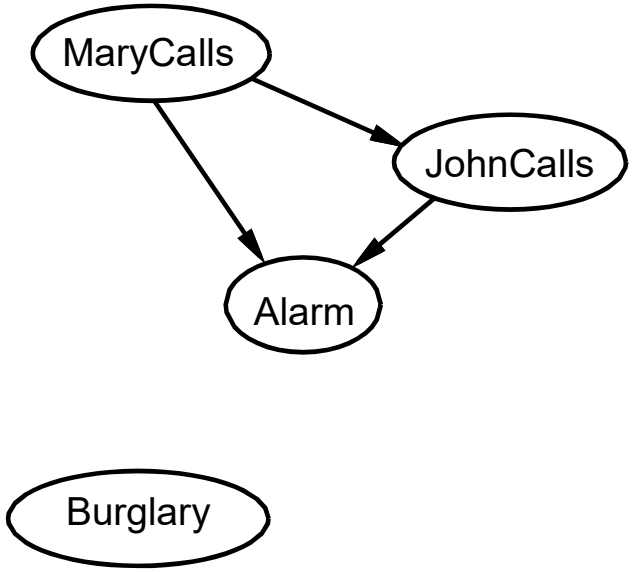
Suppose we choose the ordering M, J, A, B, E



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

Example

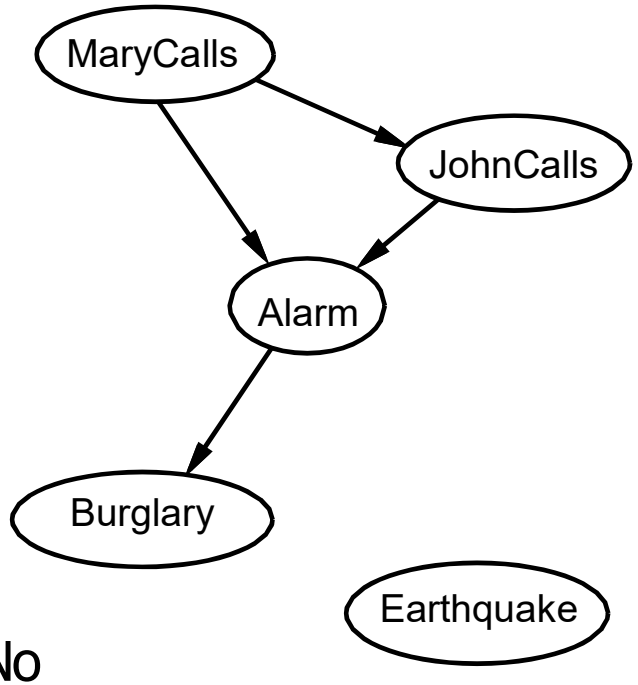
Suppose we choose the ordering M, J, A, B, E



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

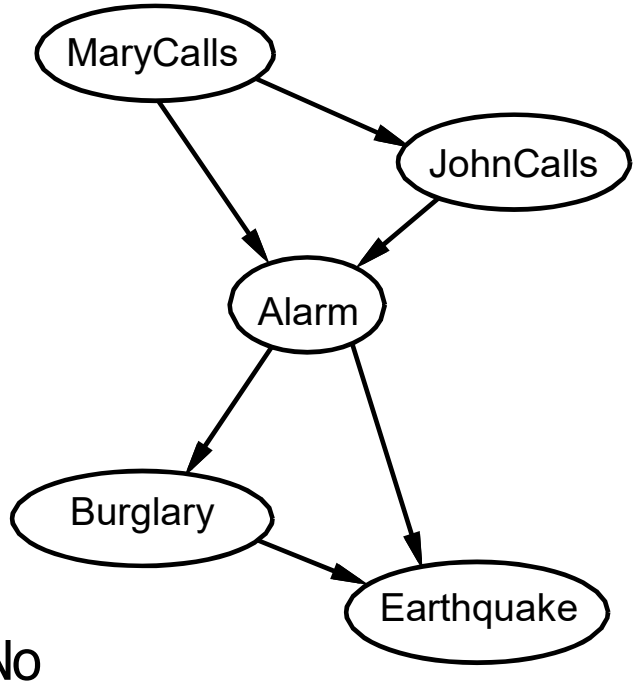
$P(B|A, J, M) = P(B)$? No

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

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

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

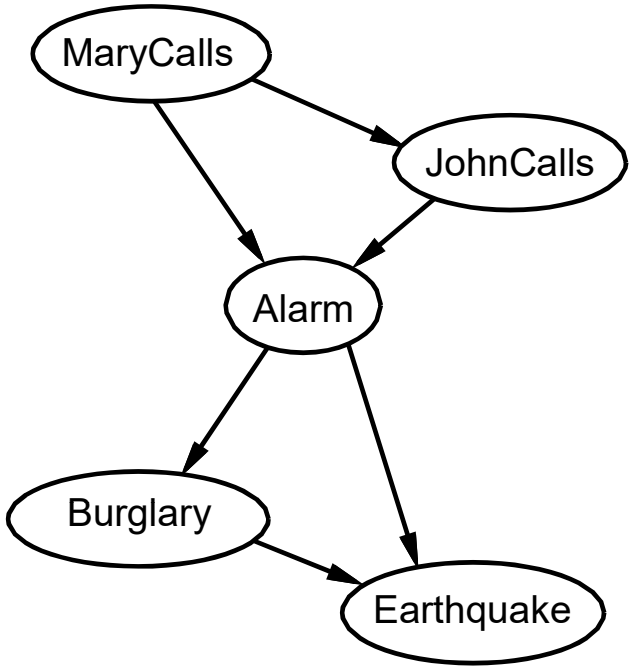
$P(B|A, J, M) = P(B|A)$? Yes

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

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

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

Example contd.



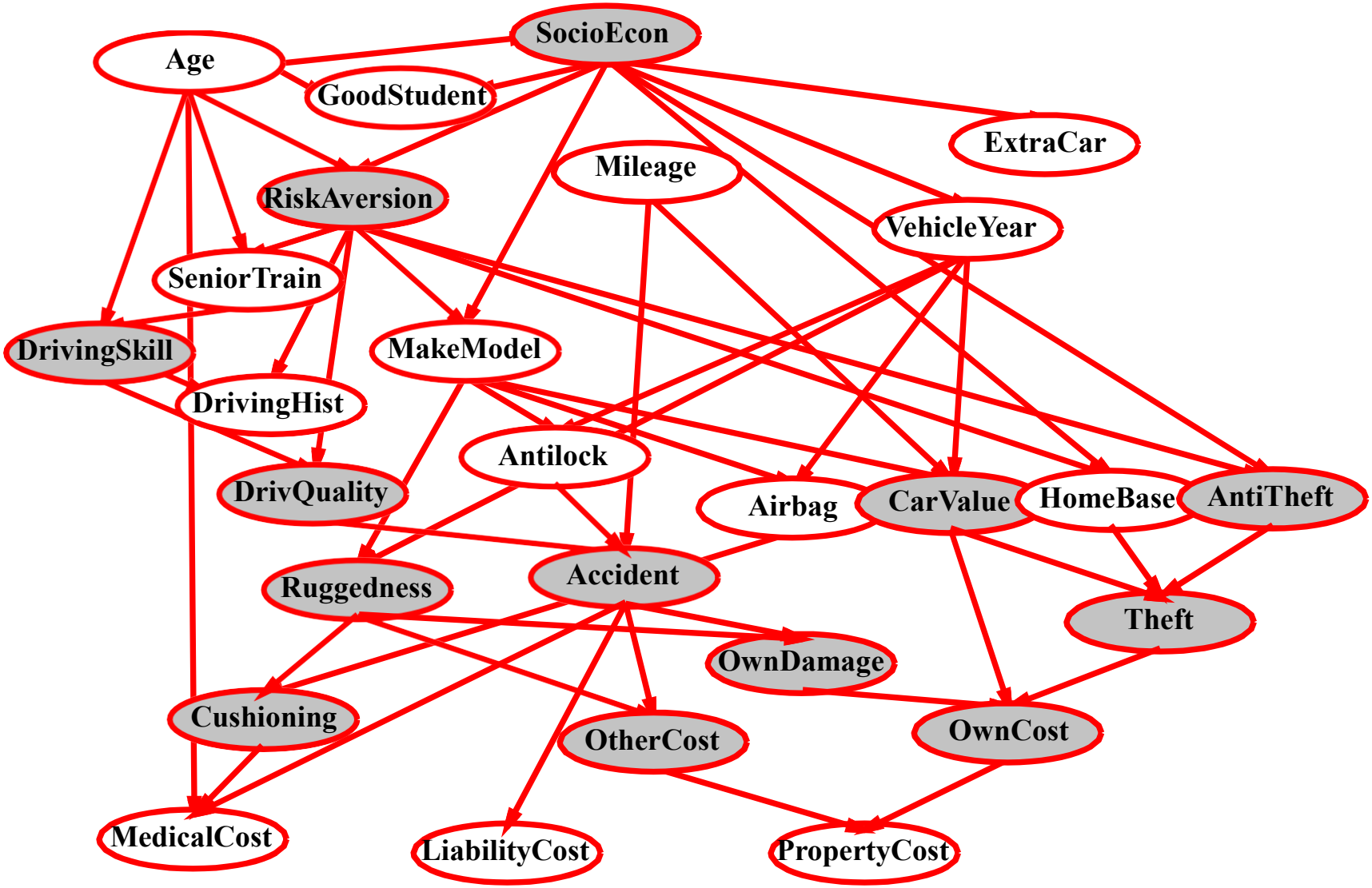
Deciding conditional independence is hard in noncausal directions

(Causal models and conditional independence seem hardwired for humans!)

Assessing conditional probabilities is hard in noncausal directions

Network is less compact: $1 + 2 + 4 + 2 + 4 = 13$ numbers needed

Example: Car insurance



Compact conditional distributions

CPT grows exponentially with number of parents
CPT becomes infinite with continuous-valued parent or child

Solution: canonical distributions that are defined compactly

Deterministic nodes are the simplest case:

$X = f(Parents(X))$ for some function f

E.g., Boolean functions

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

E.g., numerical relationships among continuous variables

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

Compact conditional distributions contd.

Noisy-OR distributions model multiple noninteracting causes

1) Parents $U_1 \ldots U_k$ include all causes (can add leak node)
2) Independent failure probability q_i for each cause alone

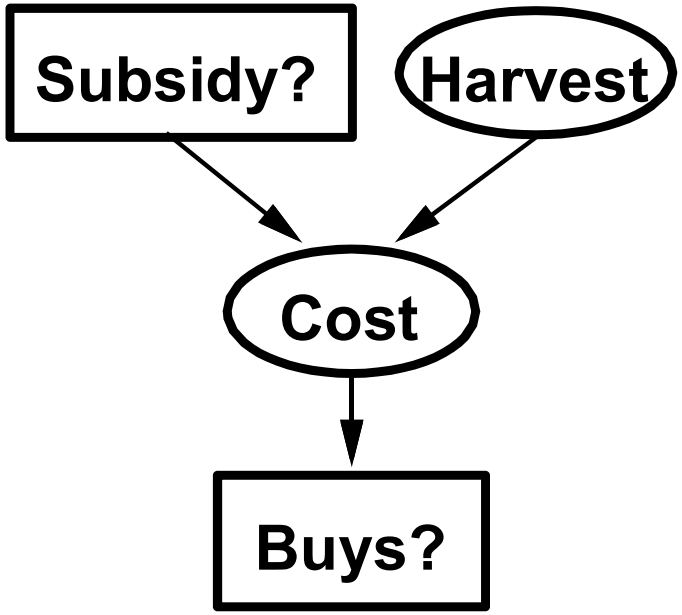
$$\Rightarrow P(X|U_1 \ldots U_j, \neg U_{j+1} \ldots \neg U_k) = 1 - \prod_{i=1}^{j} q_i$$

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	$0.012 = 0.6 \times 0.2 \times 0.1$

Number of parameters linear in number of parents

Hybrid (discrete+continuous) networks

Discrete (*Subsidy?* and *Buys?*); continuous (*Harvest* and *Cost*)



Option 1: discretization—possibly large errors, large CPTs
Option 2: finitely parameterized canonical families

1) Continuous variable, discrete+continuous parents (e.g., *Cost*)
2) Discrete variable, 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

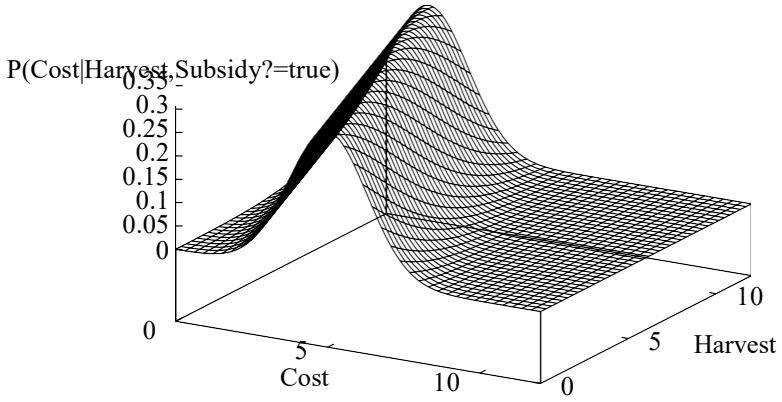
Most common is the linear Gaussian model, e.g.,:

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

Mean *Cost* varies linearly with *Harvest*, variance is fixed

Linear variation is unreasonable over the full range
but works OK if the likely range of *Harvest* is narrow

Continuous child variables



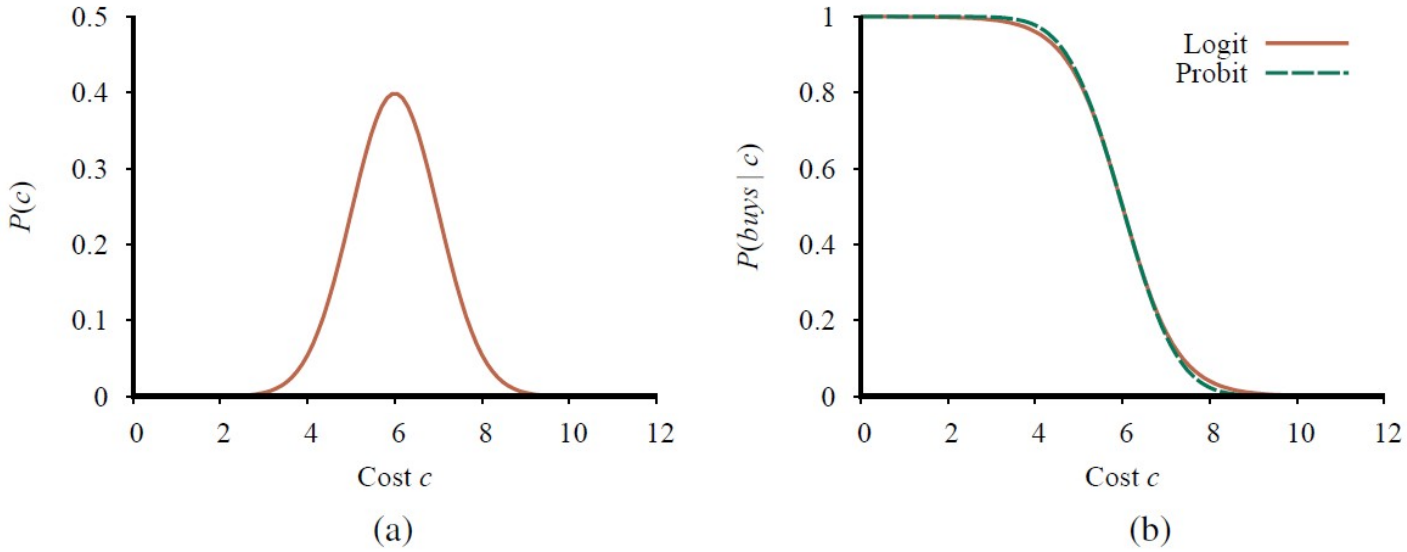
All-continuous network with LG distributions

⇒ full joint distribution is a multivariate Gaussian

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

Discrete variable w/ continuous parents

Probability of *Buys?* given *Cost* should be a "soft" threshold:



(a) A normal (Gaussian) distribution for the cost threshold, centered on $\mu = 6.0$ with standard deviation $\sigma = 1.0$. (b) Expit and probit models for the probability of *buys* given *cost*, for the parameters $\mu = 6.0$ and $\sigma = 1.0$.

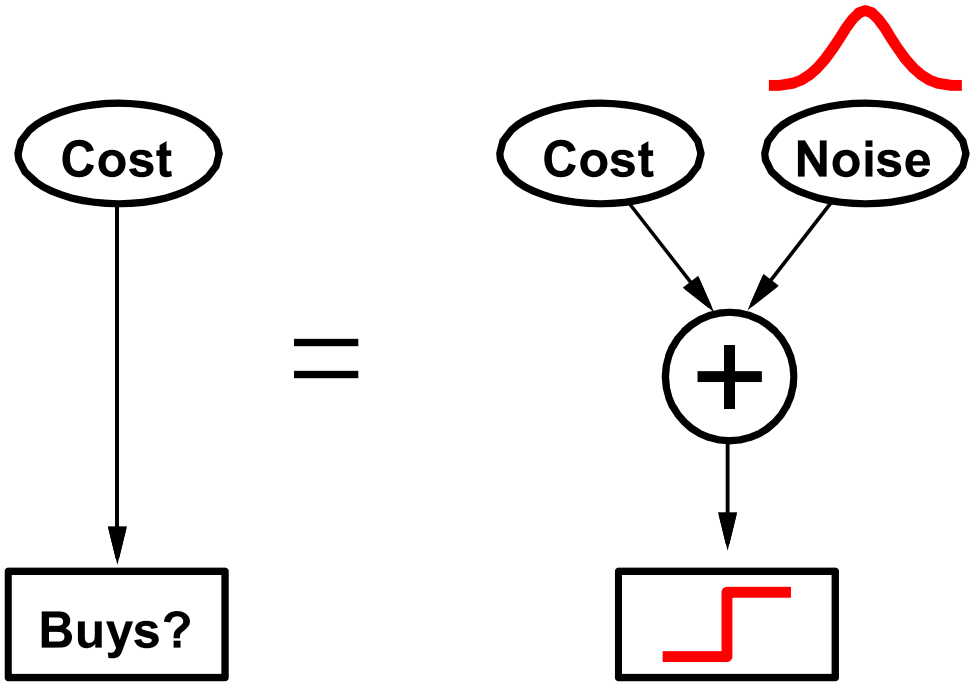
Probit distribution uses integral of Gaussian:

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

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

Why the probit?

1. It's sort of the right shape
2. Can view as hard threshold whose location is subject to noise

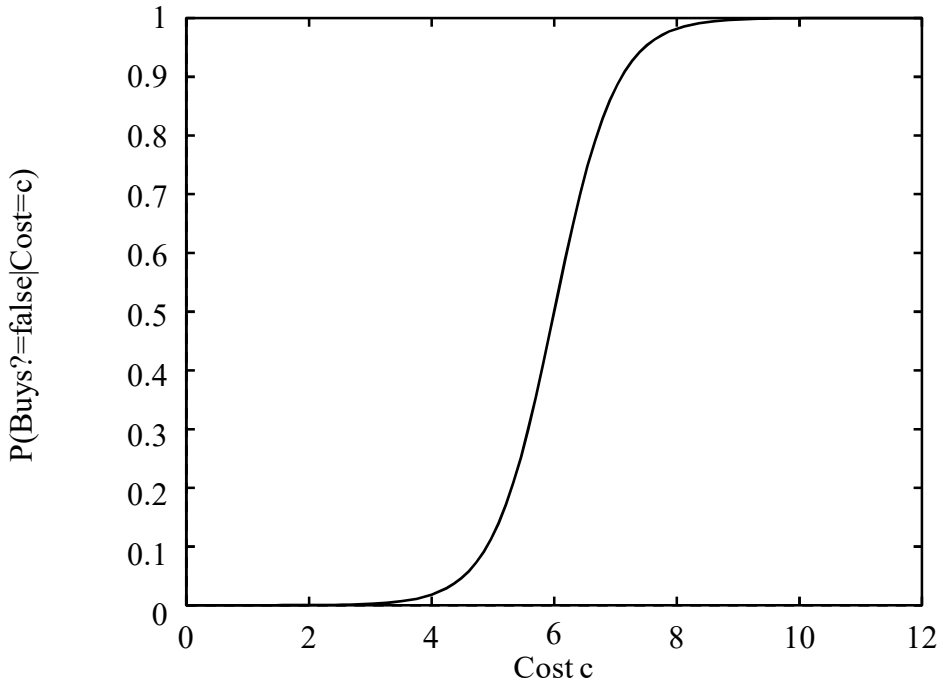


Discrete variable contd.

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 in Bayesian Networks

Simple queries: compute posterior marginal $\mathrm{P}(X_i|\mathrm{E}=\mathrm{e})$
e.g., $P(NoGas|Gauge=empty, Lights=on, Starts=false)$

Conjunctive queries: $\mathrm{P}(X_i, X_j|\mathrm{E}=\mathrm{e}) = \mathrm{P}(X_i|\mathrm{E}=\mathrm{e})\mathrm{P}(X_j|X_i, \mathrm{E}=\mathrm{e})$

Optimal decisions: decision networks include utility information;
probabilistic inference required for $P(outcome|action, evidence)$

Value of information: which evidence to seek next?

Sensitivity analysis: which probability values are most critical?

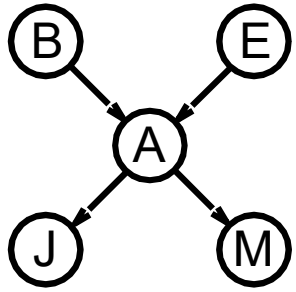
Explanation: why do I need a new starter motor?

Inference by enumeration

Slightly intelligent way to sum out variables from the joint without actually constructing its explicit representation

Simple query on the burglary network:

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



Rewrite full joint entries using product of CPT entries:

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

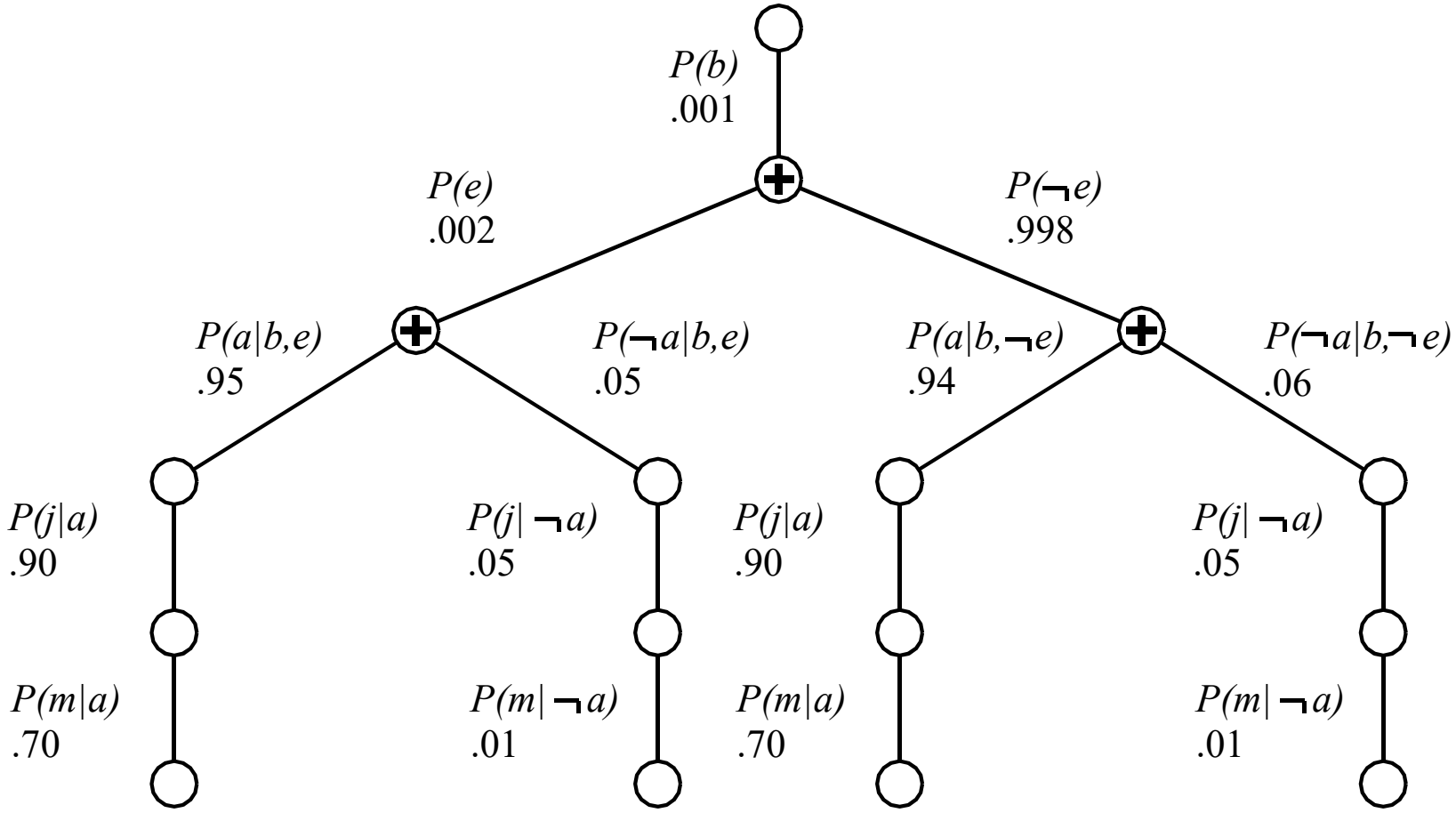
Recursive depth-first enumeration: $O(n)$ space, $O(d^n)$ time

Enumeration algorithm

```
function Enumeration-Ask(X, e, bn) returns a distribution over X
  inputs: X,  the query variable
          e, observed values for variables E
          bn, a Bayesian network with variables {X} ∪ E ∪ Y

  Q(X) ← a distribution over X, initially empty
  for each value x_i of X do
      extend e with value x_i for X
      Q(x_i) ← Enumerate-All(Vars[bn], e)
  return Normalize(Q(X))
```

```
function Enumerate-All(vars, e) returns a real number
  if Empty?(vars) then return 1.0
  Y ← First(vars)
  if Y has value y in e
      then return P(y | Pa(Y)) × Enumerate-All(Rest(vars), e)
      else return Σ_y P(y | Pa(Y)) × Enumerate-All(Rest(vars), e_y)
          where e_y is e extended with Y = y
```

Evaluation tree



Enumeration is inefficient: repeated computation

e.g., computes $P(j|a)P(m|a)$ for each value of e

Inference by variable elimination

Variable elimination: carry out summations right-to-left, storing intermediate results (factors) to avoid recomputation

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

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

Variable elimination algorithm

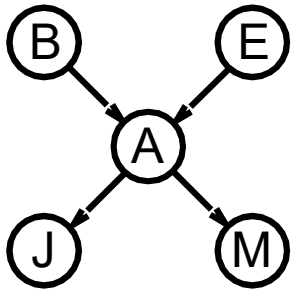
```
function Elimination-Ask(X, e, bn) returns a distribution over X
    inputs: X, the query variable
            e, evidence specified as an event
            bn, a belief network specifying joint distribution P(X_1, ..., X_n)

    factors ← []; vars ← Reverse(Vars[bn])
    for each var in vars do
        factors ← [Make-Factor(var, e)|factors]
        if var is a hidden variable then factors ← Sum-Out(var, factors)
    return Normalize(Pointwise-Product(factors))
```

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



Thm 1: 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 $MaryCalls$ is irrelevant

(Compare this to backward chaining from the query in Horn clause KBs)

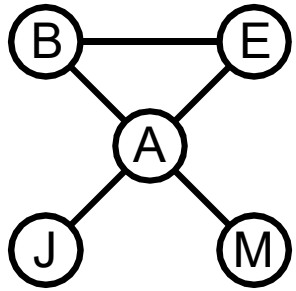
Irrelevant variables contd.

Defn: moral graph of Bayes net: marry all parents and drop arrows

Defn: A is m-separated from B by C iff separated by C in the moral graph

Thm 2: Y is irrelevant if m-separated from X by E

For $P(JohnCalls|Alarm = true)$, both $Burglary$ and $Earthquake$ are irrelevant

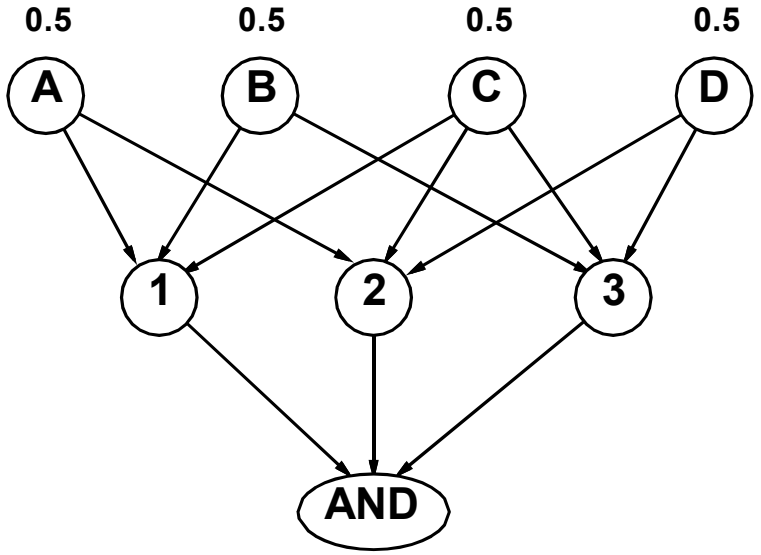


Complexity of exact inference

Singly connected networks (or polytrees):
- any two nodes are connected by at most one (undirected) path
- time and space cost of variable elimination are $O(d^k n)$

Multiply connected networks:
- can reduce 3SAT to exact inference ⇒ NP-hard
- equivalent to counting 3SAT models ⇒ #P-complete

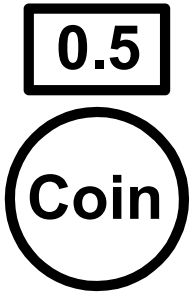
1. A v B v C
2. C v D v A
3. B v C v D



Approximate Inference for Bayesian Networks

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



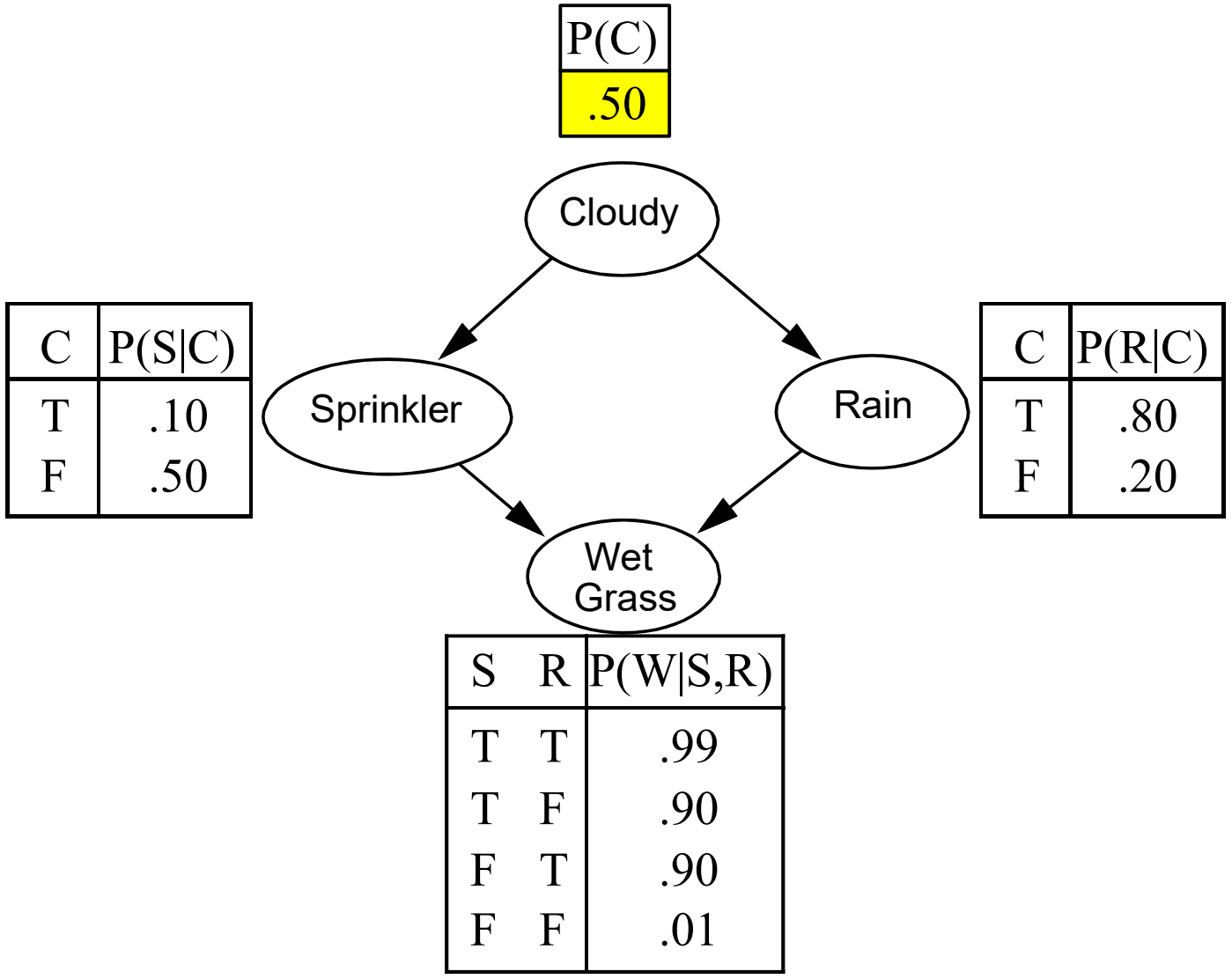
Outline:

- Sampling from an empty network
- 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

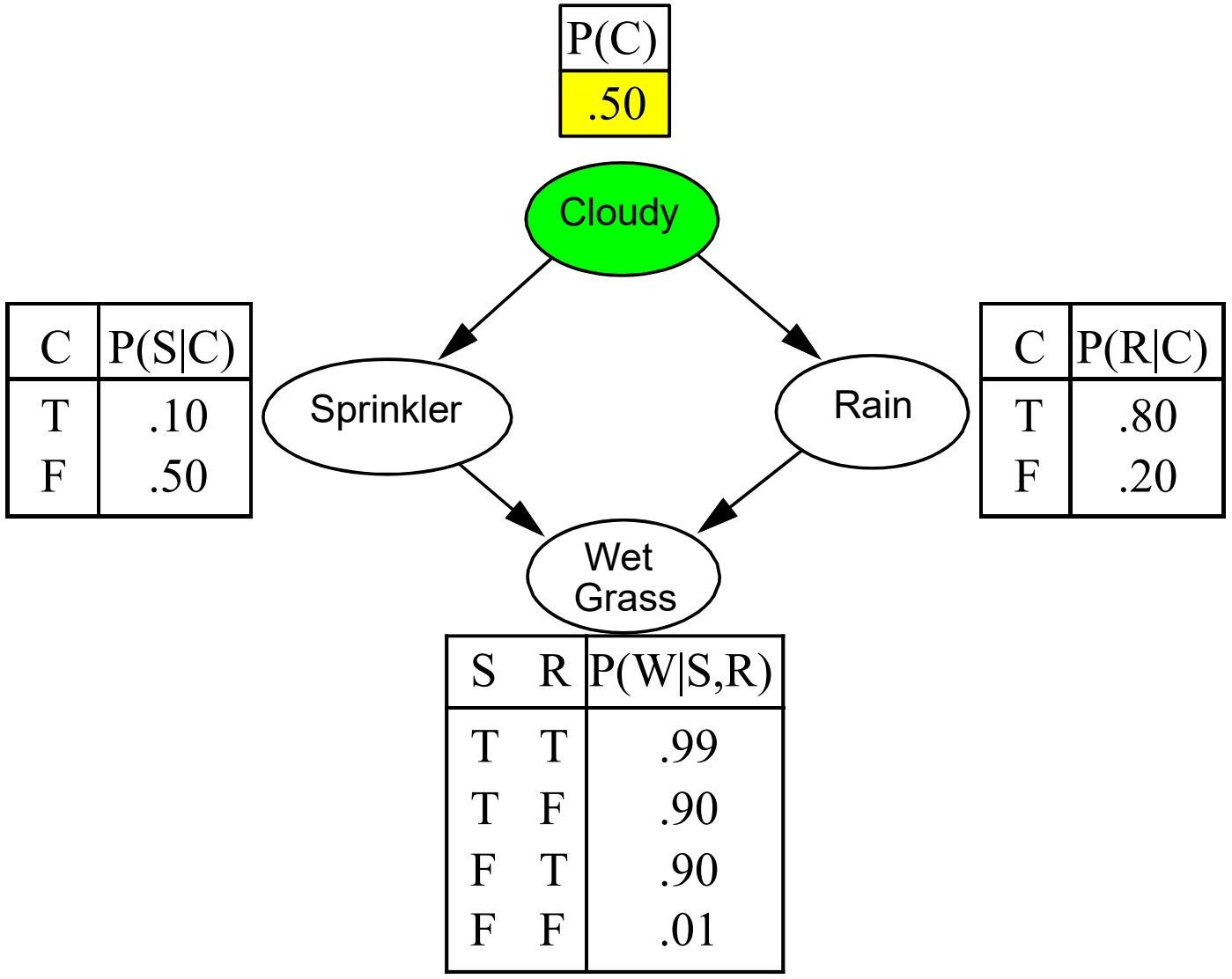
Sampling from an empty network

```
function Prior-Sample(bn) returns an event sampled from bn
    inputs: bn, a belief network specifying joint distribution P(X_1, ..., X_n)

    x ← an event with n elements
    for i = 1 to n do
        x_i ← a random sample from P(X_i | parents(X_i))
              given the values of Parents(X_i) in x
    return x
```

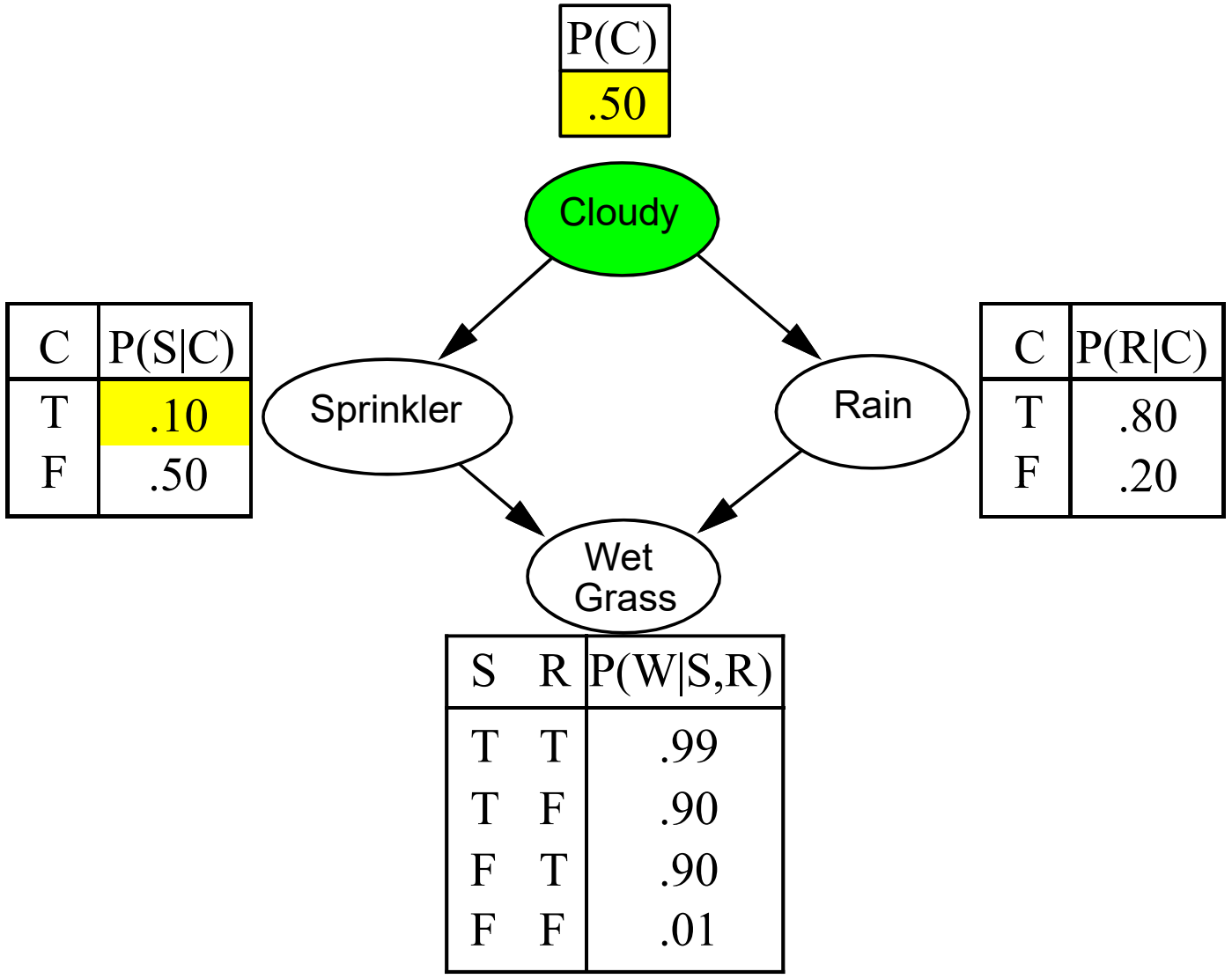
Example



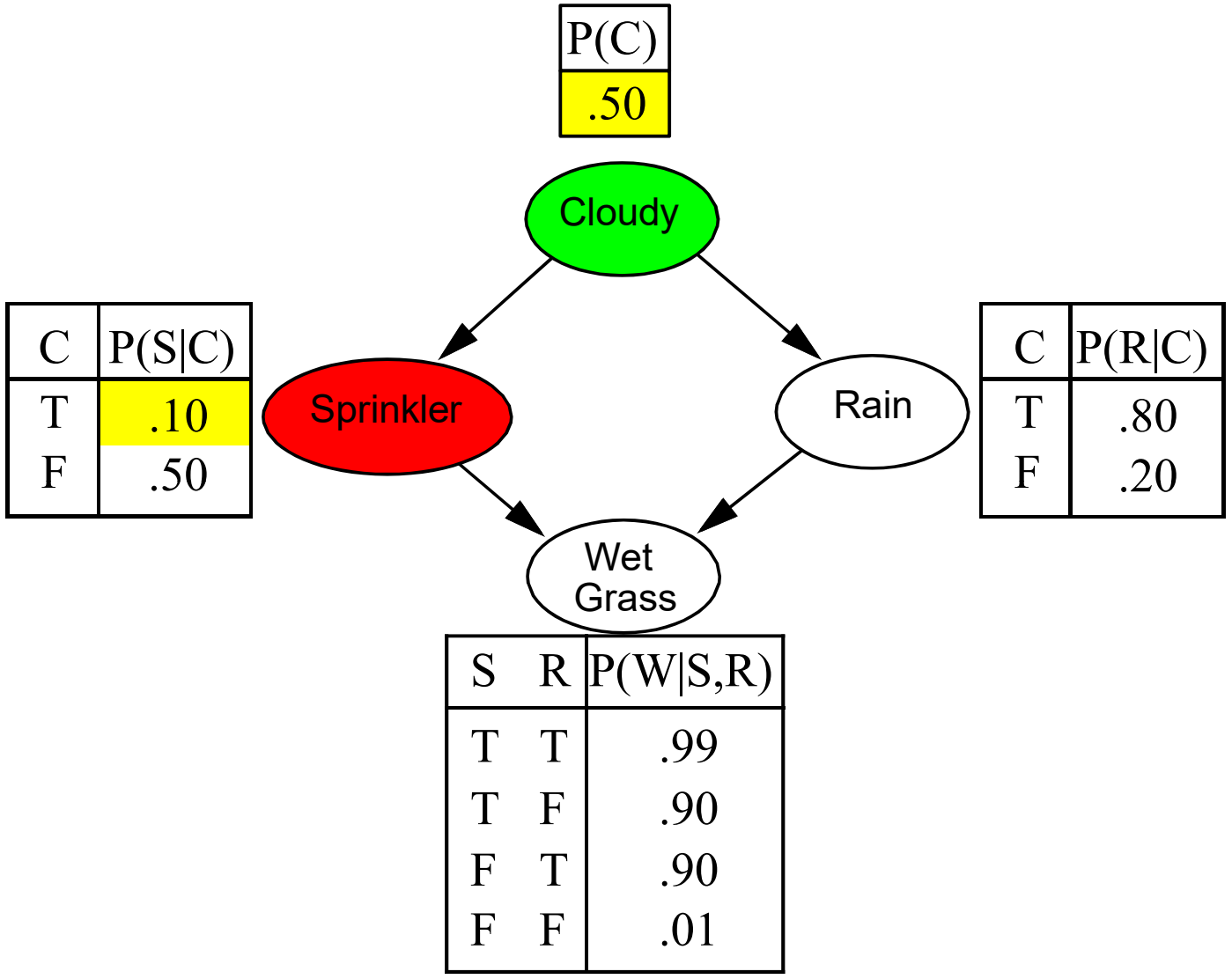
Example



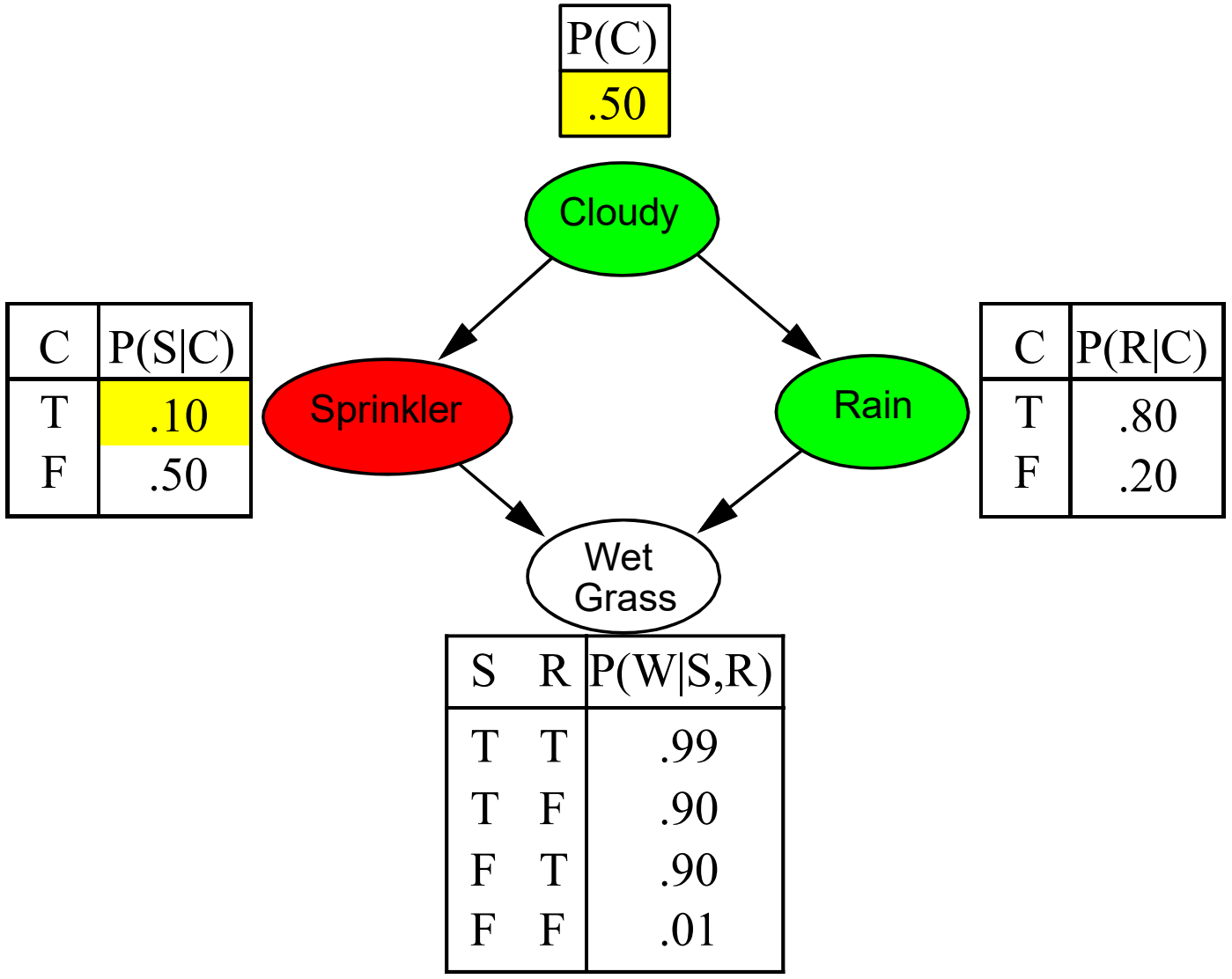
Example



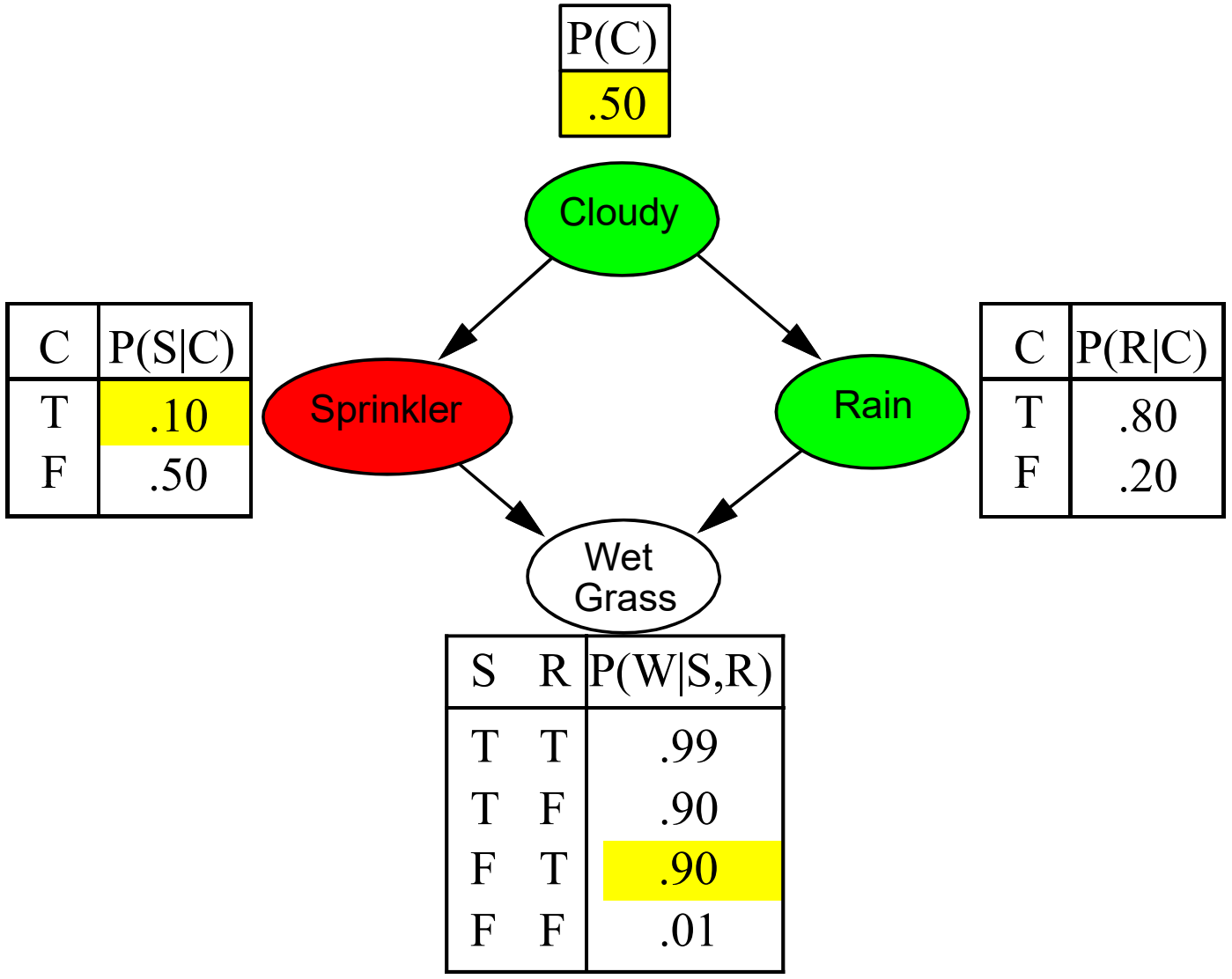
Example



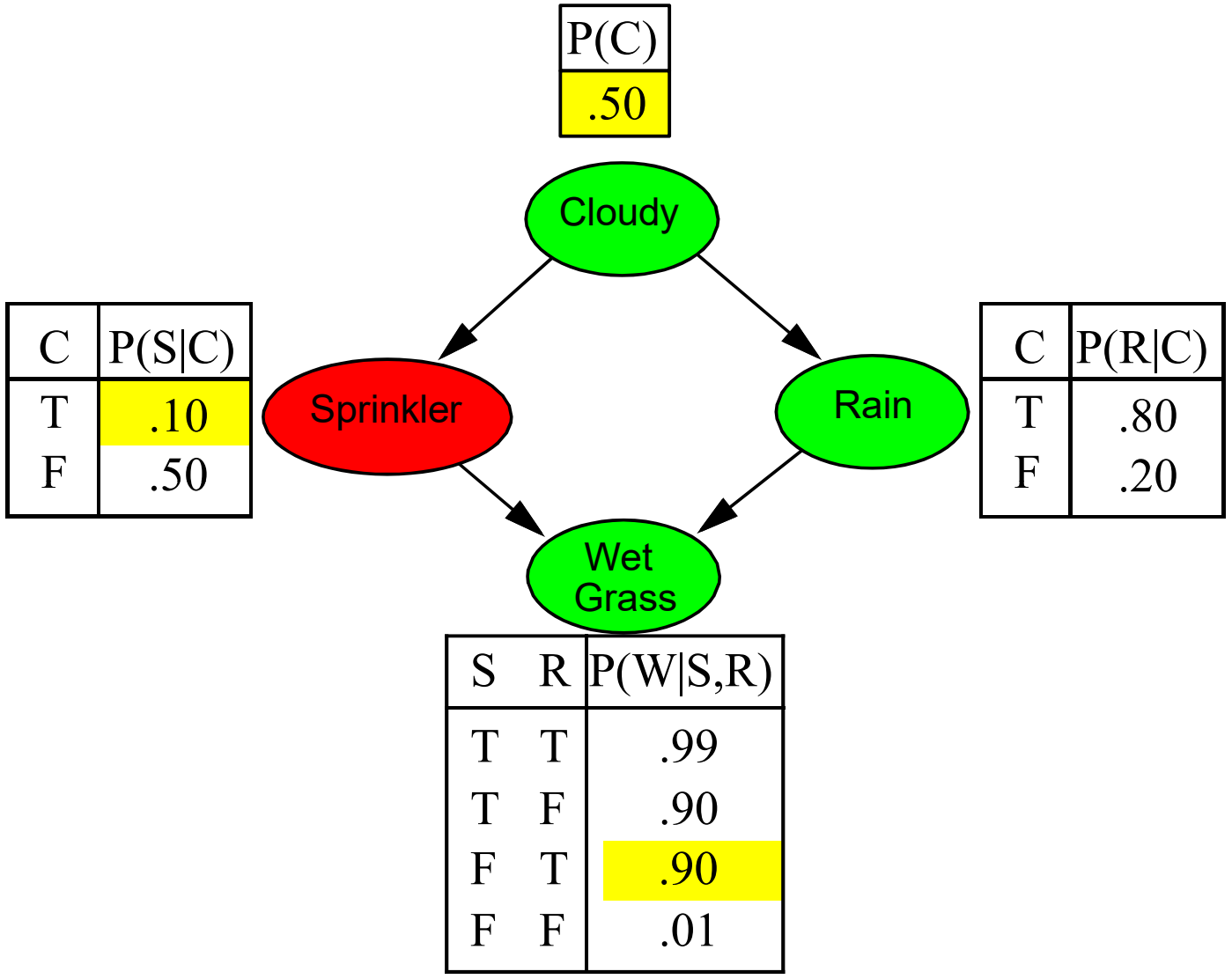
Example



Example



Example



Sampling from an empty network contd.

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 $\mathbf{e}$

```
function Rejection-Sampling(X,e,bn,N) returns an estimate of P(X|e)
  local variables: N, a vector of counts over X, initially zero

  for j = 1 to N do
      x ← Prior-Sample(bn)
      if x is consistent with e then
          N[x] ← N[x]+1 where x is the value of X in x
  return Normalize(N[X])
```

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 $Rain = false$.

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

Similar to a basic real-world empirical estimation procedure

Analysis of rejection sampling

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

Hence rejection sampling returns consistent posterior estimates

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

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

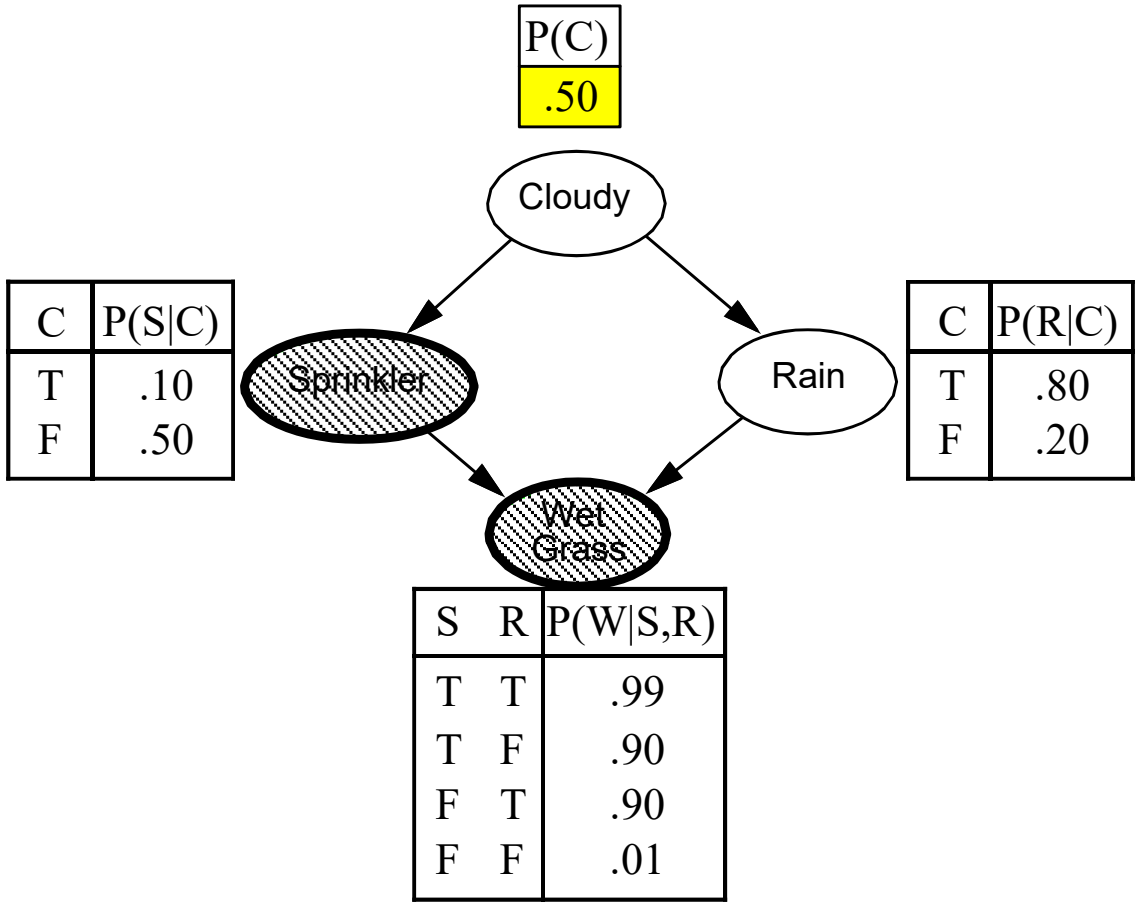
Likelihood weighting

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

```
function Likelihood-Weighting(X, e, bn, N) returns an estimate of P(X|e)
  local variables: W, a vector of weighted counts over X, initially zero

  for j = 1 to N do
      x, w ← Weighted-Sample(bn)
      W[x] ← W[x] + w where x is the value of X in x
  return Normalize(W[X])
```

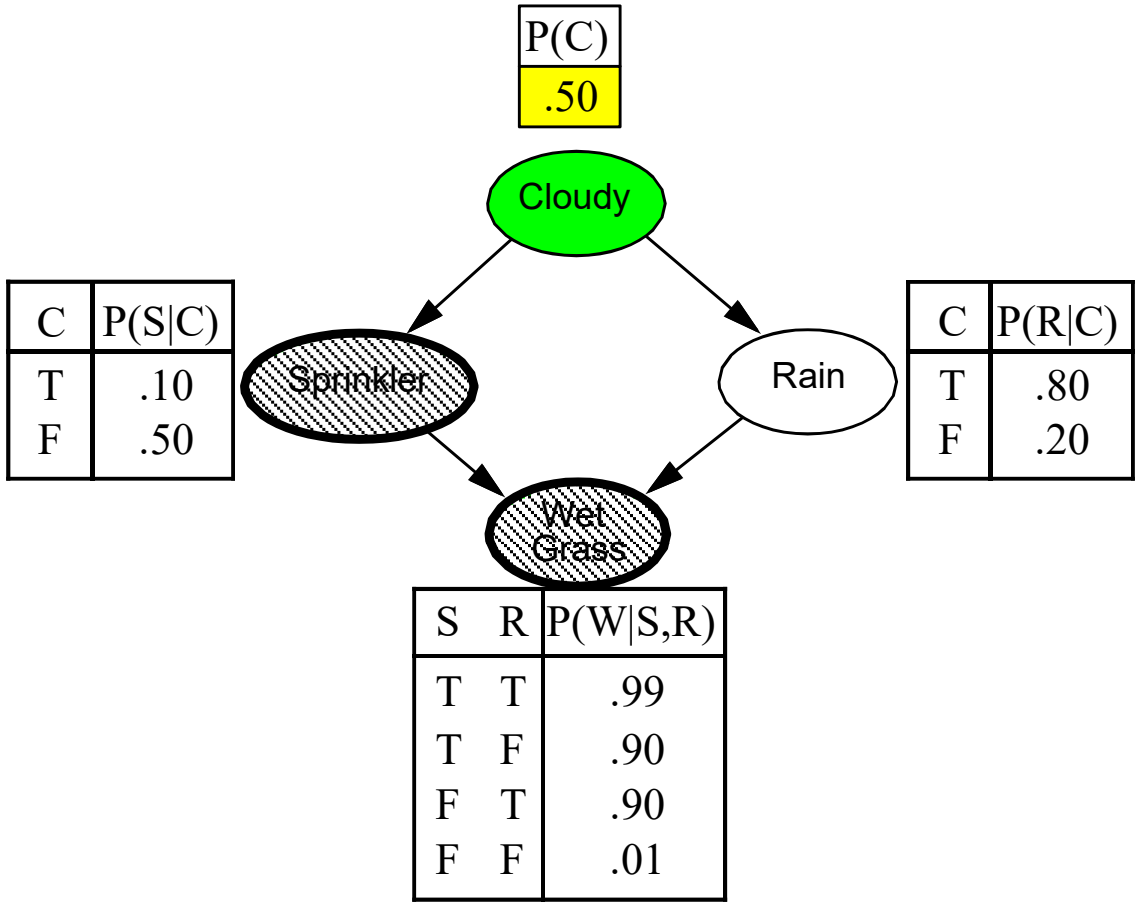
```
function Weighted-Sample(bn, e) returns an event and a weight

  x ← an event with n elements; w ← 1
  for i = 1 to n do
      if X_i has a value x_i in e
          then w ← w × P(X_i = x_i | parents(X_i))
          else x_i ← a random sample from P(X_i | parents(X_i))
  return x, w
```

Likelihood weighting example



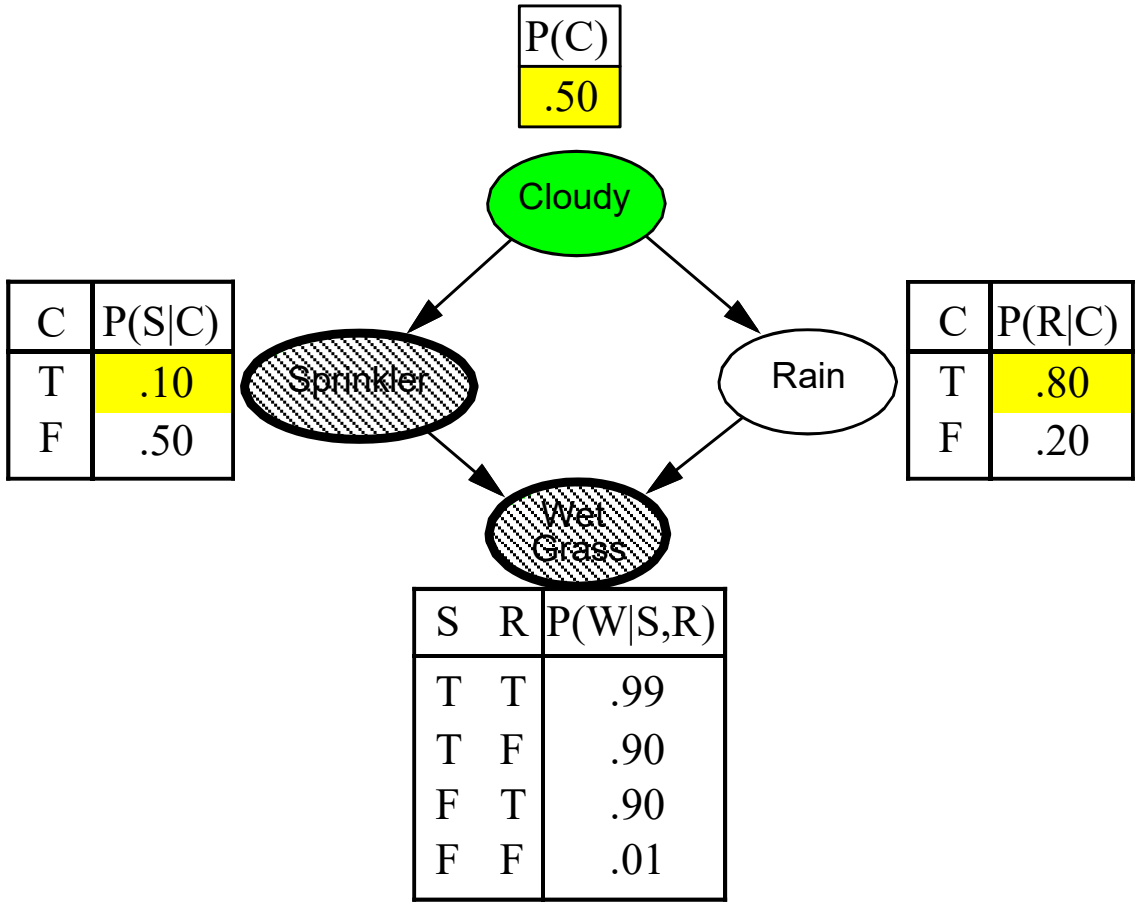
$w = 1.0$

Likelihood weighting example



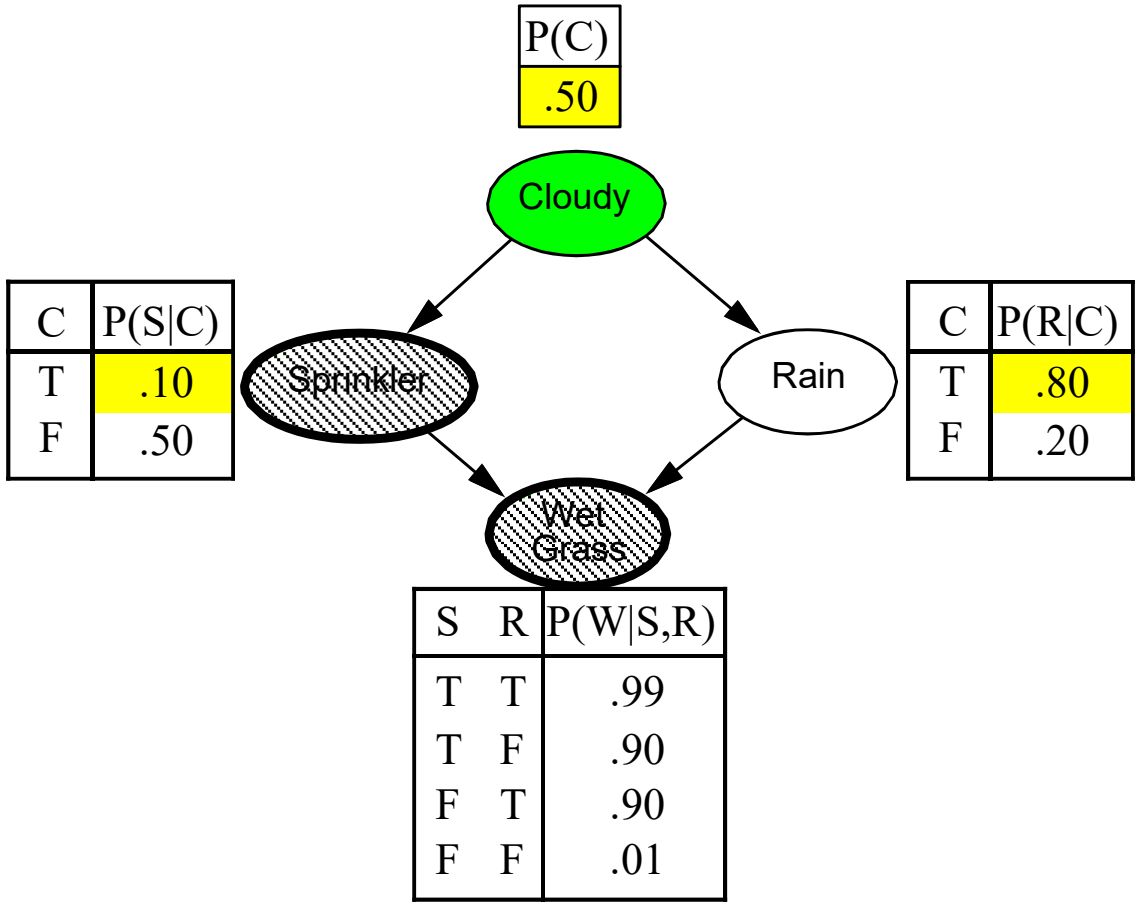
$w = 1.0$

Likelihood weighting example



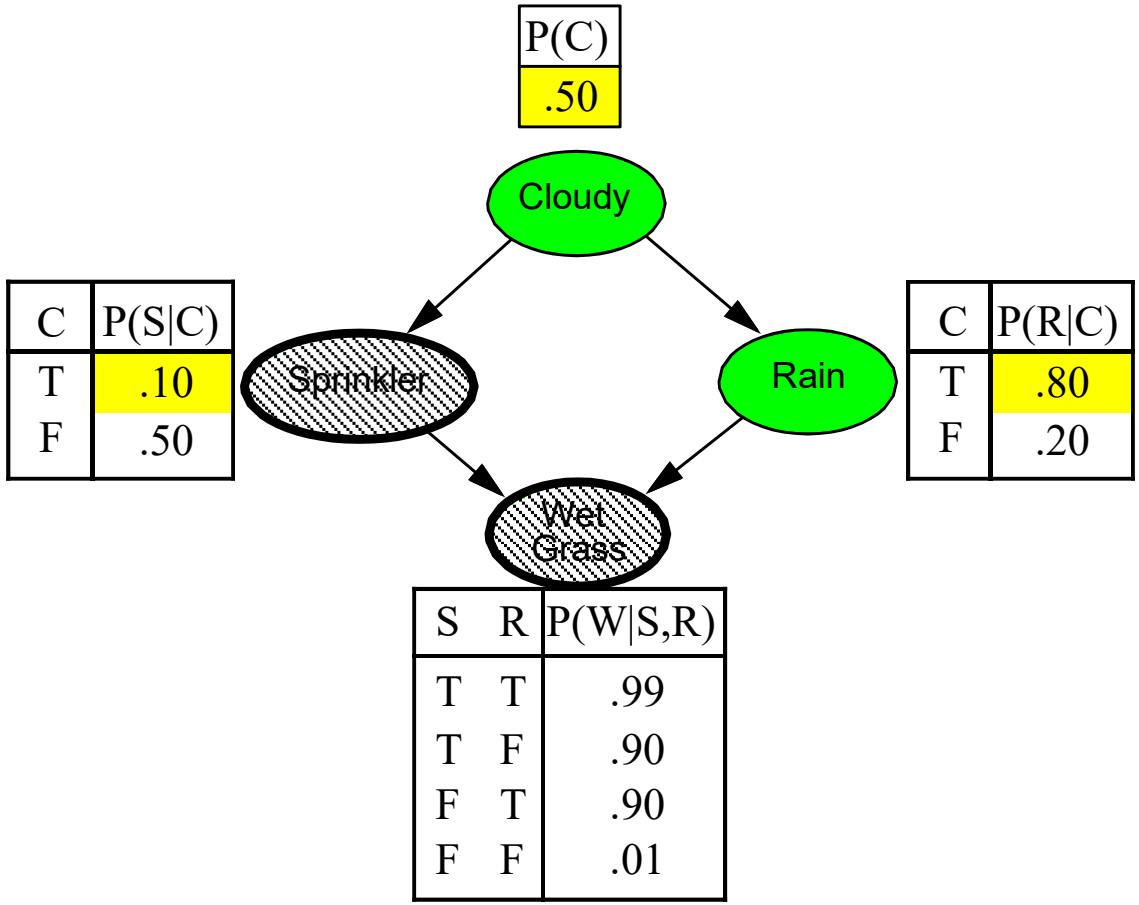
$w = 1.0$

Likelihood weighting example



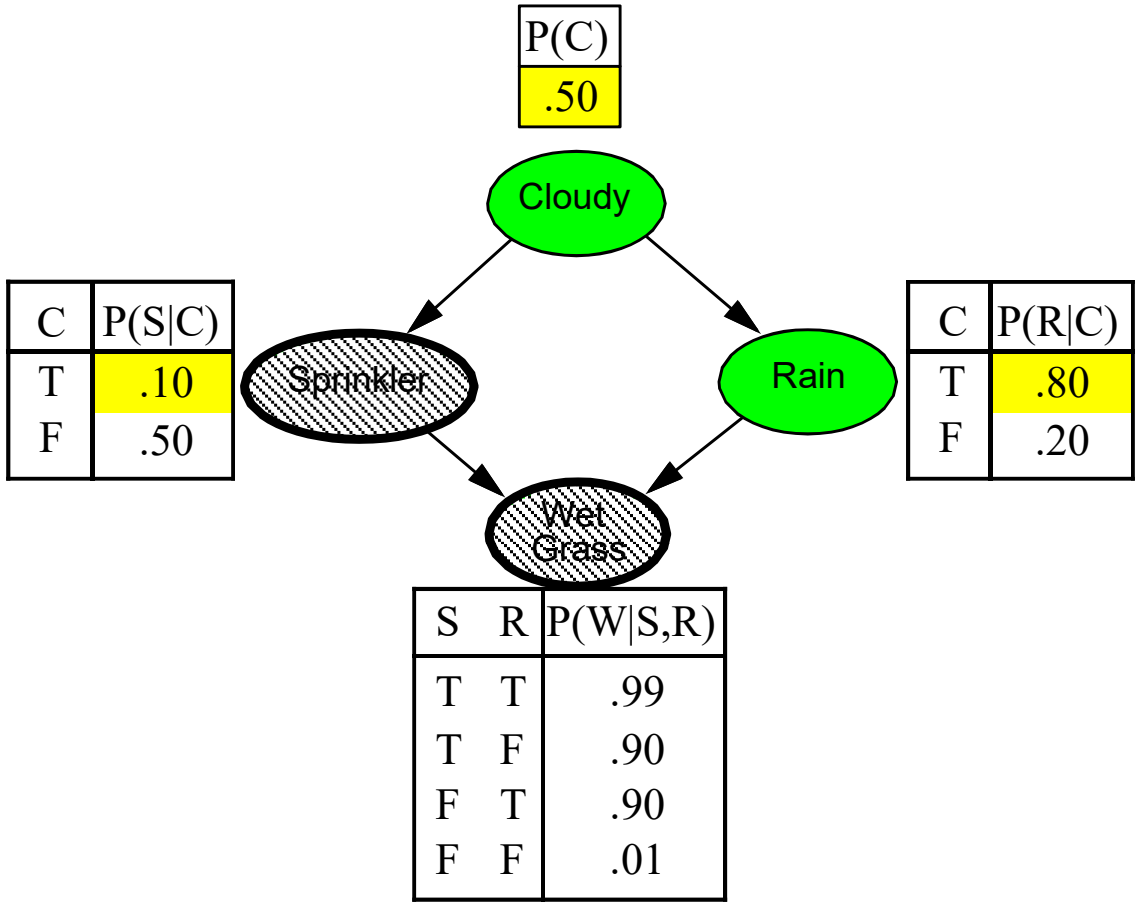
$w = 1.0 \times 0.1$

Likelihood weighting example



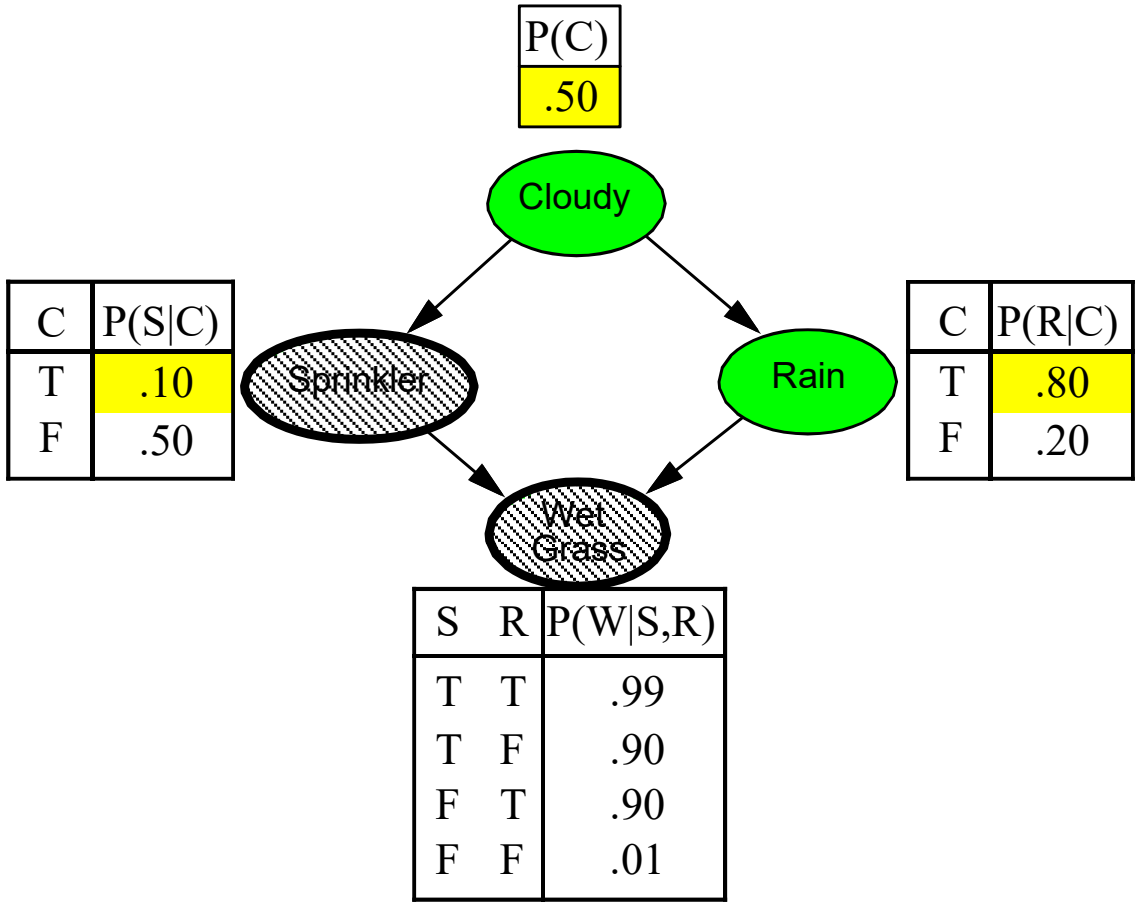
$w = 1.0 \times 0.1$

Likelihood weighting example



$w = 1.0 \times 0.1$

Likelihood weighting example



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

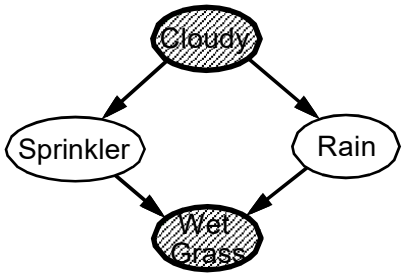
Likelihood weighting analysis

Sampling probability for WeightedSample is

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

Note: pays attention to evidence in ancestors only

⇒ somewhere "in between" prior and posterior distribution



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

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

Weighted sampling probability is

$$S_{WS}(\mathbf{z}, \mathbf{e})\, w(\mathbf{z}, \mathbf{e}) = \prod_{i=1}^{l} P(z_i | parents(Z_i)) \prod_{i=1}^{m} P(e_i | parents(E_i)) = P(\mathbf{z}, \mathbf{e})$$

(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.

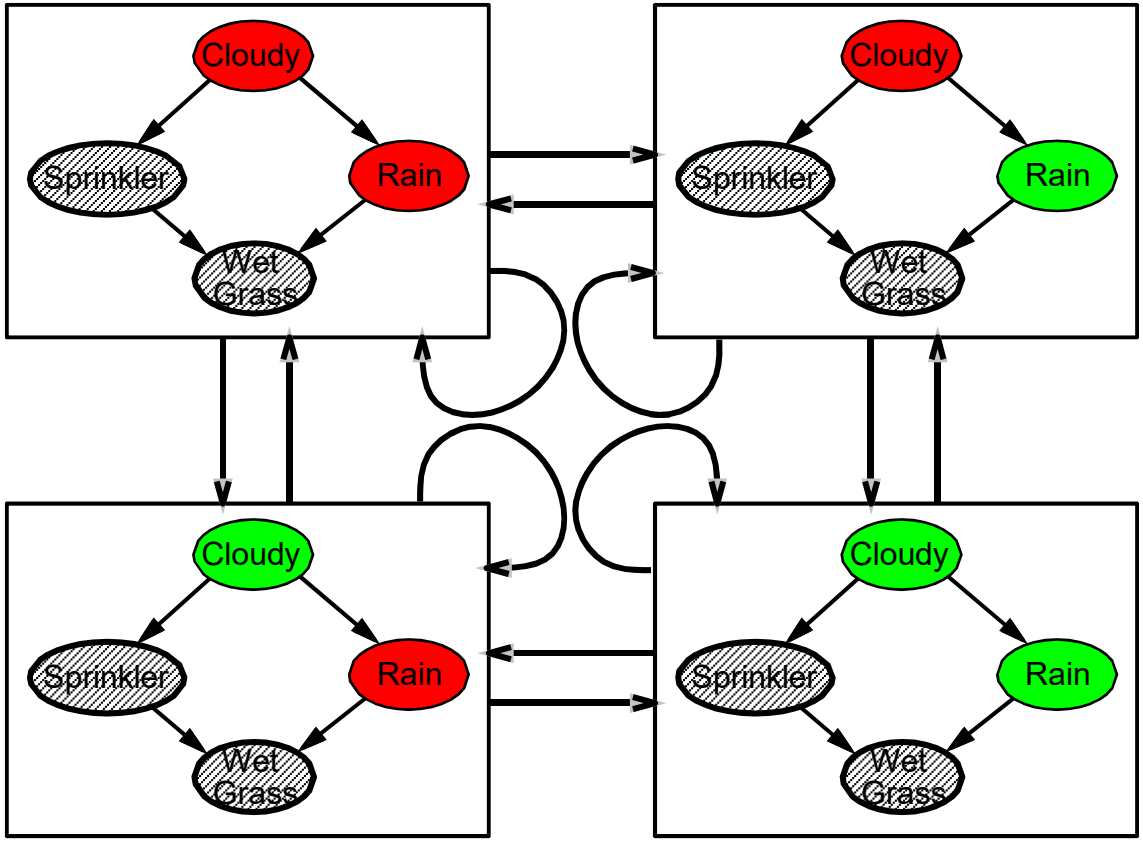
Generate next state by sampling one variable given Markov blanket
Sample each variable in turn, keeping evidence fixed

```
function MCMC-Ask(X, e, bn, N) returns an estimate of P(X|e)
  local variables: N[X], a vector of counts over 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 Y
  for j = 1 to N do
      for each Z_i in Z do
          sample the value of Z_i in x from P(Z_i | mb(Z_i))
              given the values of MB(Z_i) in x
          N[x] ← N[x] + 1 where x is the value of X in x
  return Normalize(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 contd.

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

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

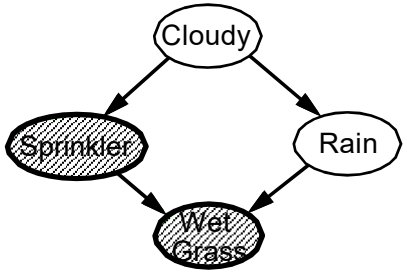
E.g., visit 100 states
31 have $Rain = true$, 69 have $Rain = false$

$\hat{\mathrm{P}}(Rain|Sprinkler = true, WetGrass = true)$
$= \mathrm{Normalize}((31, 69)) = (0.31, 0.69)$

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)

Causal Networks

Causal Networks: a restricted class of Bayesian networks that forbids all but causally compatible orderings.

$$P(c, r, s, w, g) = P(c)\, P(r \mid c)\, P(s \mid c)\, P(w \mid r, s)\, P(g \mid w)$$

$$C = f_C(U_C)$$
$$R = f_R(C, U_R)$$
$$S = f_S(C, U_S)$$
$$W = f_W(R, S, U_W)$$
$$G = f_G(W, U_G)$$

For example, suppose we turn the sprinkler on—$do(Sprinkler = true)$

$$P(c, r, w, g \mid do(S = true)) = P(c)\, P(r \mid c)\, P(w \mid r, s = true)\, P(g \mid w)$$

Causal Networks

Example:

Predict the effect of turning on the sprinkler on a downstream variable such as *GreenerGrass*, but the adjustment formula must take into account not only the direct route from Sprinkler, but also the "back door" route via Cloudy and Rain.

$$P(g|do(S = true) = \sum_r P(g|S = true, r)P(r)$$

we wish to find the effect of $do(X_j = x_{jk})$ on a variable X_i,

Back-door criterion

allows us to write an adjustment formula that conditions on any set of variables **Z** that closes the back door, so to speak

Summary

Bayes nets provide a natural representation for (causally induced) conditional independence

Topology + CPTs = compact representation of joint distribution

Generally easy for (non)experts to construct

Canonical distributions (e.g., noisy-OR) = compact representation of CPTs

Continuous variables ⇒ parameterized distributions (e.g., linear Gaussian)

Exact inference by variable elimination:
– polytime on polytrees, NP-hard on general graphs
– space = time, very sensitive to topology

Random sampling techniques such as likelihood weighting and Markov chain Monte
Carlo can give reasonable estimates of the true posterior probabilities in a network and can cope with much larger networks than can exact algorithms.