CSCI 561
Foundation for Artificial Intelligence

20. Belief (Bayesian) Networks

Professor Wei-Min Shen

University of Southern California

Belief (Bayesian) Networks

- Motivation
- Independence and Conditional Independence
- Syntax and Semantics
- Reasoning with Belief Networks
- Construction of Belief Networks
- Inference Algorithms
  - Exact inference
  - Approximate inference

Why Bayesian Networks?

A better representation for the Probability Distribution Model

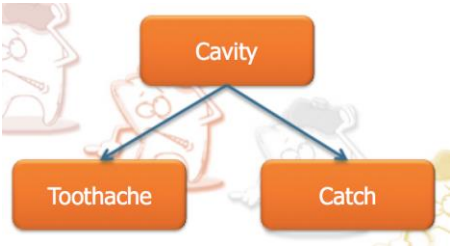
(take advantage of "variable independence")

Fully Joint Distribution Table

Cavity	Catch	Toothache	Logic Truth	Probability
0	0	0	{0,1}	0.576
0	0	1	{0,1}	0.064
0	1	0	{0,1}	0.144
0	1	1	{0,1}	0.016
1	0	0	{0,1}	0.008
1	0	1	{0,1}	0.012
1	1	0	{0,1}	0.072
1	1	1	{0,1}	0.108

$\Leftrightarrow$

Bayesian Network



$\mathbf{P}(X_1, X_2, \ldots, X_n)$

Size = $O(d^n)$

d: variable domain

$\Leftrightarrow$

$$\prod_{i=1}^{n} \mathbf{P}(X_i | Parents(X_i))$$

Size = $O(nd^k)$

k: # of parents

Independence

Two random variables A B are (absolutely) independent iff

$$P(A|B) = P(A)$$

or $P(A, B) = P(A|B)P(B) = P(A)P(B)$

e.g., A and B are two coin tosses

If n Boolean variables are independent, the full joint is

$$\mathbf{P}(X_1, \ldots, X_n) = \prod_i \mathbf{P}(X_i)$$

hence can be specified by just n numbers

Absolute independence is a very strong requirement, seldom met

Conditional Independence

Consider the dentist problem with three random variables:
$Toothache$, $Cavity$, $Catch$ (steel probe catches in my tooth)

The full joint distribution has $2^3 - 1 = 7$ independent entries

If I have a cavity, the probability that the probe catches in it doesn't depend on whether I have a toothache:

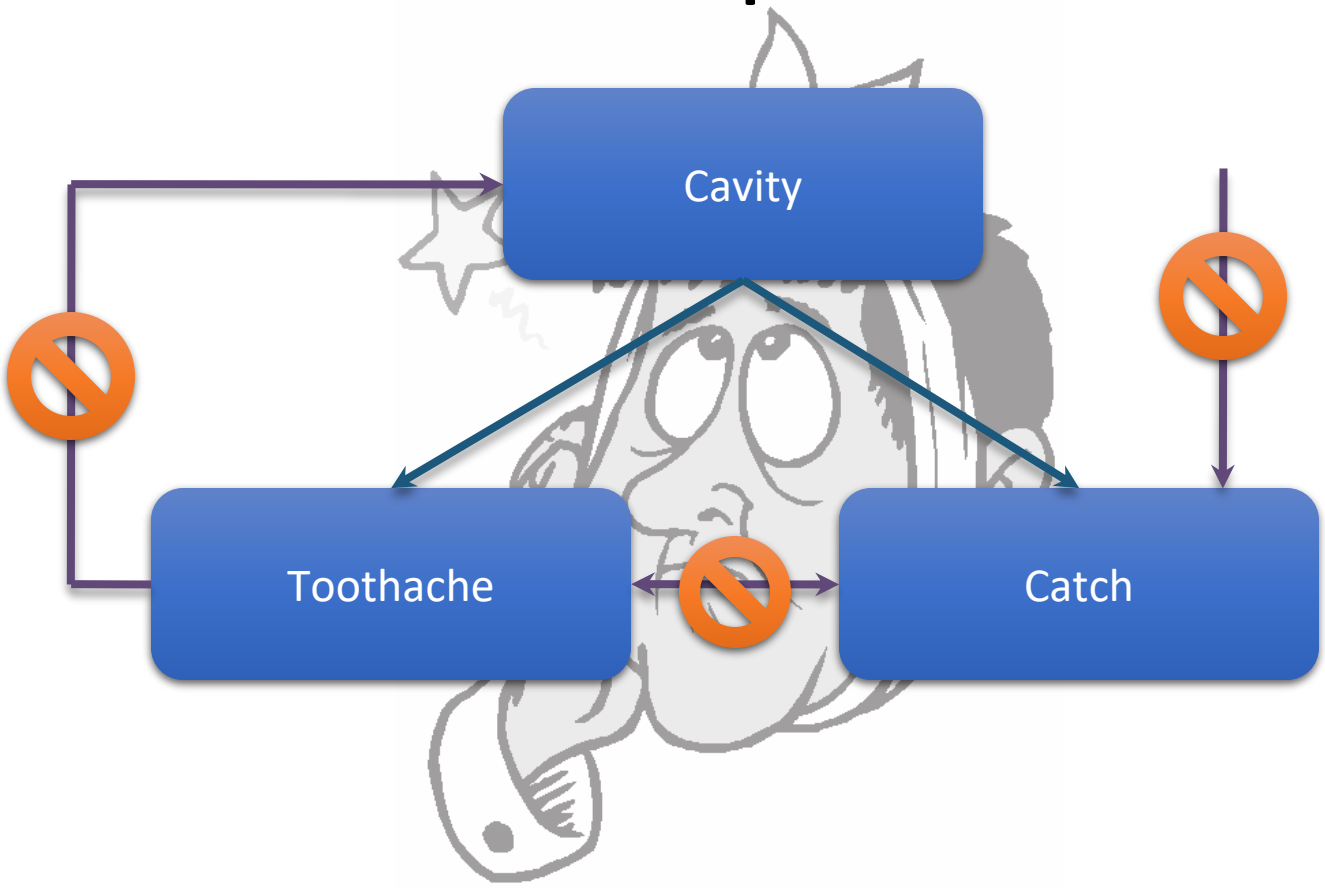
(1) $P(Catch|Toothache, Cavity) = P(Catch|Cavity)$

i.e., $Catch$ is conditionally independent of $Toothache$ given $Cavity$

The same independence holds if I haven't got a cavity:

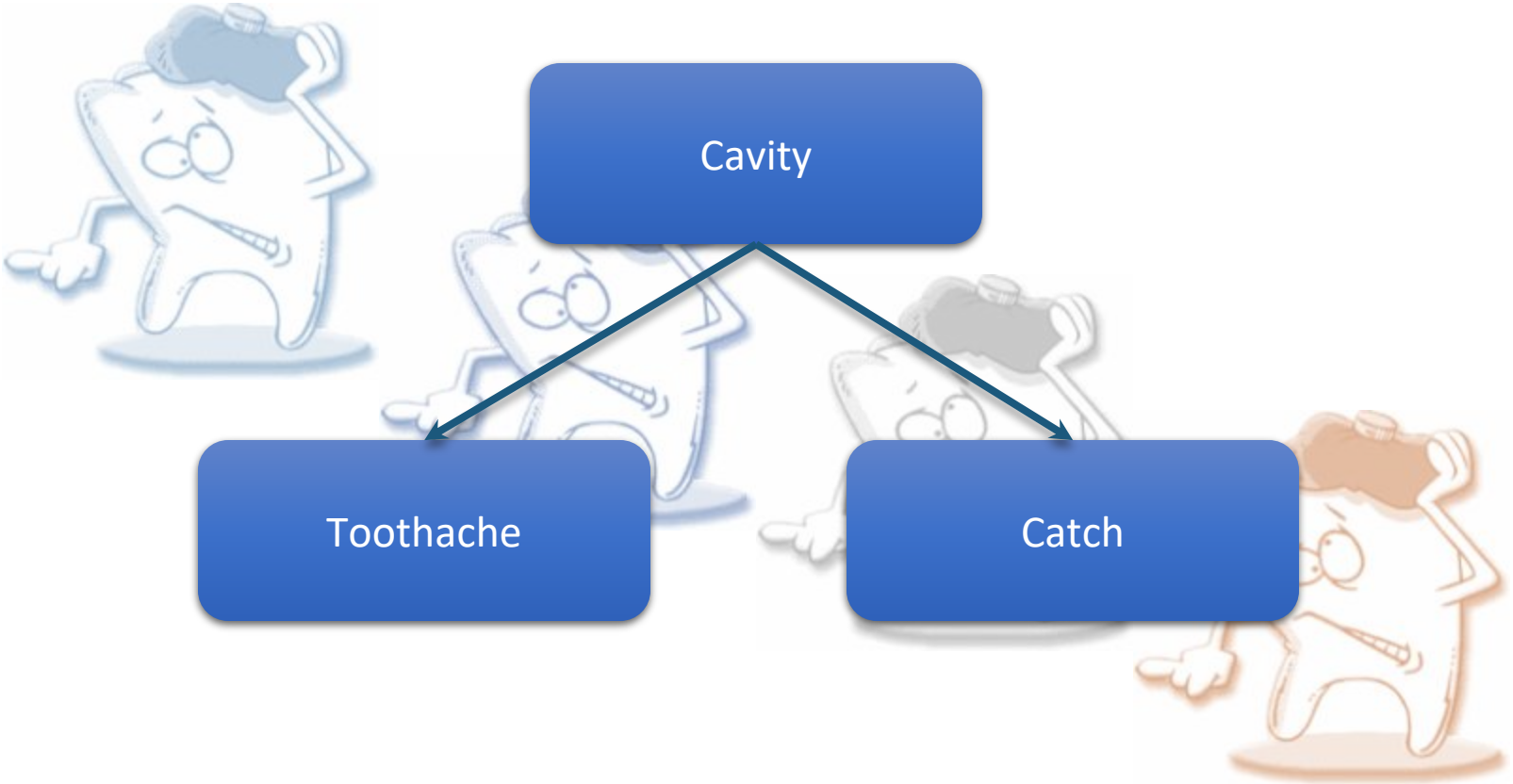
(2) $P(Catch|Toothache, \neg Cavity) = P(Catch|\neg Cavity)$

Conditional Independence



Other interactions may exist, but they are either insignificant, unknown or irrelevant. We leave them out.

Conditional Independence



We **assume** that a “catch” is not influenced by a “toothache” and visa versa.

Conditional Independence

Equivalent statements to (1)

(1a) $P(Toothache|Catch, Cavity) = P(Toothache|Cavity)$ Why??

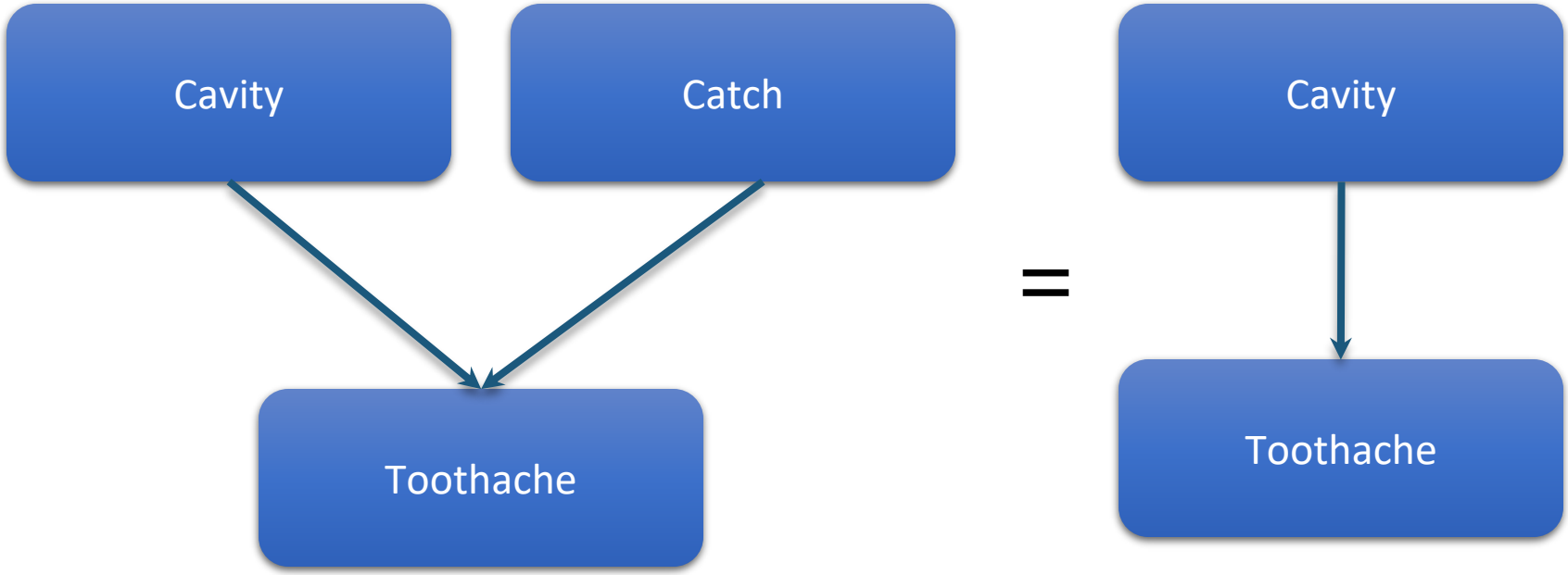
(1b) $P(Toothache, Catch|Cavity) = P(Toothache|Cavity)P(Catch|Cavity)$
Why??

Full joint distribution can now be written as

$$\mathbf{P}(Toothache, Catch, Cavity) = \mathbf{P}(Toothache, Catch|Cavity)\mathbf{P}(Cavity)$$
$$= \mathbf{P}(Toothache|Cavity)\mathbf{P}(Catch|Cavity)\mathbf{P}(Cavity)$$

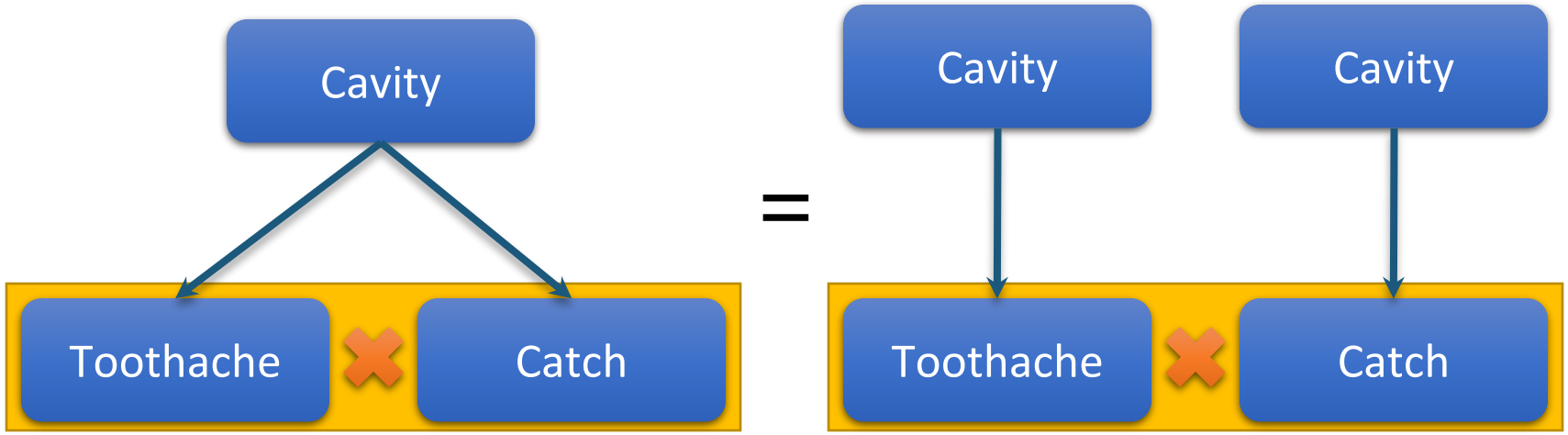
i.e., $2 + 2 + 1 = 5$ independent numbers (equations 1 and 2 remove 2)

Conditional Independence



(1a) Since Catch does not affect Toothache
P(Toothache | Catch, Cavity) = P(Toothache | Cavity)

Conditional Independence



(1b) Algebraically, these statements are equivalent

P(Toothache, Catch | Cavity) = P(Toothache | Cavity) P(Catch |Cavity)

Belief Networks

A simple, graphical notation for conditional independence assertions and hence for compact specification of full joint distributions

Syntax:

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

In the simplest case, conditional distribution represented as a conditional probability table (CPT)

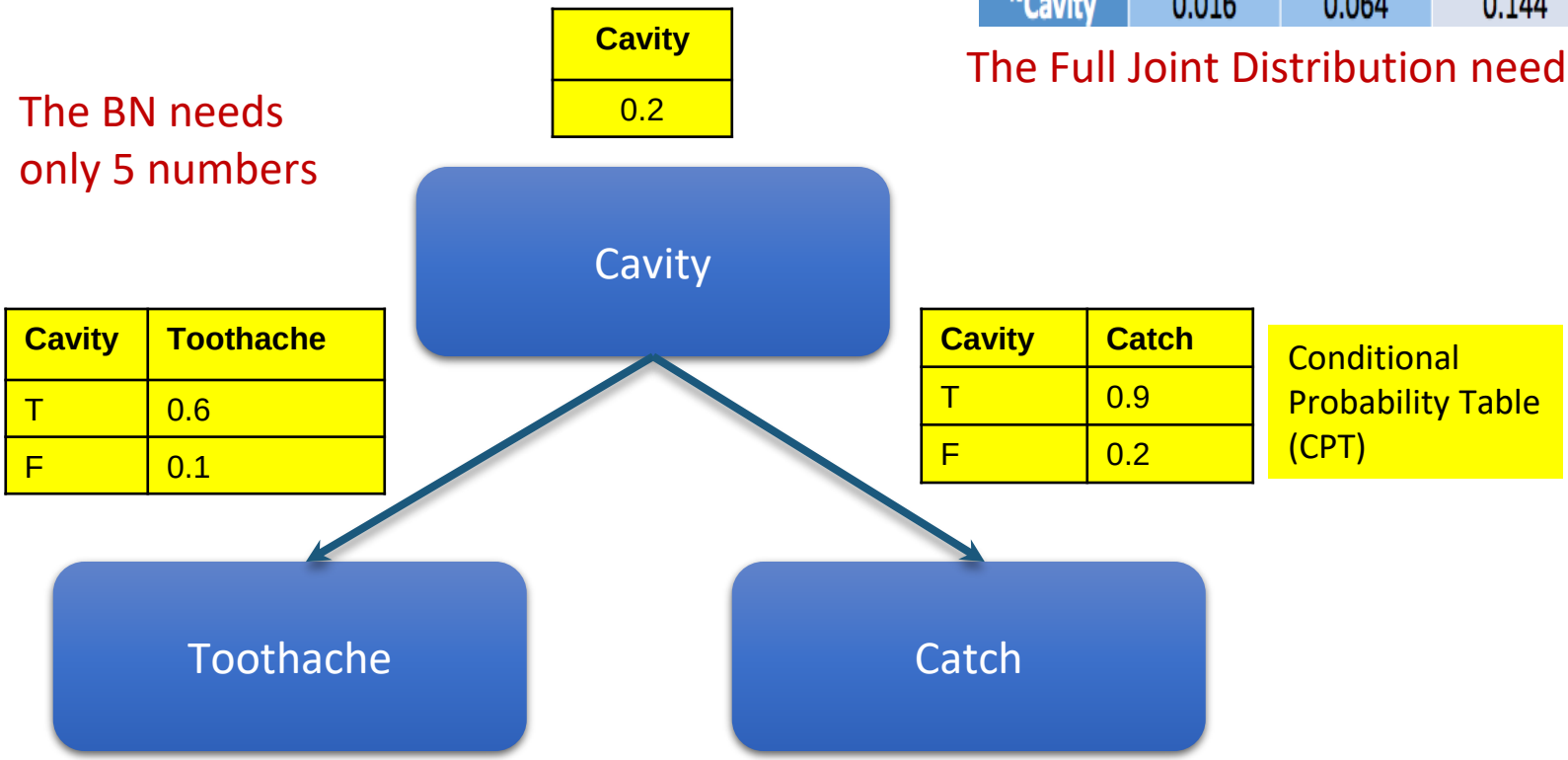
Bayesian Network vs. Full Joint Distribution

BN has less entries than FJD,
but contains the same information!

	Toothache		~Toothache	
	Catch	~Catch	Catch	~Catch
Cavity	0.108	0.012	0.072	0.008
~Cavity	0.016	0.064	0.144	0.576

The Full Joint Distribution needs 8 numbers

The BN needs
only 5 numbers

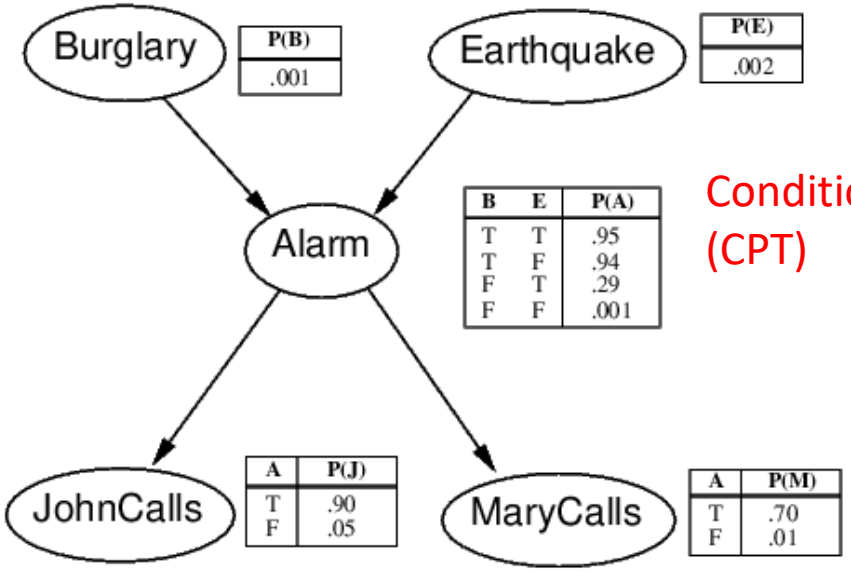


Example

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?

The Full Joint Distribution needs 2^5 = 32 values

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



The BN needs only 10 values

Conditional Probability Table (CPT)

k=2
d=2
n=5

Note: $\leq k$ parents $\Rightarrow$ $O(d^k n)$ numbers vs. $O(d^n)$

Semantics

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

$$\mathbf{P}(X_1, \ldots, X_n) = \prod_{i=1}^{n} \mathbf{P}(X_i | Parents(X_i))$$

e.g., $P(J \wedge M \wedge A \wedge \neg B \wedge \neg E)$ is given by??

$$= P(\neg B)P(\neg E)P(A|\neg B \wedge \neg E)P(J|A)P(M|A)$$

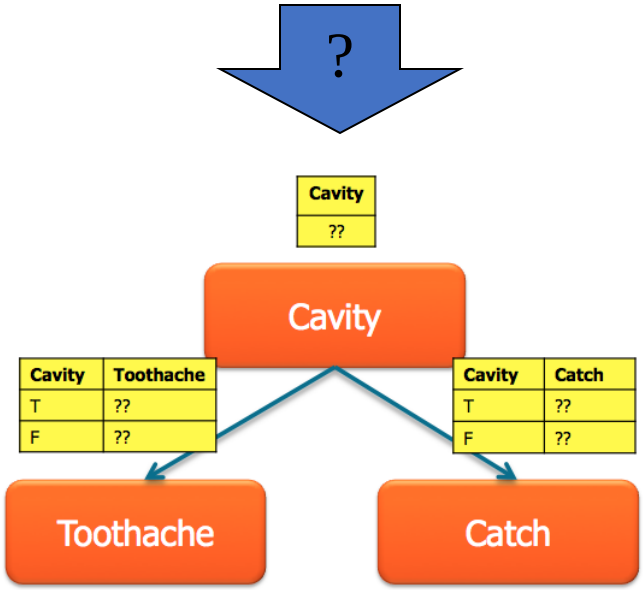
"Local" semantics: each node is conditionally independent of its nondescendants given its parents

Theorem: Local semantics $\Leftrightarrow$ global semantics

Bayesian Network ⇔ Full Joint Distribution

Cavity	Catch	Toothache	Logic Truth	Probability
0	0	0	{0,1}	0.576
0	0	1	{0,1}	0.064
0	1	0	{0,1}	0.144
0	1	1	{0,1}	0.016
1	0	0	{0,1}	0.008
1	0	1	{0,1}	0.012
1	1	0	{0,1}	0.072
1	1	1	{0,1}	0.108

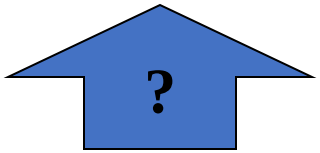
Try it yourself!

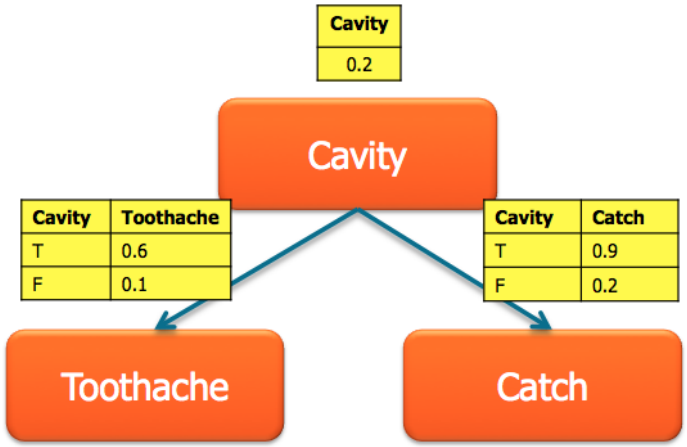


Bayesian Network ⇔ Full Joint Distribution

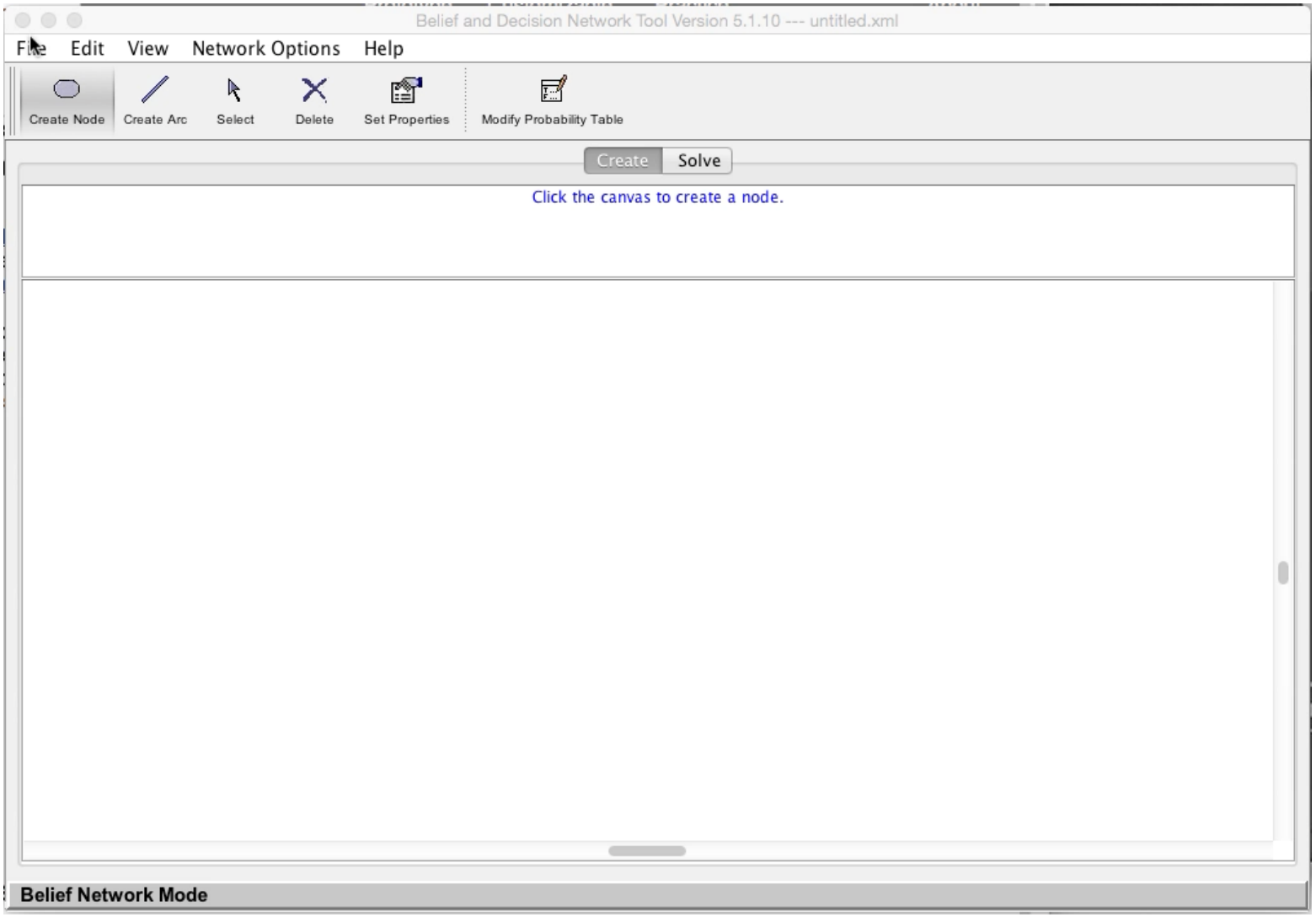
Cavity	Catch	Toothache	Logic Truth	Probability
0	0	0	{0,1}	??
0	0	1	{0,1}	??
0	1	0	{0,1}	??
0	1	1	{0,1}	??
1	0	0	{0,1}	??
1	0	1	{0,1}	??
1	1	0	{0,1}	??
1	1	1	{0,1}	??

Try it yourself!



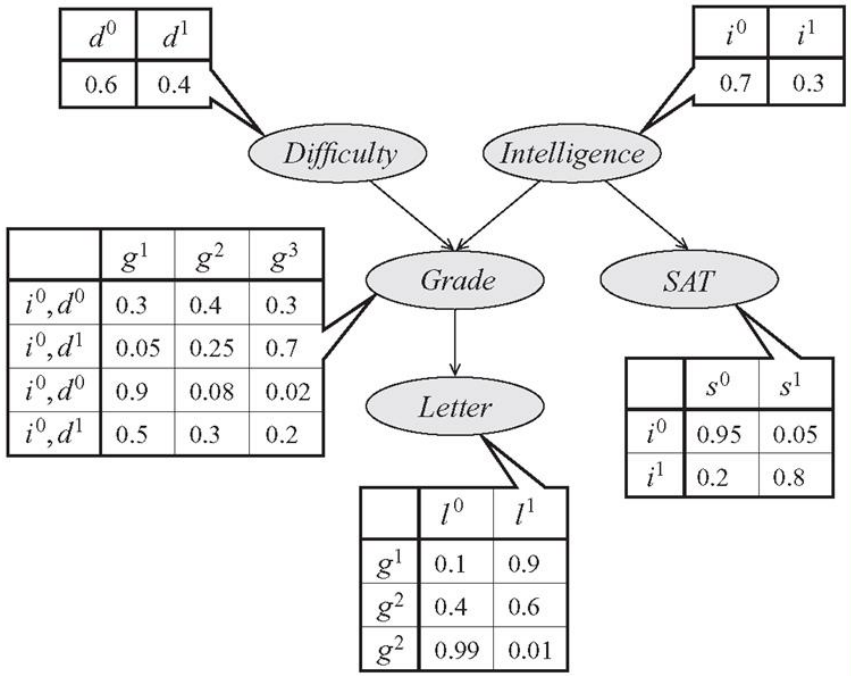


Demo


Belief and Decision Network Tool Version 5.1.10 --- untitled.xml
File
Edit
View
Network Options
Help
Create Node
Create Arc
Select
Delete
Set Properties
Modify Probability Table
Create
Solve
Click the canvas to create a node.
Belief Network Mode

Bayesian Network: Student Model

Graph and CPDs



d^0	d^1
0.6	0.4

i^0	i^1
0.7	0.3

	g^1	g^2	g^3
i^0, d^0	0.3	0.4	0.3
i^0, d^1	0.05	0.25	0.7
i^0, d^0	0.9	0.08	0.02
i^0, d^1	0.5	0.3	0.2

typo: these two rows are for i^1

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6
g^2	0.99	0.01

$Val(I)=\{i^0=$low intelligence, $i^1=$ high intelligence$\}$

$Val(D)=\{d^0=$easy, $d^1=$hard$\}$

$Val(G)=\{g^1=A, g^2=B, g^3=C\}$

$Val(S)=\{s^0=$low, $s^1=$high$)$

$Val(L)=\{l^0=$weak, $l^1=$strong$\}$

$$P(D,I,G,S,L) = P(D)P(I)P(G \mid D,I)P(S \mid I)P(L \mid G)$$

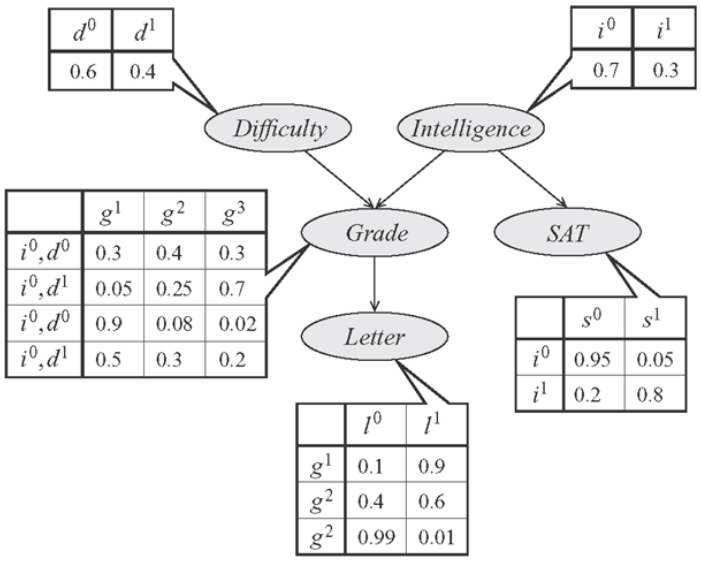
$$P(i^1,d^0,g^2,s^1,l^0) = P(i^1)P(d^0)P(g^2 \mid i^1,d^0)P(s^1 \mid i^1)P(l^0 \mid g^2)$$

$$=0.3 \cdot 0.6 \cdot 0.08 \cdot 0.8 \cdot 0.4 = 0.004608$$

Chain rule for Bayesian network

Reasoning Patterns

Reasoning about a student George using the model



- Causal Reasoning
  - George is interested in knowing as to how likely he is to get a strong letter (based on intelligence, difficulty)?
- Evidential Reasoning
  - Recruiter is interested in knowing whether George is intelligent (based on letter, SAT)

4

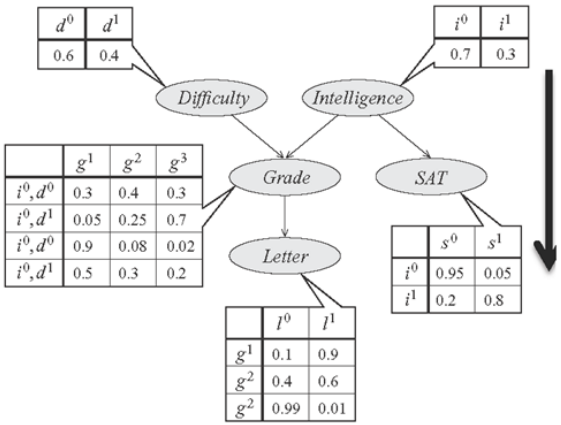
Causal Reasoning

1. How likely is George to get a strong letter (knowing nothing else)?
   - $P(l^1)=0.502$
   - Obtained by summing-out other variables in joint distribution
2. But George is not so intelligent (i^0)
   - $P(l^1|i^0)=0.389$ $=P(l^1 i^0)/P(i^0)$
3. Next we find out ECON101 is easy (d^0)
   - $P(l^1|i^0, d^0)=0.513$ $=P(l^1 i^0 d^0)/P(i^0 d^0)$

Observe how probabilities change as evidence is obtained

Try it yourself:

$$P(D,I,G,S,l^1)=\sum_{D,I,G,S} P(D)P(I)P(G|D,I)P(S|I)P(l^1|G)$$

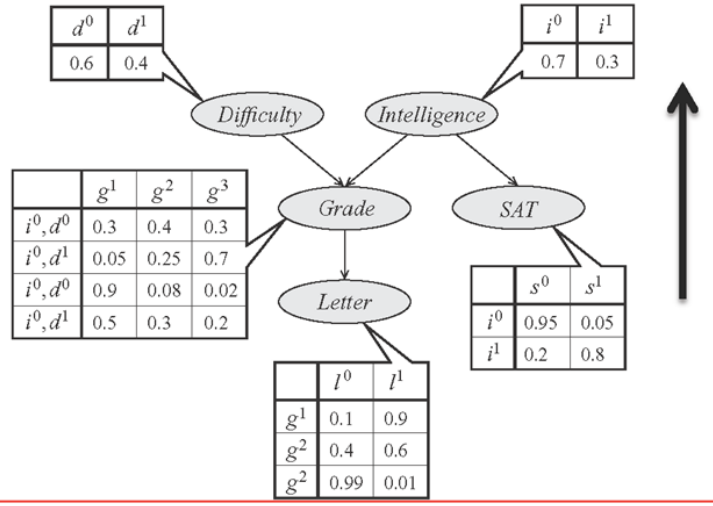


Query is Example of Causal Reasoning:

Predicting downstream effects of factors such as intelligence

Evidential Reasoning

- Recruiter wants to hire intelligent student
- A priori George is 30% likely to be intelligent
- $P(i^1)=0.3$
- Finds that George received grade C (g^3) in ECON101
- $P(i^1|g^3)=0.079$
- Similarly probability class is difficult goes up from 0.4 to
- $P(d^1|g^3)=0.629$
- If recruiter has lost grade but has letter
- $P(i^1|l^0)=0.14$



d^0	d^1
0.6	0.4

i^0	i^1
0.7	0.3

	g^1	g^2	g^3
i^0, d^0	0.3	0.4	0.3
i^0, d^1	0.05	0.25	0.7
i^0, d^0	0.9	0.08	0.02
i^0, d^1	0.5	0.3	0.2

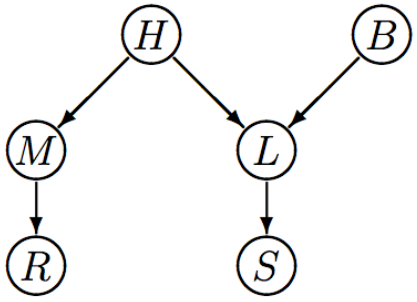
	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6
g^2	0.99	0.01

- Recruiter has both grade and letter
- $P(i^1|l^0,g^3)=0.079$
  - Same as if he had only grade
  - Letter is immaterial
- Reasoning from effects to causes is called evidential reasoning

6

Evidential Reasoning Example



H = hardware problem
B = bug in lab
M = mailer is running
L = lab is running
R = received email
S = problem solved

Brute force calculation of $P(H \mid E)$ is done by:

1. Apply the conditional probability rule.

$$P(H \mid E) = P(H \wedge E) \,/\, P(E)$$

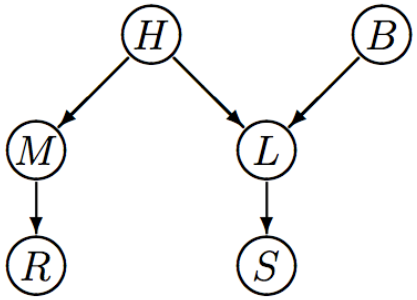
2. Apply the marginal distribution rule to the unknown vertices $\mathbf{U}$.

$$P(H \wedge E) = \sum_{\mathbf{U}=\mathbf{u}} P(H \wedge E \wedge \mathbf{U} = \mathbf{u})$$

3. Apply joint distribution rule for Bayesian networks.

$$\mathbf{P}(X_1, \ldots, X_n) = \prod_{i=1}^{n} \mathbf{P}(X_i \mid Parents(X_i))$$

Example



H = hardware problem
B = bug in lab
M = mailer is running
L = lab is running
R = received email
S = problem solved

Each node needs a probability table. Size of table depends on number of parents.

$\mathbf{P}(H)$	
$True$	$False$
0.01	0.99

	$\mathbf{P}(M \mid H)$	
H	$True$	$False$
$True$	0.1	0.9
$False$	0.99	0.01

e.g.,

		$\mathbf{P}(L \mid H, B)$	
H	B	$True$	$False$
$True$	$True$	0.01	0.99
$True$	$False$	0.1	0.9
$False$	$True$	0.02	0.98
$False$	$False$	1.0	0.0

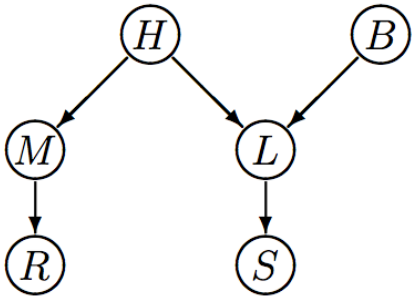
Evidential Reasoning Example

Calculate $P(B \mid \neg R, S)$ in the buggy lab example.

1. Apply the conditional probability rule.

$$P(B \mid \neg R, S) = \frac{P(B, \neg R, S)}{P(\neg R, S)}$$

2. Apply the marginal distribution rule to the unknown vertices. $P(B, \neg R, S)$ has 3 unknown vertices with $2^3 = 8$ possible value assignments.



H = hardware problem
B = bug in lab
M = mailer is running
L = lab is running
R = received email
S = problem solved

Each node needs a probability table. Size of table depends on number of parents.

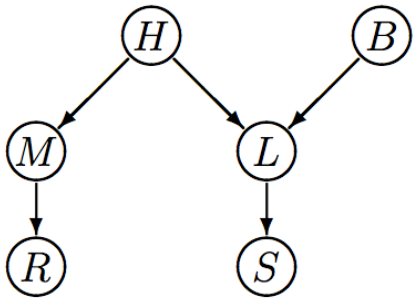
Evidential Reasoning Example

Calculate $P(B \mid \neg R, S)$ in the buggy lab example.

1. Apply the conditional probability rule.

$$P(B \mid \neg R, S) = \frac{P(B, \neg R, S)}{P(\neg R, S)}$$

2. Apply the marginal distribution rule to the unknown vertices. $P(B, \neg R, S)$ has 3 unknown vertices with $2^3 = 8$ possible value assignments.



H = hardware problem
B = bug in lab
M = mailer is running
L = lab is running
R = received email
S = problem solved

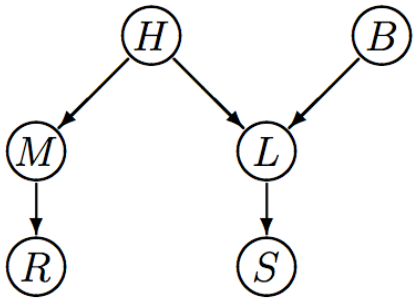
$$\begin{aligned} & P(B, \neg R, S) \\ & = P(B, \neg R, S, H, M, L) \\ & \quad + P(B, \neg R, S, H, M, \neg L) \\ & \quad + P(B, \neg R, S, H, \neg M, L) \\ & \quad + P(B, \neg R, S, H, \neg M, \neg L) \\ & \quad + P(B, \neg R, S, \neg H, M, L) \\ & \quad + P(B, \neg R, S, \neg H, M, \neg L) \\ & \quad + P(B, \neg R, S, \neg H, \neg M, L) \\ & \quad + P(B, \neg R, S, \neg H, \neg M, \neg L) \end{aligned}$$

Evidential Reasoning Example

3. Apply joint distribution rule for Bayesian networks. Here are two examples.

$$
\begin{aligned}
&P(B, \neg R, S, H, M, L) \\
&= P(B)\, P(H) \\
&\quad P(M \mid H)\, P(\neg R \mid M) \\
&\quad \boldsymbol{P(L|H,B)P(S|L)}
\end{aligned}
$$

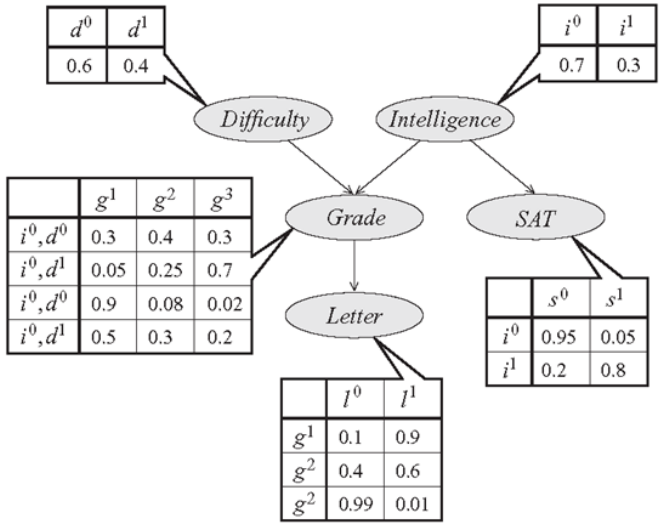
$$
\begin{aligned}
&P(B, \neg R, S, \neg H, M, \neg L) \\
&= P(B)\, P(\neg H) \\
&\quad P(M \mid \neg H)\, P(\neg R \mid M) \\
&\quad \boldsymbol{P(\neg L|\neg H,B)P(S|\neg L)}
\end{aligned}
$$



H = hardware problem
B = bug in lab
M = mailer is running
L = lab is running
R = received email
S = problem solved

Intercausal reasoning

- Recruiter has grade (letter does not matter)
- $P(i^1|g^3)=P(i^1|l^0,g^3)=0.079$
- Recruiter receives high SAT score (leads to dramatic increase)
- $P(i^1|g^3,s^1)=0.578$
- Intuition:
  - High SAT score outweighs poor grade since low intelligence rarely gets good SAT scores
  - Smart students more likely to get Cs in hard classes
- Probability of class is difficult also goes up from
- $P(d^1|g^3)=0.629$ to
- $P(d^1|g^3,s^1)=0.76$



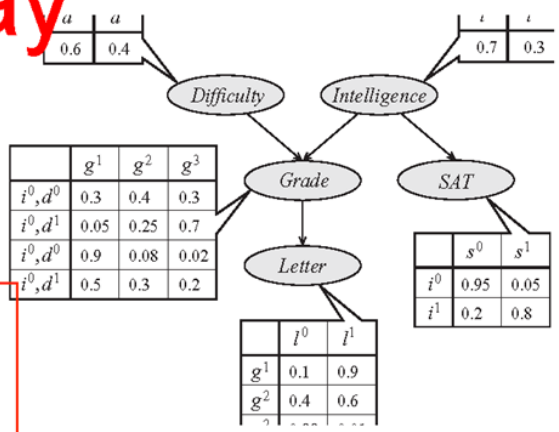
Information about SAT score gave us information about Intelligence

which with Grade told us about difficulty of course

One causal factor for Grade (Intelligence) gives us information about another (Difficulty)

7

Explaining Away



	g^1	g^2	g^3
i^0,d^0	0.3	0.4	0.3
i^0,d^1	0.05	0.25	0.7
i^0,d^0	0.9	0.08	0.02
i^0,d^1	0.5	0.3	0.2

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6

An example:

- Given grade
- $P(i^1|l^0,g^3)=0.079$
- If we observe ECON101 is a hard class
- $P(i^1|g^3,d^1)=0.11$
- We have provided partial explanation for George's performance in ECON101

Another example:

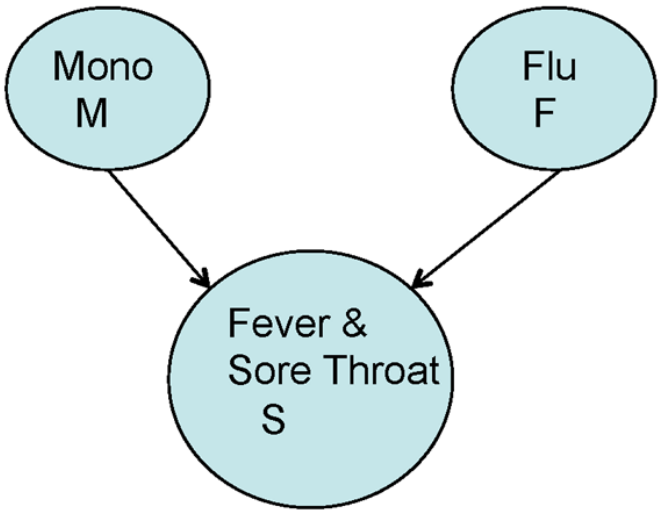
- If George gets a B in ECON101
- $P(i^1|g^2)=0.175$
- If we observe ECON101 is a hard class
- $P(i^1|g^2,d^1)=0.34$
- We have explained away the poor grade via the difficulty of the class

Explaining away is one type of intercausal reasoning

- Different causes of the same effect can interact
- All determined by probability calculation rather than heuristics

Intercausal Reasoning is Common in Human Reasoning

Another example of explaining away



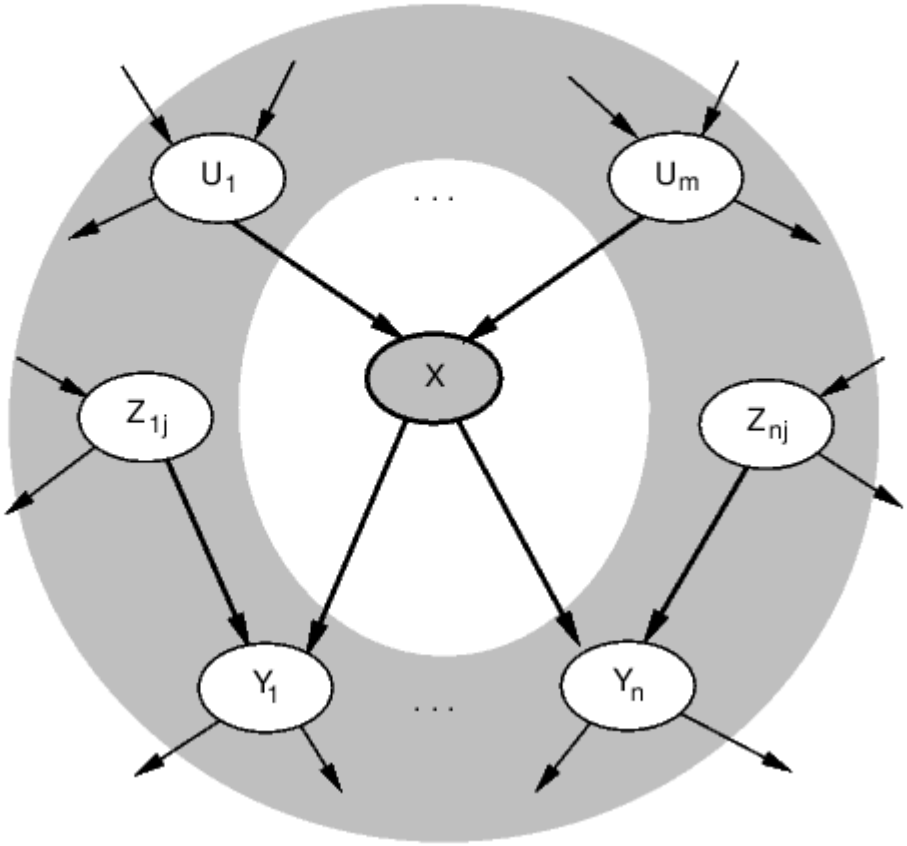
- Binary Variables
- Fever & Sore Throat can be caused by mono and flu
- When flu is diagnosed probability of mono is reduced (although mono could still be present)
- It provides an alternative explanation of symptoms
- $P(m^1|s^1) > P(m^1|s^1,f^1)$

9

Constructing Belief Networks

The Markov Blanket

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



Constructing Belief Networks

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

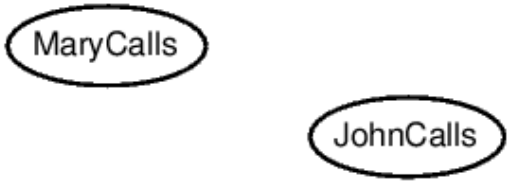
1. Choose an ordering of variables $X_1, \ldots, X_n$
2. For $i = 1$ to n
   add X_i to the network
   select parents from $X_1, \ldots, X_{i-1}$ such that
   $\mathbf{P}(X_i|Parents(X_i)) = \mathbf{P}(X_i|X_1, \ldots, X_{i-1})$

This choice of parents guarantees the global semantics:

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

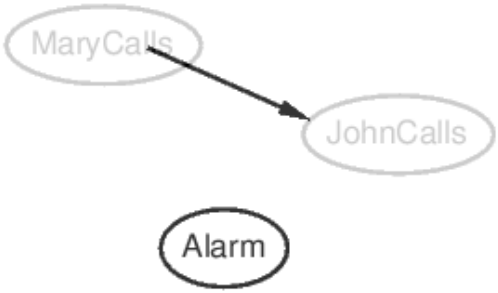
Example

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



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

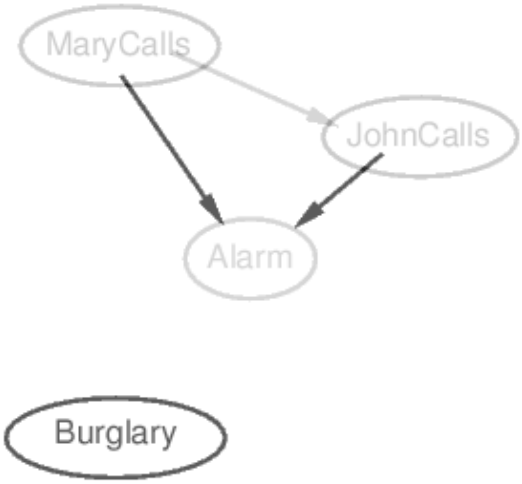
Suppose we choose the ordering M, J, A, B, E



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

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

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



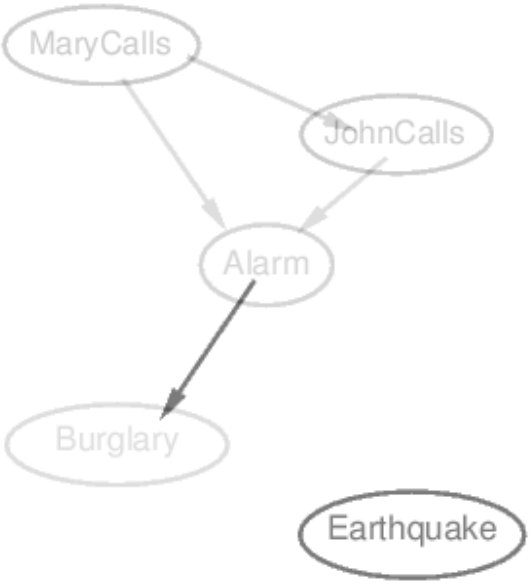
$P(J|M) = P(J)$? No

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

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

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

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

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

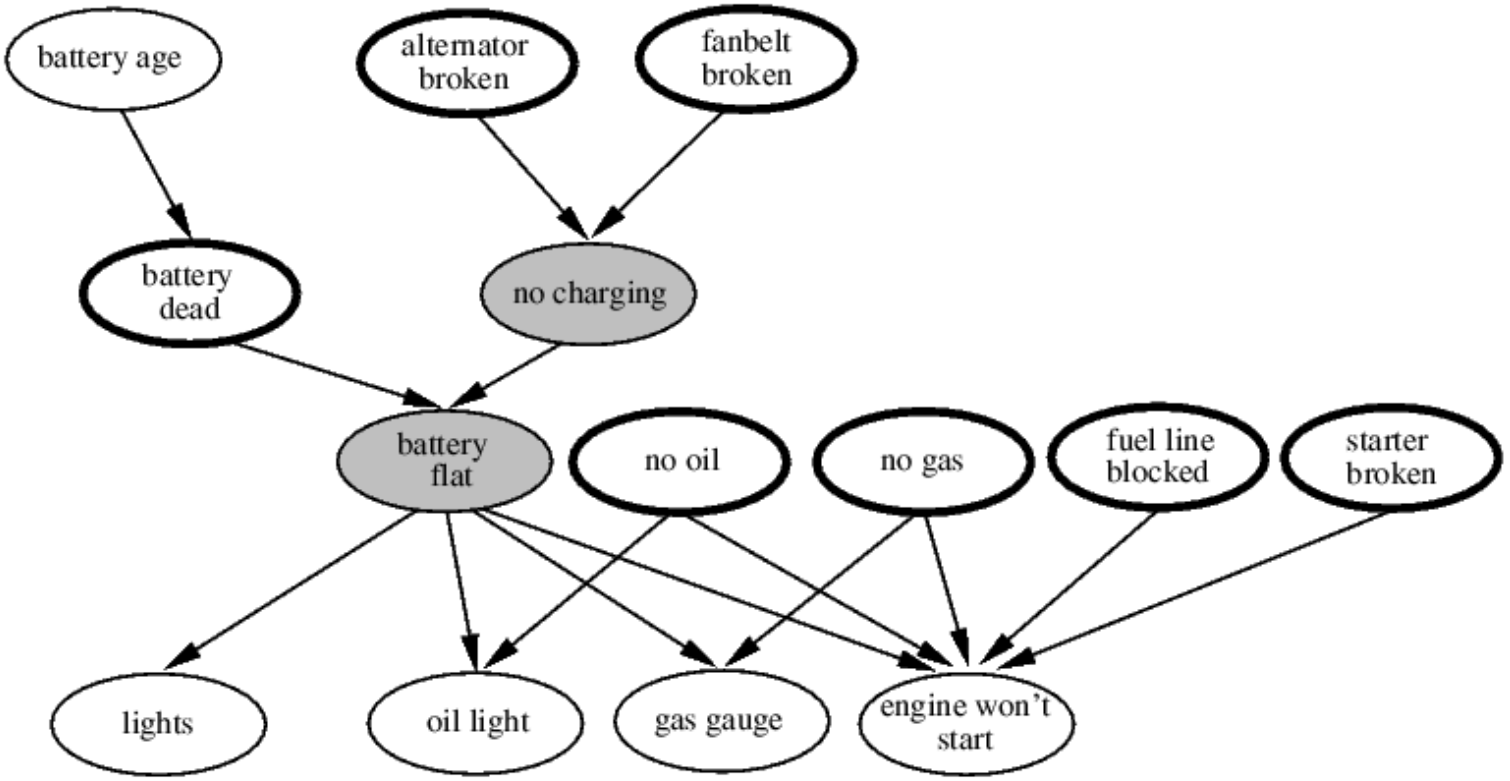
$P(E|B, A, J, M) = P(E|A, B)$?

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

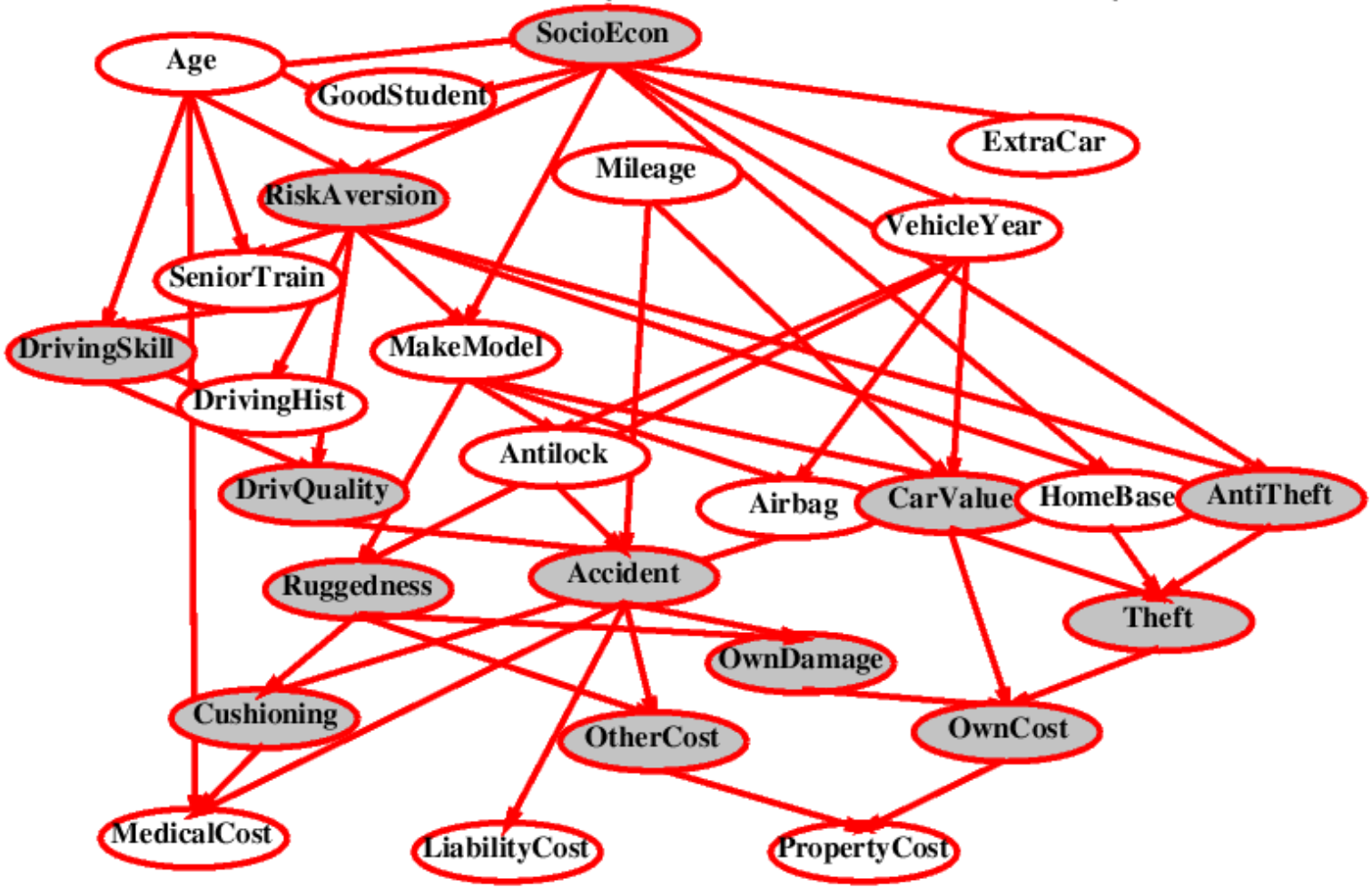
Application: Car Diagnosis

Initial evidence: engine won't start
Testable variables (thin ovals), diagnosis variables (thick ovals)
Hidden variables (shaded) ensure sparse structure, reduce parameters



Application: Car Insurance

Predict claim costs (medical, liability, property)
given data on application form (other unshaded nodes)



Conditional Distributions (for Compactness)

CPT grows exponentially with no. 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

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

Compact conditional distributions

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

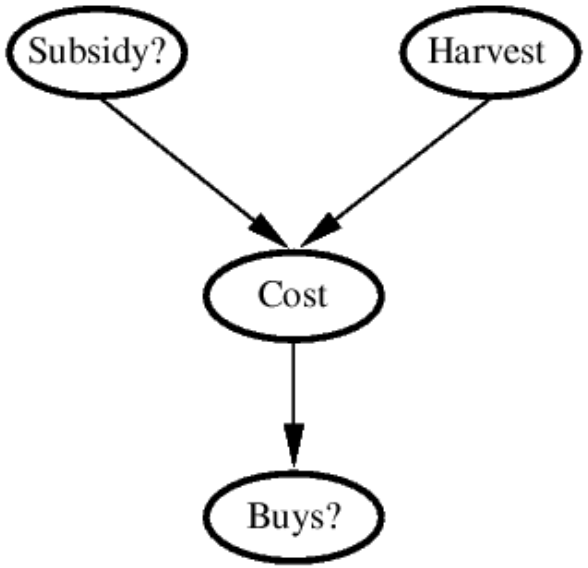
$$\Rightarrow P(X|U_1 \ldots U_j, \neg U_{j+1} \ldots \neg U_k) = 1 - \prod_{i=1}^{j} q_i$$

$Cold$	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

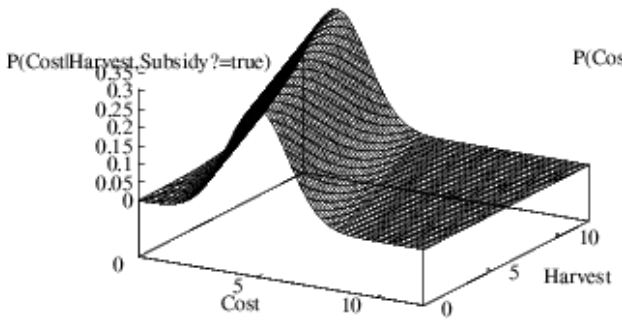
Need one conditional density function for child variable given continuous parents, for each possible assignment to discrete parents

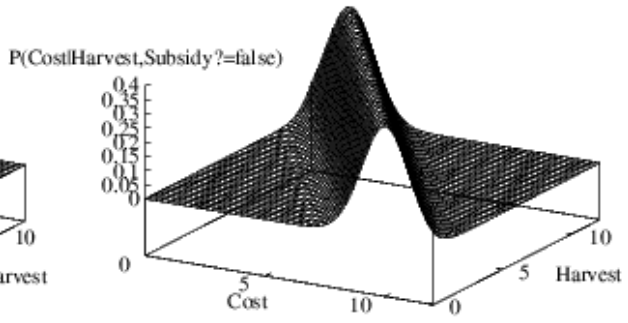
Most common is the linear Gaussian model, e.g.,:

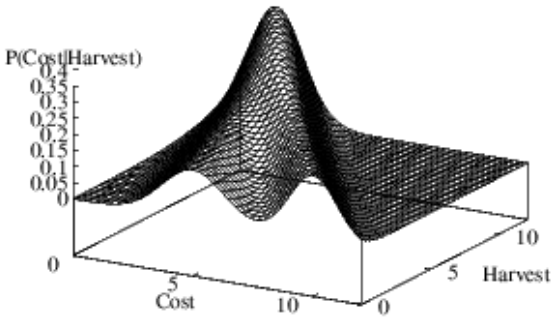
$$\begin{aligned} & P(Cost = c | Harvest = h, Subsidy? = true) \\ &= N(a_t h + b_t, \sigma_t)(c) \\ &= \frac{1}{\sigma_t \sqrt{2\pi}} exp\left(-\frac{1}{2}\left(\frac{c - (a_t h + b_t)}{\sigma_t}\right)^2\right) \end{aligned}$$

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







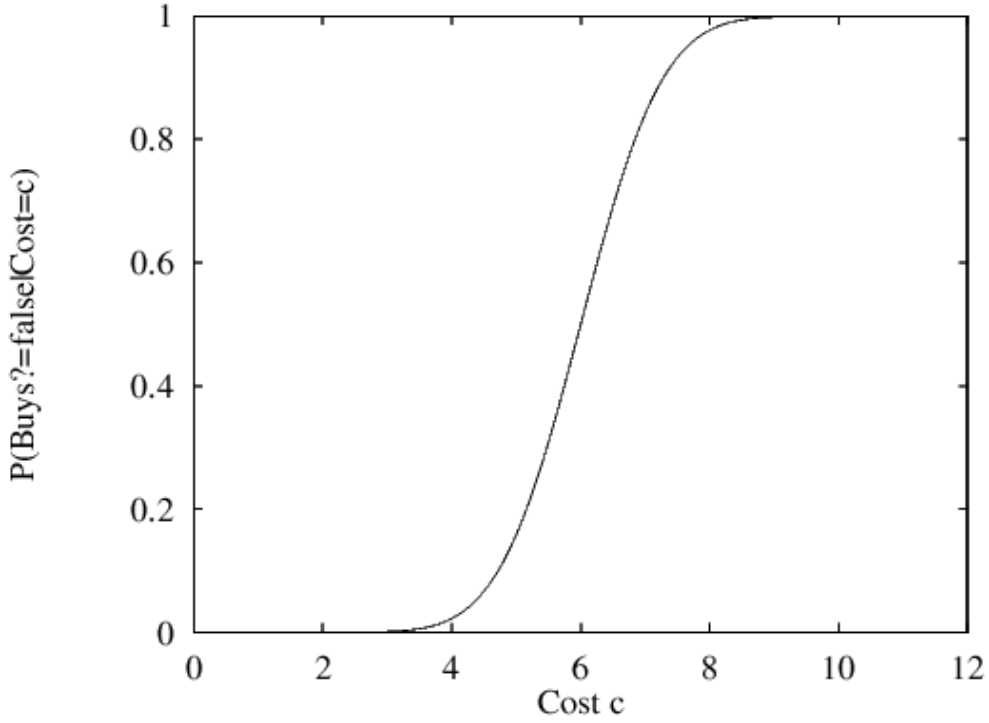
All-continuous network with LG distributions

$\Rightarrow$ full joint 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 & Continuous Parents

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



Probit distribution uses integral of Gaussian:

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

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

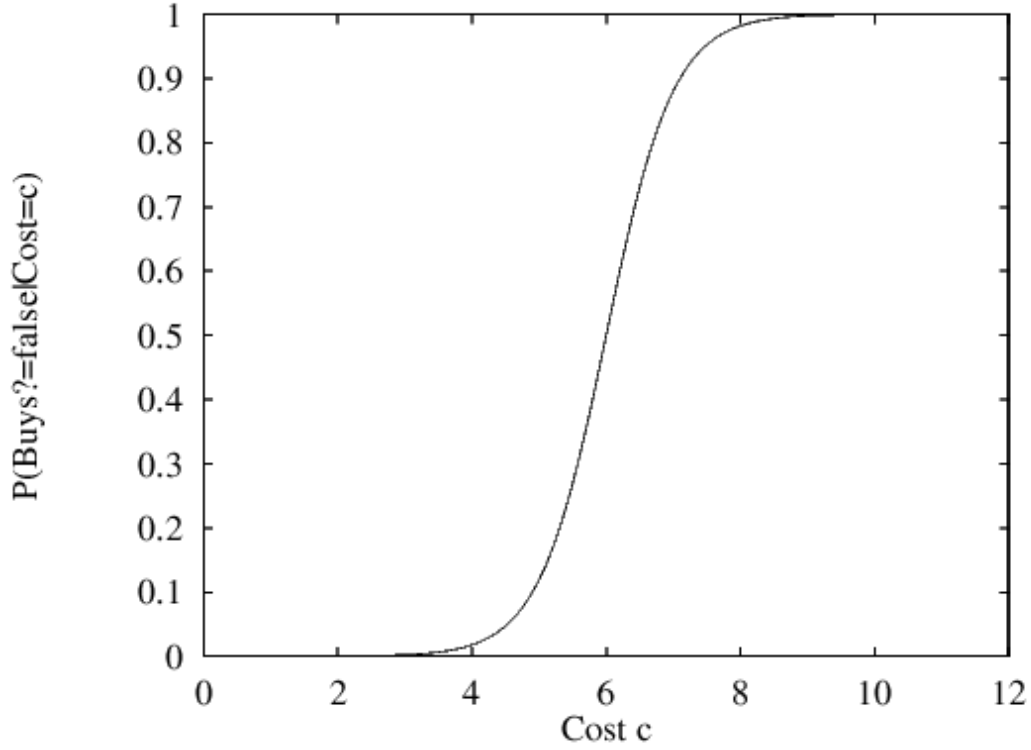
Can view as hard threshold whose location is subject to noise

Discrete Variable & 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:



Belief (Bayesian) Networks

- Motivation
- Conditional Independence
- Syntax and Semantics
- Reasoning with Belief Networks
- Construction of Belief Networks
- Inference Algorithms
  - Exact Inferences
    - By Enumeration
    - By Variable Eliminations
  - Approximate inference

Inference Tasks

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

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

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?

Enumeration Algorithm

The Key Ideas Are:

Brute force calculation of $P(H \mid E)$ is done by:

1. Apply the conditional probability rule.

$$P(H \mid E) = P(H \wedge E) \,/\, P(E)$$

2. Apply the marginal distribution rule to the unknown vertices $\mathbf{U}$.

$$P(H \wedge E) = \sum_{\mathbf{U}=\mathbf{u}} P(H \wedge E \wedge \mathbf{U} = \mathbf{u})$$

3. Apply joint distribution rule for Bayesian networks.

$$\mathbf{P}(X_1, \ldots, X_n) = \prod_{i=1}^{n} \mathbf{P}(X_i \mid Parents(X_i))$$

Enumeration Algorithm

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

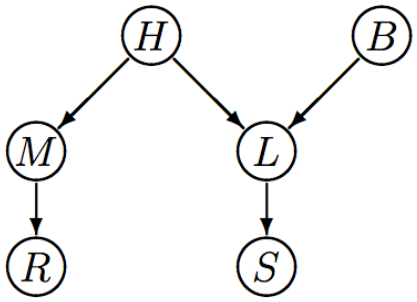
ENUMERATIONASK(X,**e**,bn) **returns** a distribution over X
inputs: X, the query variable
e, evidence specified as an event
bn, a belief network specifying joint distribution $\mathbf{P}(X_1, \ldots, X_n)$

```
Q(X) ← a distribution over X
for each value x_i of X do
    extend e with value x_i for X
    Q(x_i) ← ENUMERATEALL(VARS[bn],e)
return NORMALIZE(Q(X))
```

ENUMERATEALL($vars$,**e**) **returns** a real number

```
if EMPTY?(vars) then return 1.0
else do
    Y ← FIRST(vars)
    if Y has value y in e
        then return P(y | Pa(Y)) × ENUMERATEALL(REST(vars),e)
        else return Σ_y P(y | Pa(Y)) × ENUMERATEALL(REST(vars),e_y)
            where e_y is e extended with Y = y
```

Enumeration Example (1)



H = hardware problem
B = bug in lab
M = mailer is running
L = lab is running
R = received email
S = problem solved

Query: *P(B|~R,S)*

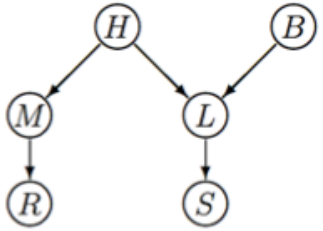
Each node needs a probability table. Size of table depends on number of parents.

$\mathbf{P}(H)$	
$True$	$False$
0.01	0.99

	$\mathbf{P}(M \mid H)$	
H	$True$	$False$
$True$	0.1	0.9
$False$	0.99	0.01

		$\mathbf{P}(L \mid H, B)$	
H	B	$True$	$False$
$True$	$True$	0.01	0.99
$True$	$False$	0.1	0.9
$False$	$True$	0.02	0.98
$False$	$False$	1.0	0.0

Enumeration Example (1)



H = hardware problem
B = bug in lab
M = mailer is running
L = lab is running
R = received email
S = problem solved

Calculate $P(B \mid \neg R, S)$ in the buggy lab example.

1. Apply the conditional probability rule.

$$P(B \mid \neg R, S) = \frac{P(B, \neg R, S)}{P(\neg R, S)} \quad = \alpha\, P(B, \sim R, S)$$

2. Apply the marginal distribution rule to the unknown vertices. $P(B, \neg R, S)$ has 3 unknown vertices with $2^3 = 8$ possible value assignments.

$$\begin{aligned} & P(B, \neg R, S) \\ &= P(B, \neg R, S, H, M, L) \\ &\quad + P(B, \neg R, S, H, M, \neg L) \\ &\quad + P(B, \neg R, S, H, \neg M, L) \\ &\quad + P(B, \neg R, S, H, \neg M, \neg L) \\ &\quad + P(B, \neg R, S, \neg H, M, L) \\ &\quad + P(B, \neg R, S, \neg H, M, \neg L) \\ &\quad + P(B, \neg R, S, \neg H, \neg M, L) \\ &\quad + P(B, \neg R, S, \neg H, \neg M, \neg L) \end{aligned}$$

P(~R,S)

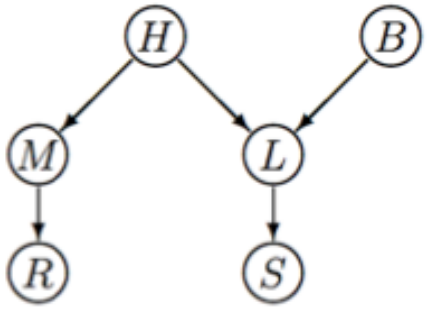
=P(B,~R,S,H,M,L) *+P(~B,~R,S,H,M,L)*
+P(B,~R,S,H,M,~L) *+P(~B,~R,S,H,M,~L)*
+P(B,~R,S,H,~M,L) *+P(~B,~R,S,H,~M,L)*
+P(B,~R,S,H,~M,~L) *+P(~B,~R,S,H,~M,~L)*
+P(B,~R,S,~H,M,L) *+P(~B,~R,S,~H,M,L)*
+P(B,~R,S,~H,~M,L) *+P(~B,~R,S,~H,~M,L)*
+P(B,~R,S,~H,~M,~L) *+P(~B,~R,S,~H,~M,~L)*

Enumeration Example (1)

3. Apply joint distribution rule for Bayesian networks. Here are two examples.

$$P(B, \neg R, S, H, M, L)$$
$$= P(B)\, P(H)$$
$$P(M \mid H)\, P(\neg R \mid M)$$
$$P(L \mid H, \neg B)\, P(S \mid L)$$

$$P(B, \neg R, S, \neg H, M, \neg L)$$
$$= P(B)\, P(\neg H)$$
$$P(M \mid \neg H)\, P(\neg R \mid M)$$
$$P(\neg L \mid \neg H, B)\, P(S \mid \neg L)$$



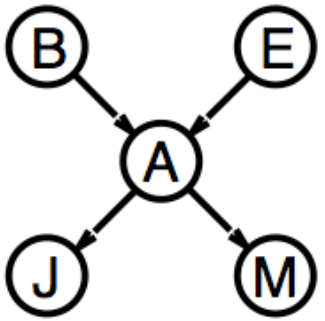
H = hardware problem
B = bug in lab
M = mailer is running
L = lab is running
R = received email
S = problem solved

Enumeration Example (2)

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

Simple query on the burglary network:

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

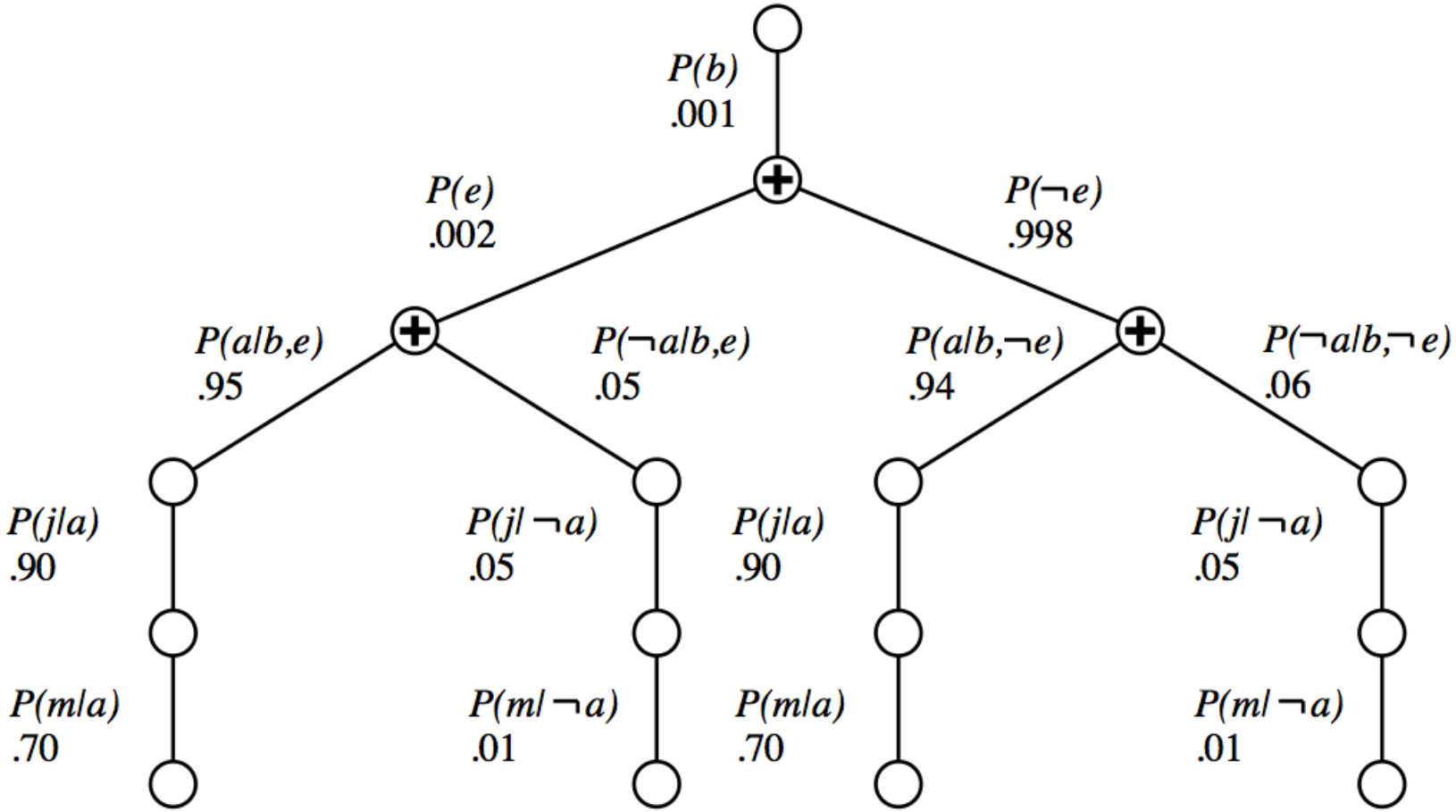


Rewrite full joint entries using product of CPT entries:

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

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

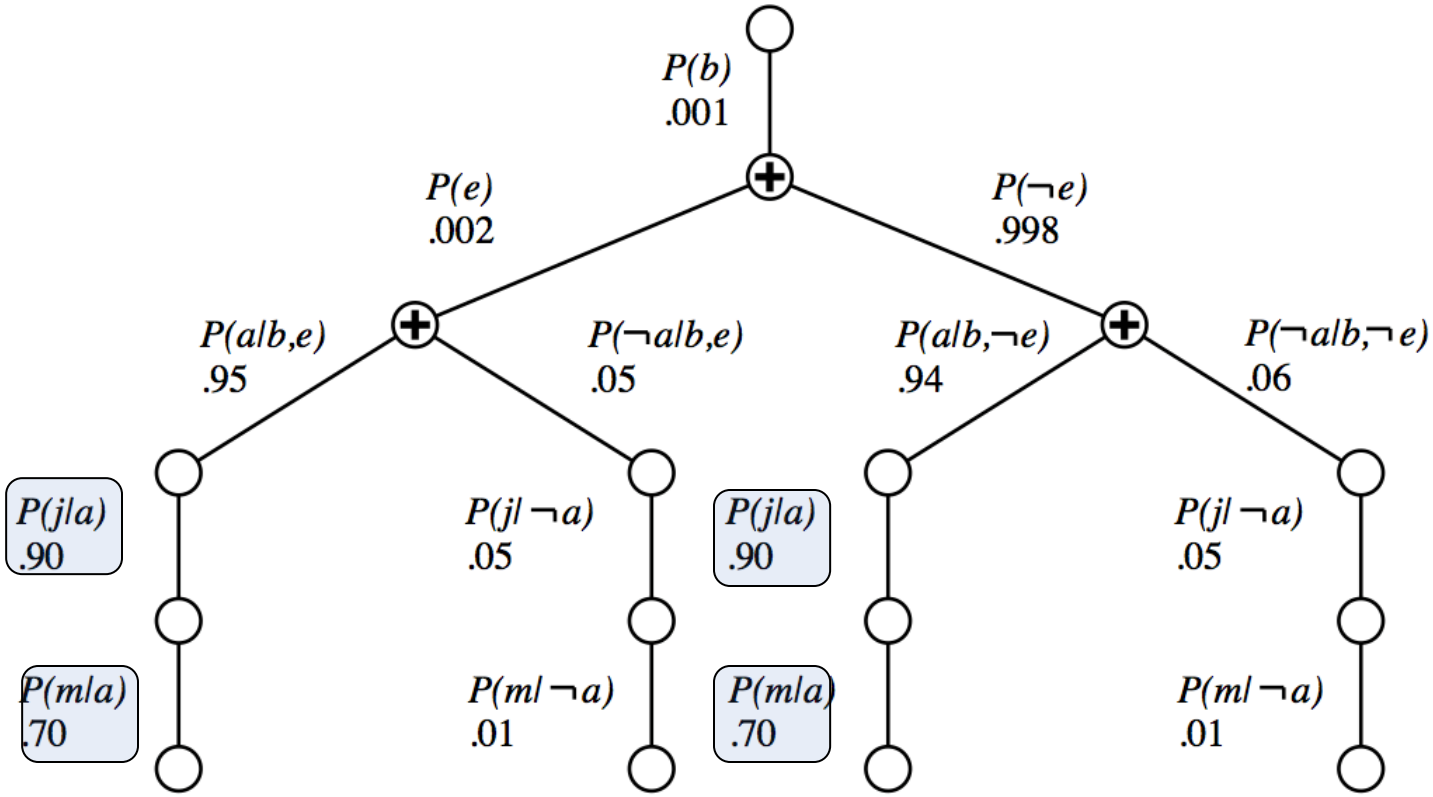
Enumeration Example (2)



Enumeration is inefficient: repeated computation

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

Enumeration Example (2)



Enumeration is inefficient: repeated computation

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

Variable Elimination Algorithm

Enumeration is inefficient: repeated computation
e.g., computes $P(J=true|a)P(M=true|a)$ for each value of e

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

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

Variable Elimination: Two Basic Operations

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

Summing out a variable from a product of factors: move any constant factors outside the summation:

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

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

Pointwise Product

- Pointwise multiplication of factors when variable is summed out or at last step
- 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	$f_1(a, b)$
T	T	.3
T	F	.7
F	T	.9
F	F	.1

b	c	$f_2(b, c)$
T	T	.2
T	F	.8
F	T	.6
F	F	.4

a	b	c	$f(a, b, c)$
T	T	T	.3 * .2
T	T	F	.3 * .8
T	F	T	.7 * .6
T	F	F	.7 * .4
F	T	T	.9 * .2
F	T	F	.9 * .8
F	F	T	.1 * .6
F	F	F	.1 * .4

Summing Out

- Summing out a variable from a product of factors
  - Move any constant factors outside the summation
  - Add up sub-matrices in pointwise product of remaining factors

$$\sum_x f_1 \times \cdots \times f_k = f_1 \times \cdots \times f_l \times \left(\sum_x f_{l+1} \times \cdots \times f_k \right) = f_1 \times \cdots \times f_l \times f_{\overline{X}}$$

- E.g. $\sum_a f(a, b, c) = f_{\overline{a}}(b, c)$:

a	b	c	$f(a, b, c)$	b	c	$f_{\overline{a}}(b, c)$
T	T	T	.3 * .2	T	T	.3 * .2 + .9 * .2
T	T	F	.3 * .8	T	F	.3 * .8 + .9 * .8
T	F	T	.7 * .6	F	T	.7 * .6 + .1 * .6
T	F	F	.7 * .4	F	F	.7 * .4 + .1 * .4
F	T	T	.9 * .2			
F	T	F	.9 * .8			
F	F	T	.1 * .6			
F	F	F	.1 * .4			

Variable Elimination Algorithm

function ELIMINATIONASK(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 $\mathbf{P}(X_1, \ldots, X_n)$

  if $X \in$ **e** **then return** observed point distribution for X
  $factors \leftarrow [\,]$; $vars \leftarrow$ REVERSE(VARS[bn])
  for each var **in** $vars$ **do**
    $factors \leftarrow [\text{MAKEFACTOR}(var, \mathbf{e}) | factors]$
    if var is a hidden variable **then** $factors \leftarrow$ SUMOUT(var,$factors$)
  return NORMALIZE(POINTWISEPRODUCT($factors$))

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 $O(d^k n)$

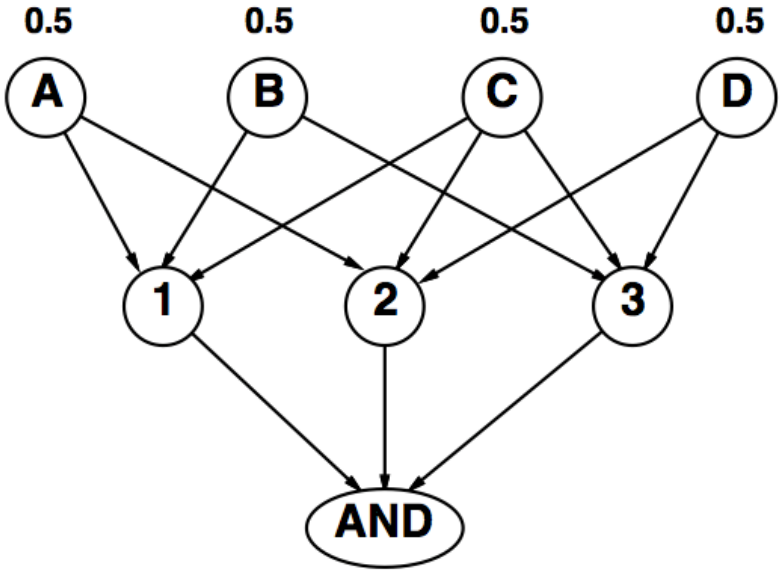
Multiply connected networks:

- can reduce 3SAT to exact inference $\Rightarrow$ NP-hard
- equivalent to *counting* 3SAT models $\Rightarrow$ #P-complete

1. A v B v C

2. C v D v ~A

3. B v C v ~D



Belief (Bayesian) Networks

- Motivation
- Conditional Independence
- Syntax and Semantics
- Reasoning with Belief Networks
- Construction of Belief Networks
- Inference Algorithms
  - Exact Inferences
  - Approximate Inference
    - By Stochastic Simulation
    - By Markov Chain Monte Carlo

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

Outline:

- Sampling from an empty network
- Rejection sampling: reject samples disagreeing with evidence
- Likelihood weighting: use evidence to weight samples
- MCMC: sample from a stochastic process whose stationary distribution is the true posterior

Sampling from an empty network (1)

function PRIORSAMPLE(bn) **returns** an event sampled from $\mathbf{P}(X_1, \ldots, X_n)$ specified by bn
  $\mathbf{x} \leftarrow$ an event with n elements
  for $i = 1$ **to** n **do**
    $x_i \leftarrow$ a random sample from $\mathbf{P}(X_i \mid Parents(X_i))$
  return x

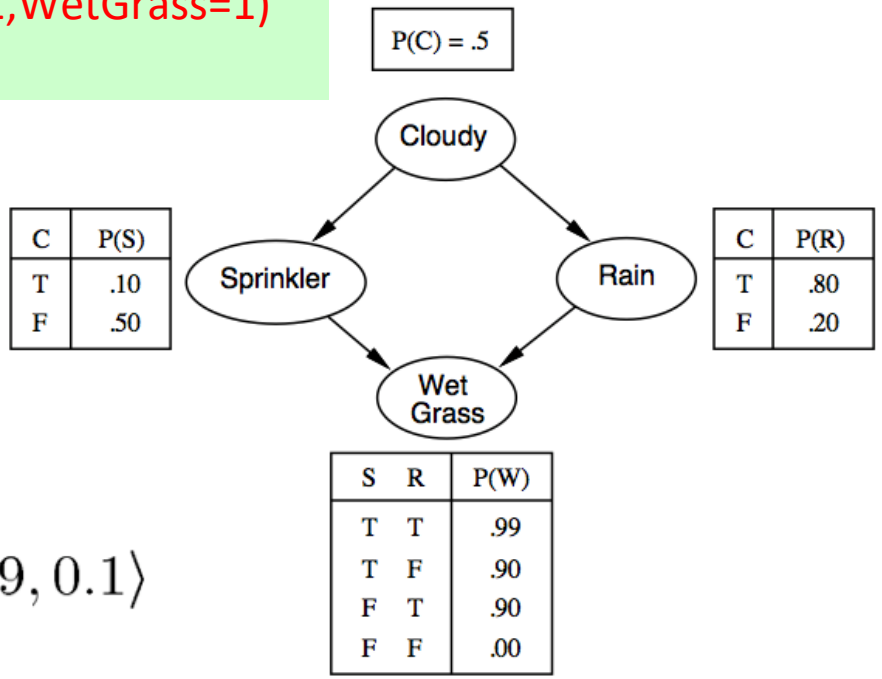
P(Cloudy=1,Sprinkler=0,Rain=1,WetGrass=1)
=0.5*0.9*0.8*0.9=0.324

$$\mathbf{P}(Cloudy) = \langle 0.5, 0.5 \rangle$$
sample $\rightarrow$ *true*

$$\mathbf{P}(Sprinkler|Cloudy) = \langle 0.1, 0.9 \rangle$$
sample $\rightarrow$ *false*

$$\mathbf{P}(Rain|Cloudy) = \langle 0.8, 0.2 \rangle$$
sample $\rightarrow$ *true*

$$\mathbf{P}(WetGrass|\neg Sprinkler, Rain) = \langle 0.9, 0.1 \rangle$$
sample $\rightarrow$ *true*



Sampling from an empty network (2)

function PRIORSAMPLE(bn) **returns** an event sampled from $\mathbf{P}(X_1, \ldots, X_n)$ specified by bn
  x ← an event with n elements
  for $i = 1$ **to** n **do**
    $x_i \leftarrow$ a random sample from $\mathbf{P}(X_i \mid Parents(X_i))$
  return x

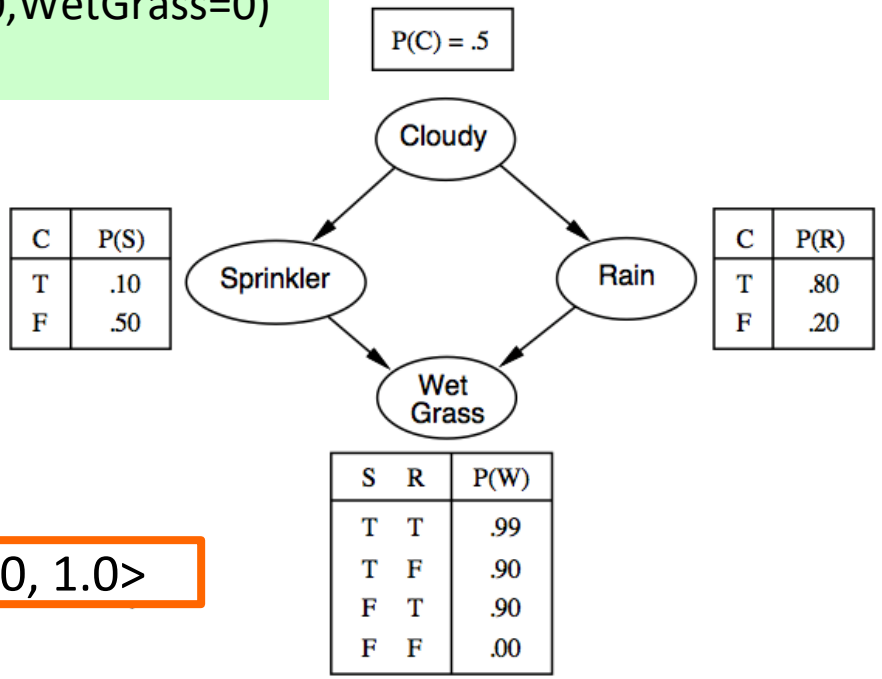
P(Cloudy=0,Sprinkler=0,Rain=0,WetGrass=0)
=0.5*0.5*0.8*1.0=0.20

$\mathbf{P}(Cloudy) = \langle 0.5, 0.5 \rangle$
sample → *false*

$\mathbf{P}(Sprinkler|\overline{Cloudy}) =$ <0.5,0.5>
sample → *false*

$\mathbf{P}(Rain|\overline{Cloudy}) =$ <0.2, 0.8>
sample → *false*

$\mathbf{P}(WetGrass|\neg\overline{Sprinkler}, \overline{Rain}) =$ <0.0, 1.0>
sample → *false*



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

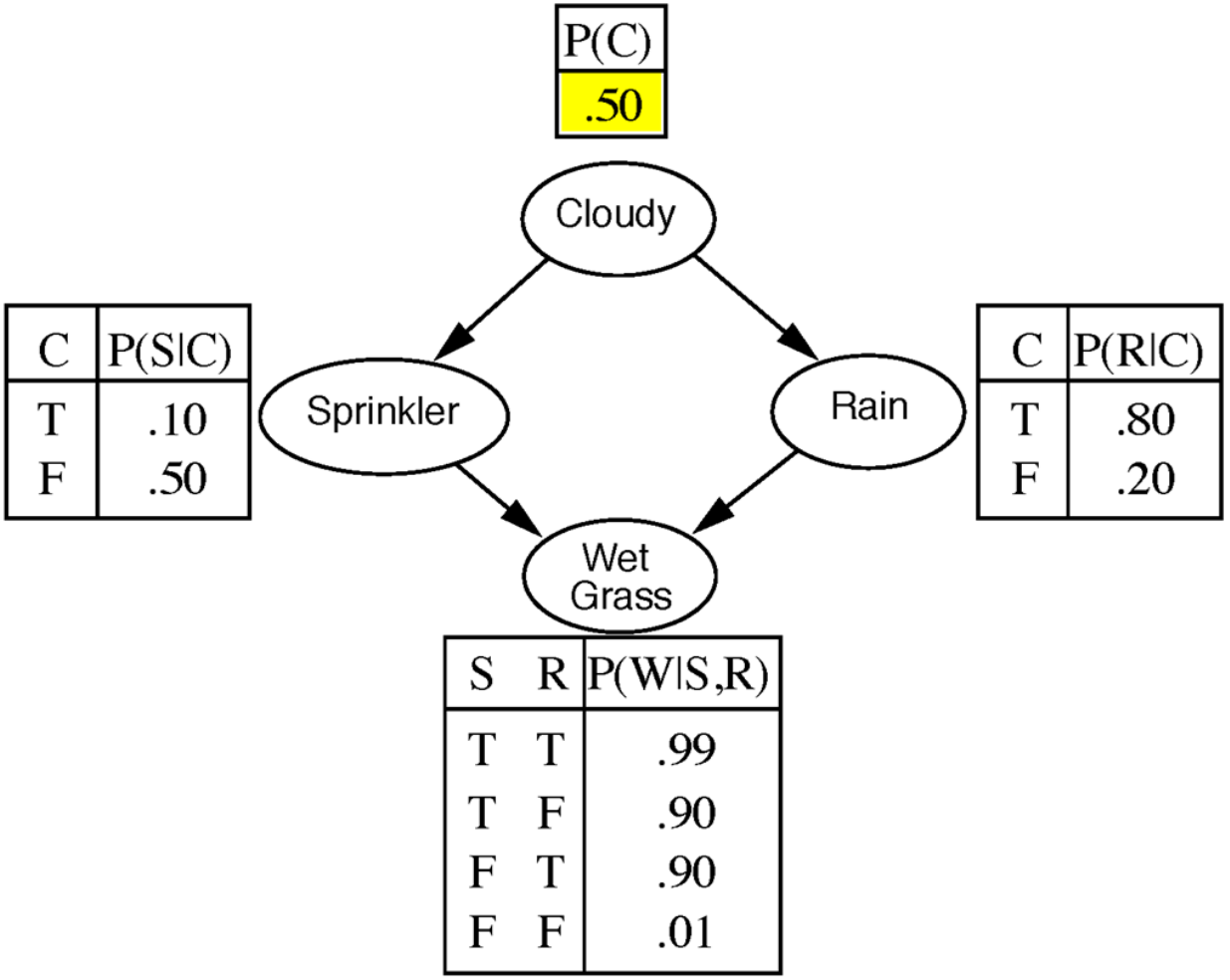
Let $N_{PS}(\mathbf{Y} = \mathbf{y})$ be the number of samples generated for which $\mathbf{Y} = \mathbf{y}$, for any set of variables $\mathbf{Y}$.

Then $\hat{P}(\mathbf{Y} = \mathbf{y}) = N_{PS}(\mathbf{Y} = \mathbf{y})/N$ and

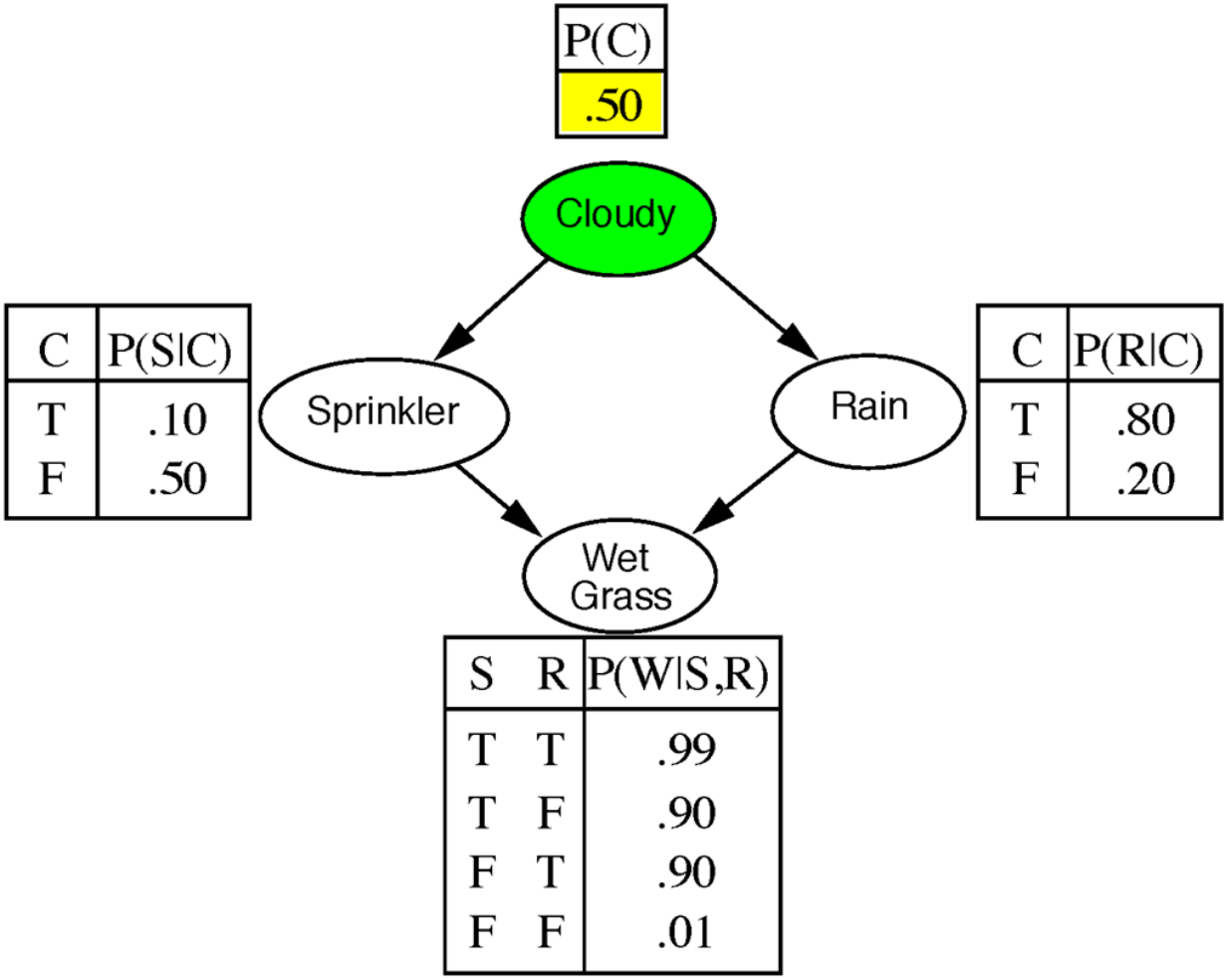
$$\begin{aligned} \lim_{N \to \infty} \hat{P}(\mathbf{Y} = \mathbf{y}) &= \Sigma_{\mathbf{h}} S_{PS}(\mathbf{Y} = \mathbf{y}, \mathbf{H} = \mathbf{h}) \\ &= \Sigma_{\mathbf{h}} P(\mathbf{Y} = \mathbf{y}, \mathbf{H} = \mathbf{h}) \\ &= P(\mathbf{Y} = \mathbf{y}) \end{aligned}$$

That is, estimates derived from PRIORSAMPLE are consistent

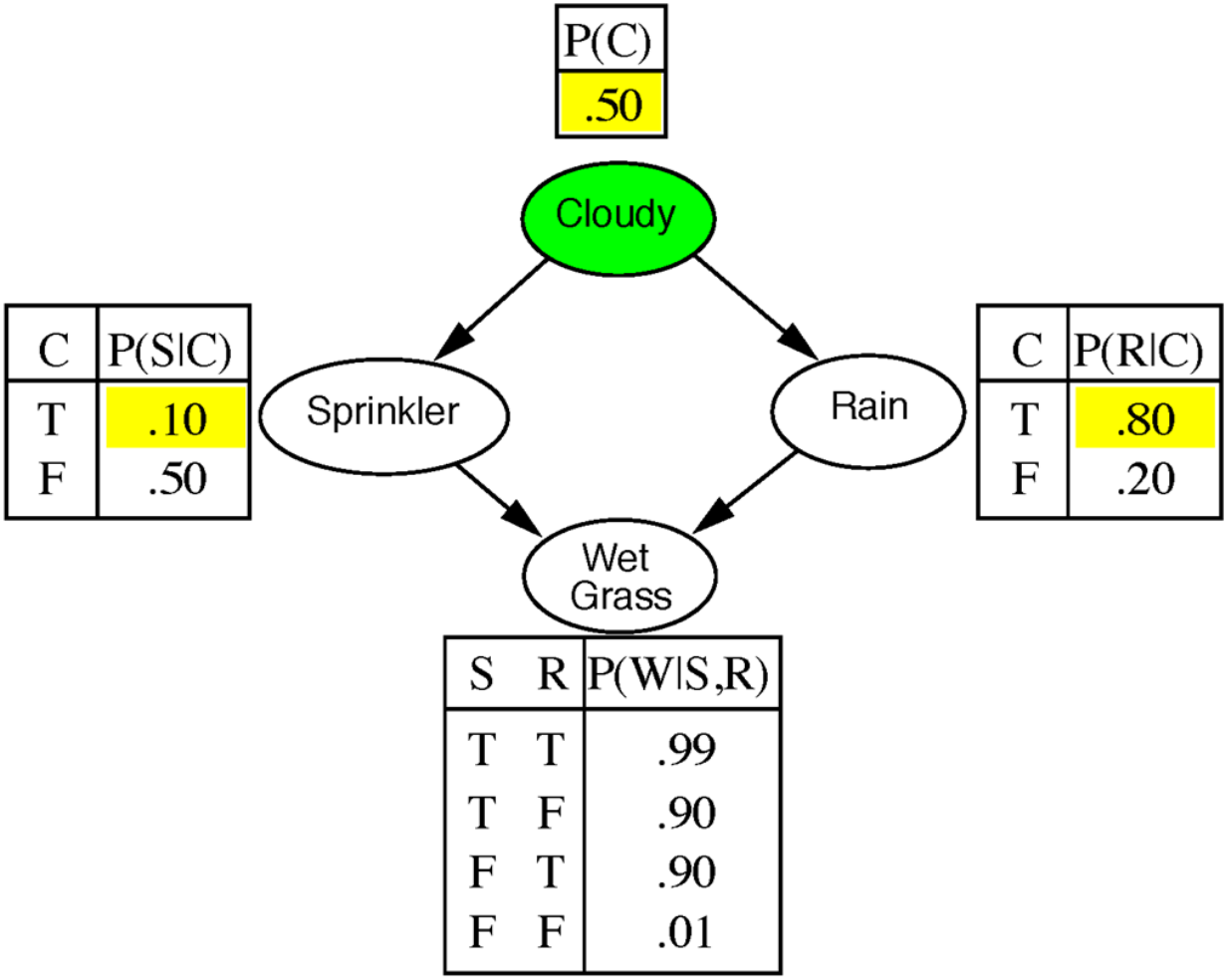
Example


P(C)
.50
Cloudy
C P(S|C)
T .10
F .50
Sprinkler
Rain
C P(R|C)
T .80
F .20
Wet Grass
S R P(W|S,R)
T T .99
T F .90
F T .90
F F .01

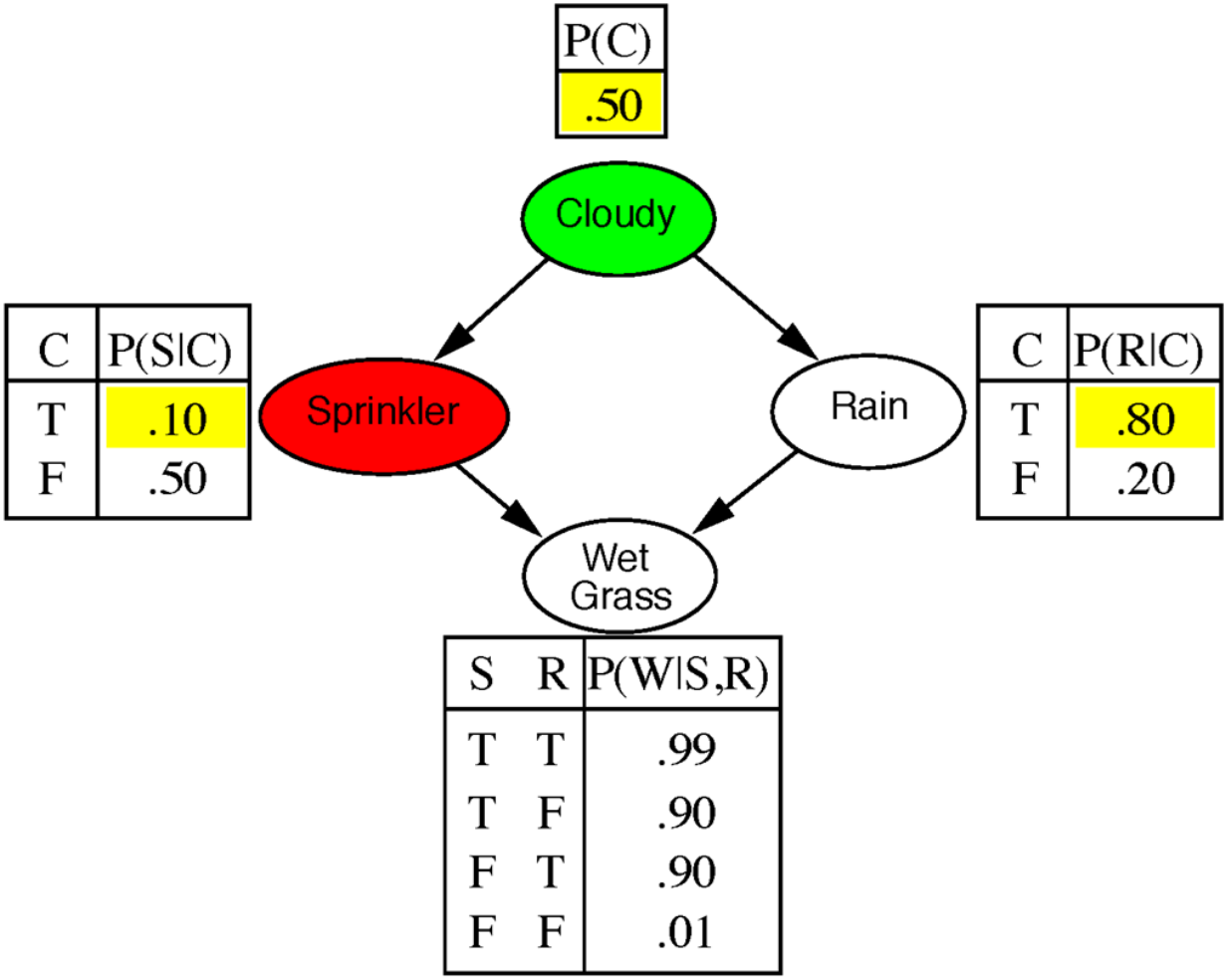
Example


P(C)
.50
Cloudy
C P(S|C)
T .10
F .50
Sprinkler
Rain
C P(R|C)
T .80
F .20
Wet Grass
S R P(W|S,R)
T T .99
T F .90
F T .90
F F .01

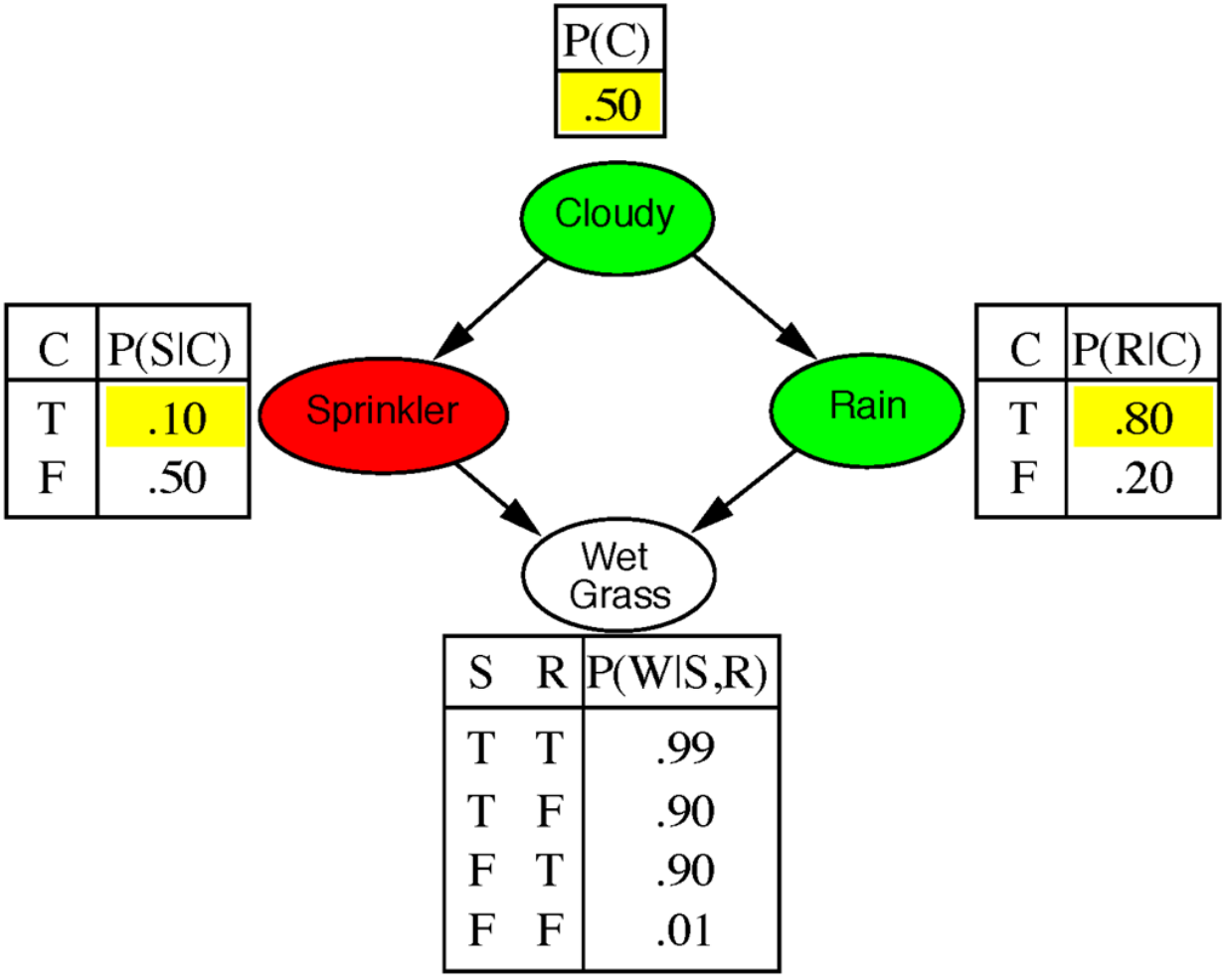
Example


P(C)
.50
Cloudy
C P(S|C)
T .10
F .50
Sprinkler
Rain
C P(R|C)
T .80
F .20
Wet Grass
S R P(W|S,R)
T T .99
T F .90
F T .90
F F .01

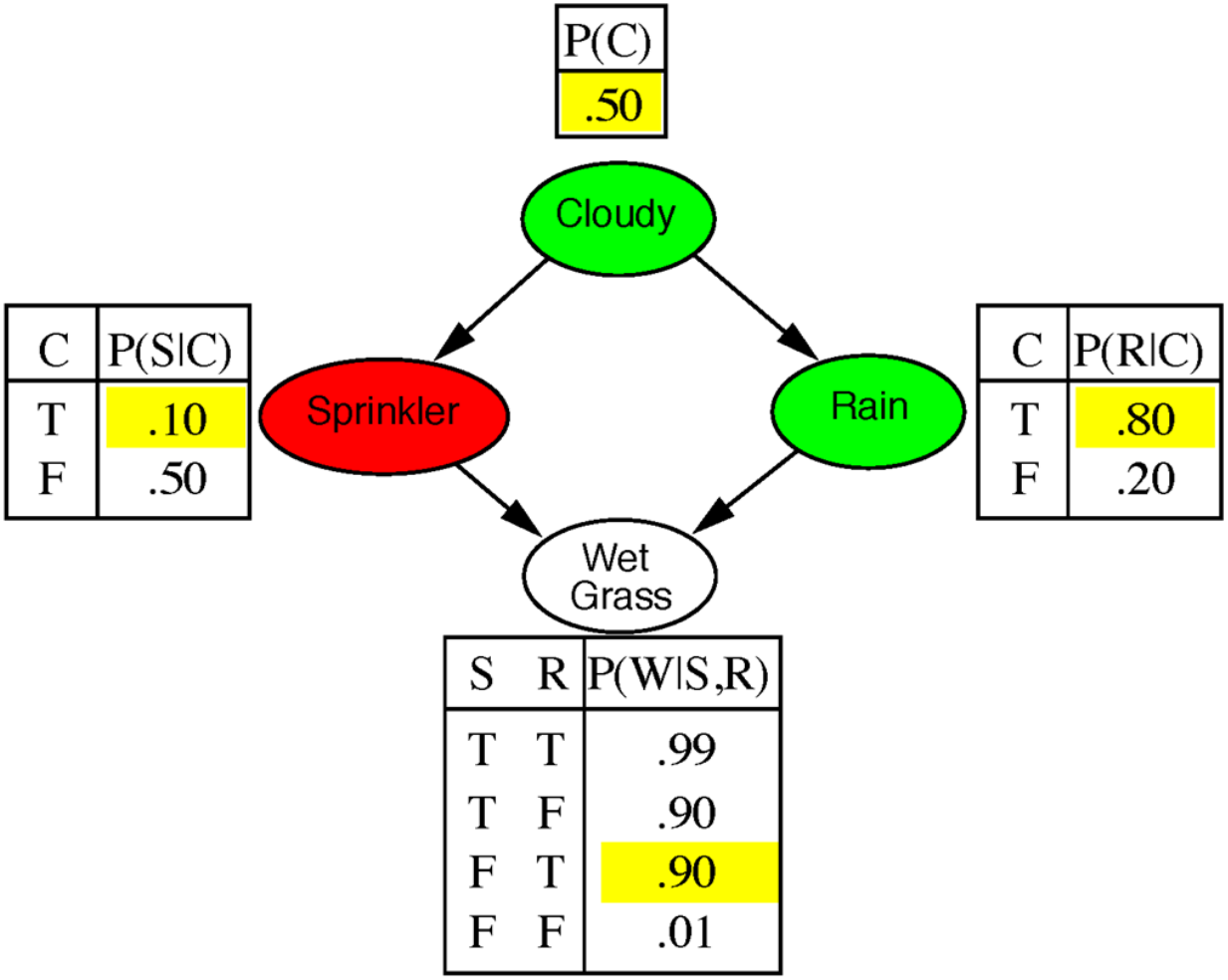
Example


P(C)
.50
Cloudy
C P(S|C)
T .10
F .50
Sprinkler
Rain
C P(R|C)
T .80
F .20
Wet Grass
S R P(W|S,R)
T T .99
T F .90
F T .90
F F .01

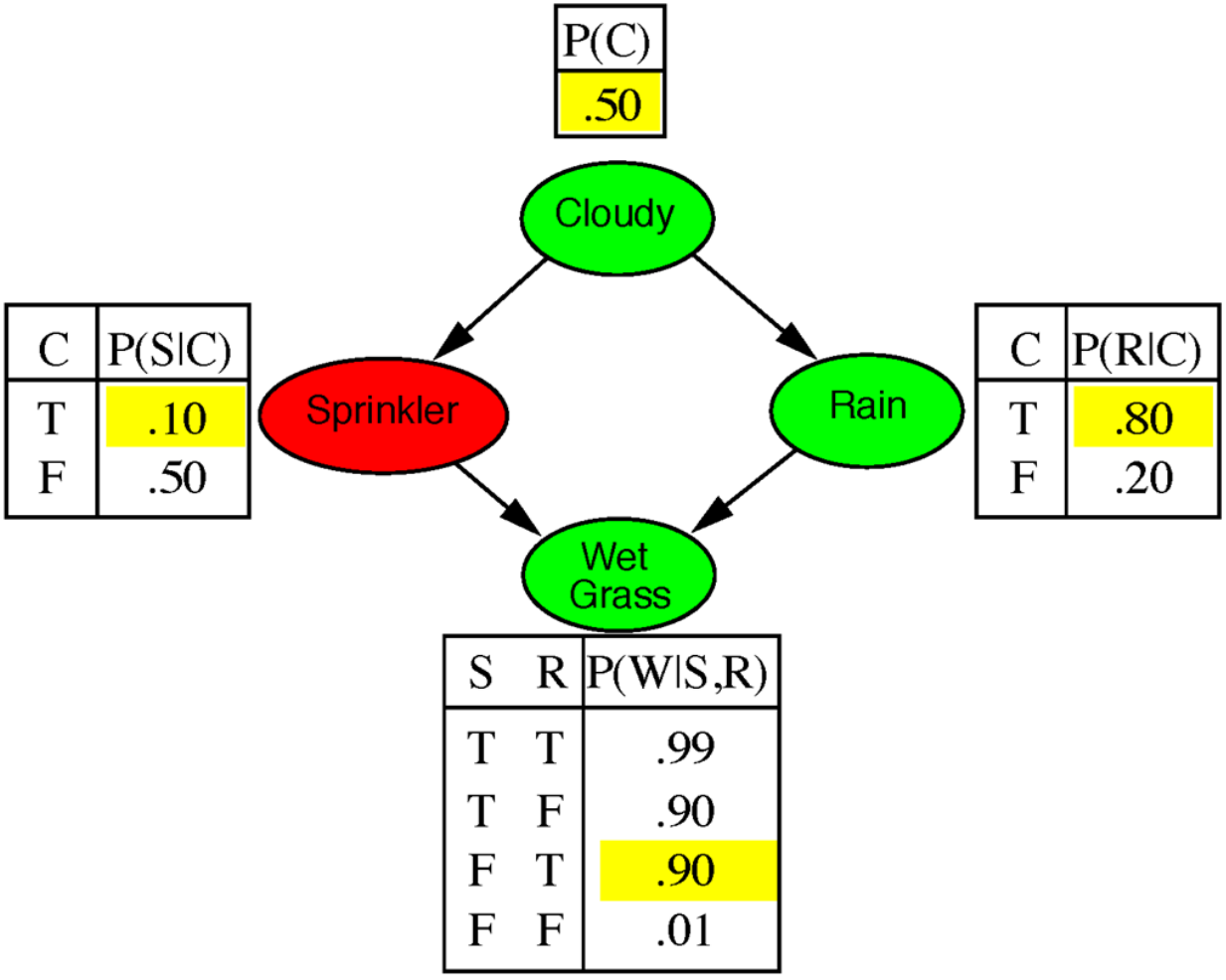
Example


P(C)
.50
Cloudy
C P(S|C)
T .10
F .50
Sprinkler
Rain
C P(R|C)
T .80
F .20
Wet Grass
S R P(W|S,R)
T T .99
T F .90
F T .90
F F .01

Example


P(C)
.50
Cloudy
C P(S|C)
T .10
F .50
Sprinkler
Rain
C P(R|C)
T .80
F .20
Wet Grass
S R P(W|S,R)
T T .99
T F .90
F T .90
F F .01

Example


P(C)
.50
Cloudy
C P(S|C)
T .10
F .50
Sprinkler
Rain
C P(R|C)
T .80
F .20
Wet Grass
S R P(W|S,R)
T T .99
T F .90
F T .90
F F .01

Rejection Sampling

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

```
function REJECTIONSAMPLING(X,e,bn,N) returns an approximation to P(X|e)
    N[X] ← a vector of counts over X, initially zero
    for j = 1 to N do
        x ← PRIORSAMPLE(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

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

Hence rejection sampling returns consistent posterior estimates

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

Likelihood Weighting

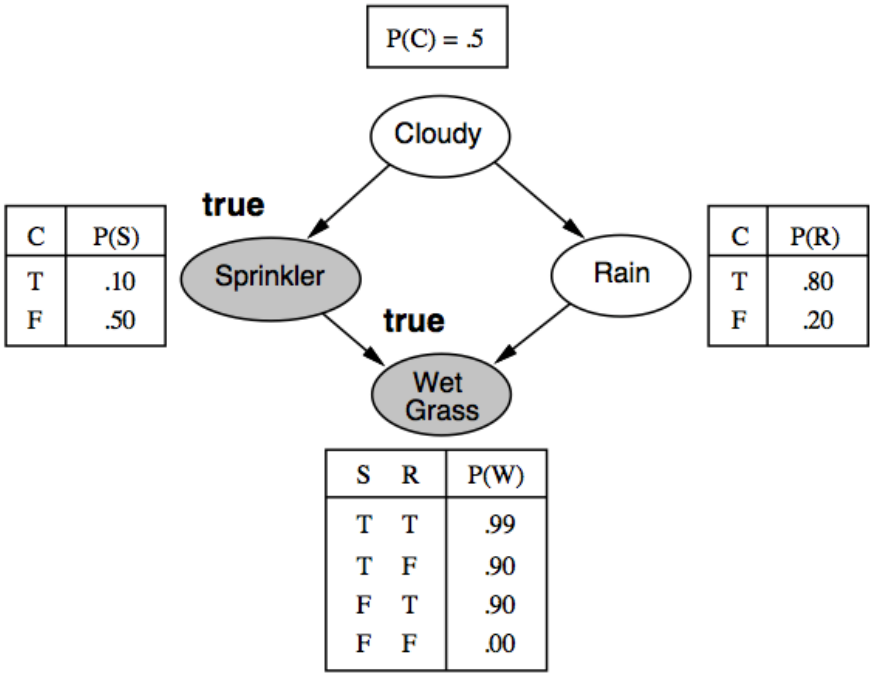
Idea: fix evidence variables, sample only nonevidence variables, and weight each sample by the likelihood it accords the evidence

function WEIGHTEDSAMPLE(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 $w \leftarrow w \times P(X_i = x_i \mid Parents(X_i))$
      else $x_i \leftarrow$ a random sample from $\mathbf{P}(X_i \mid Parents(X_i))$
  return **x**, w

function LIKELIHOODWEIGHTING(X,**e**,bn,N) **returns** an approximation to $P(X|\mathbf{e})$
  $\mathbf{W}[X] \leftarrow$ a vector of weighted counts over X, initially zero
  for $j = 1$ to N **do**
    x,$w \leftarrow$ WEIGHTEDSAMPLE(bn)
    $\mathbf{W}[x] \leftarrow \mathbf{W}[x] + w$ where x is the value of X in **x**
  return NORMALIZE($\mathbf{W}[X]$)

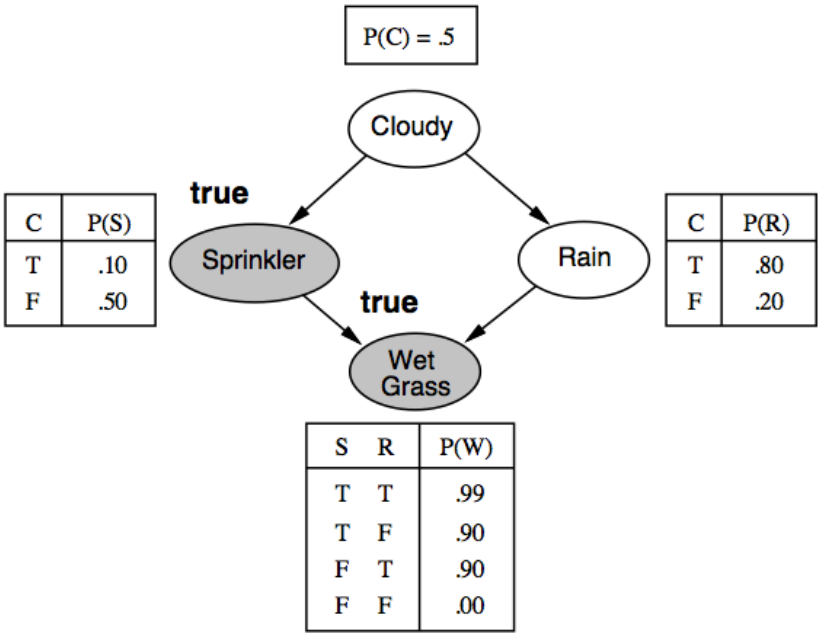
Likelihood Weighting Example

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



Likelihood Weighting Example

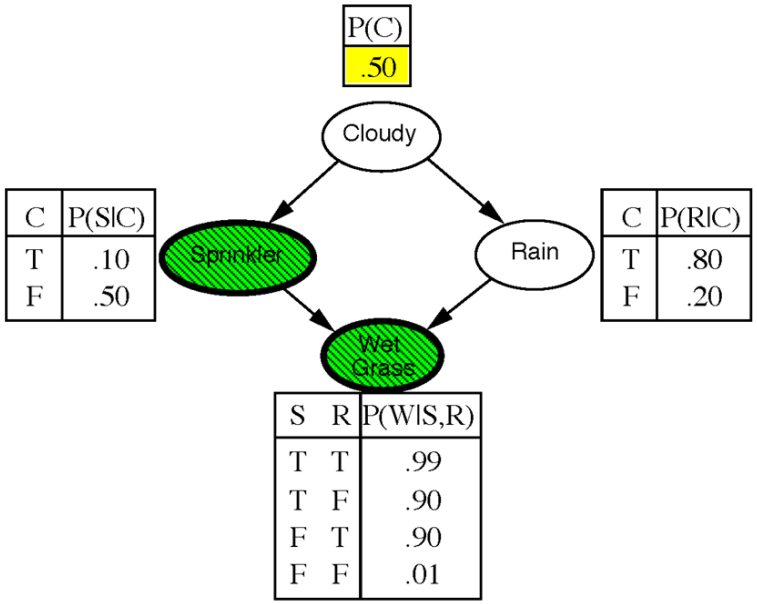
Estimate $\mathbf{P}(Rain|Sprinkler = true, WetGrass = true)$



Sample generation process:

1. $w \leftarrow 1.0$
2. Sample $\mathbf{P}(Cloudy) = \langle 0.5, 0.5 \rangle$; say $true$
3. $Sprinkler$ has value $true$, so
   $w \leftarrow w \times P(Sprinkler = true|Cloudy = true) = 0.1$
4. Sample $\mathbf{P}(Rain|Cloudy = true) = \langle 0.8, 0.2 \rangle$; say $true$
5. $WetGrass$ has value $true$, so
   $w \leftarrow w \times P(WetGrass = true|Sprinkler = true, Rain = true) = 0.099$

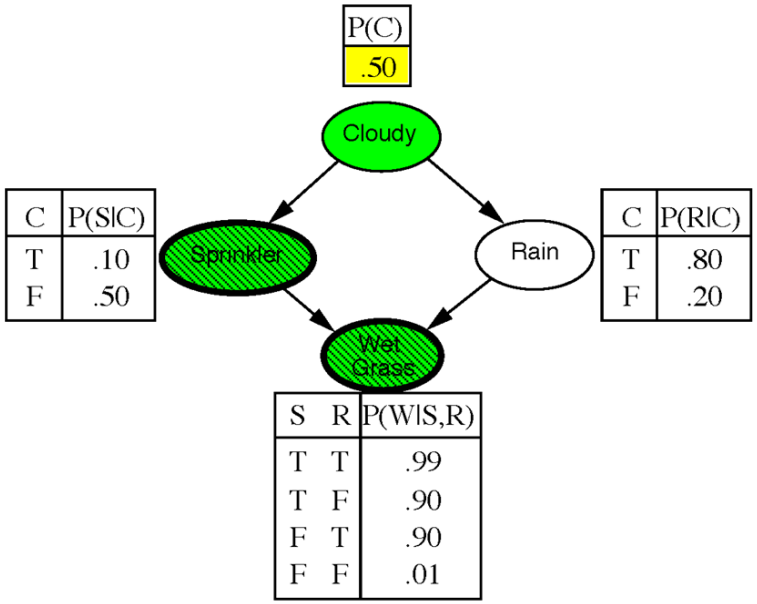
Likelihood weighting example



$w = 1.0$

Likelihood weighting example

Sample Cloudy: true

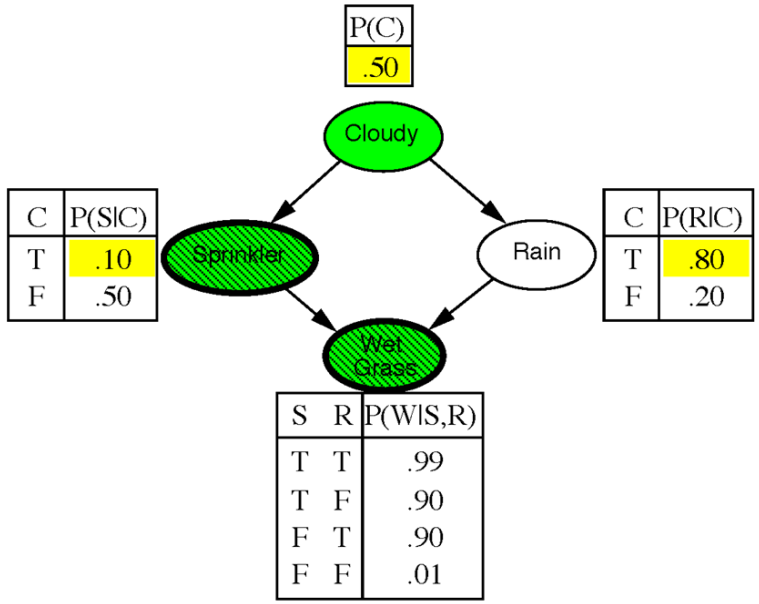


$w = 1.0$

Chapter 14.4–5 27

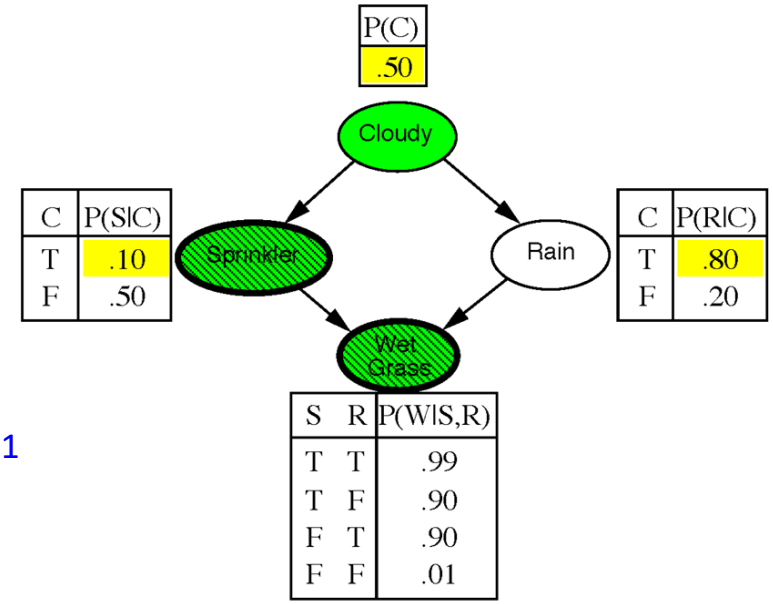
Likelihood weighting example

Cloudy=true now given for the remainder of this sample



$w = 1.0$

Likelihood weighting example



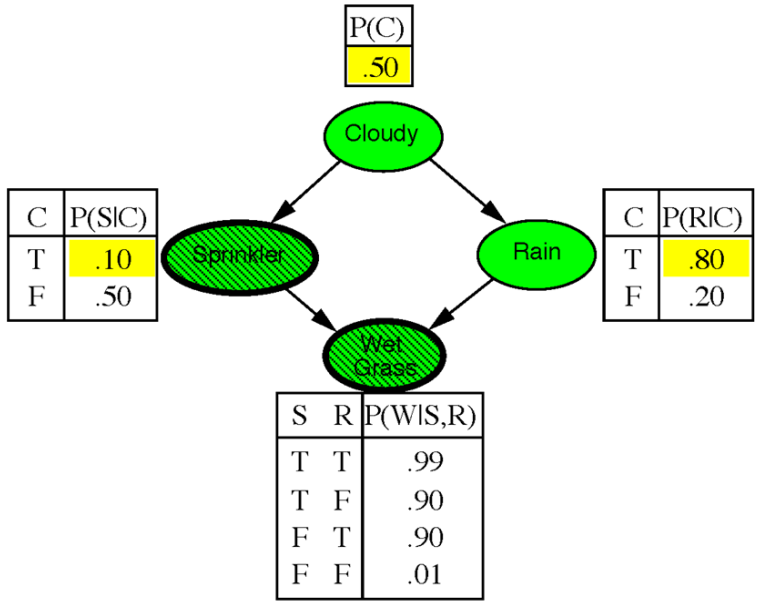
Sprinkler=true was given.

So, update w with
P(Sprinkler=true|parents) =
P(Sprinkler=true|Cloudy=true) = 0.1

$w = 1.0 \times 0.1$

Likelihood weighting example

Sample Rain: say, true

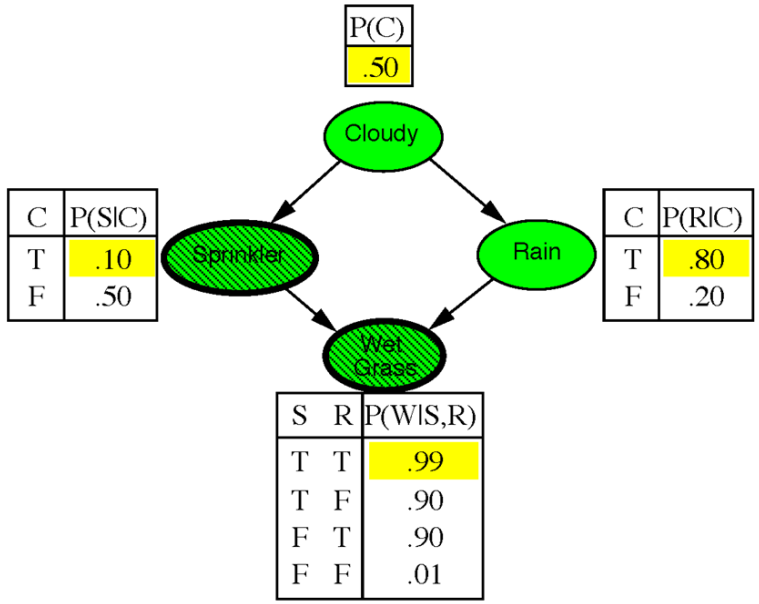


$w = 1.0 \times 0.1$

Chapter 14.4–5 30

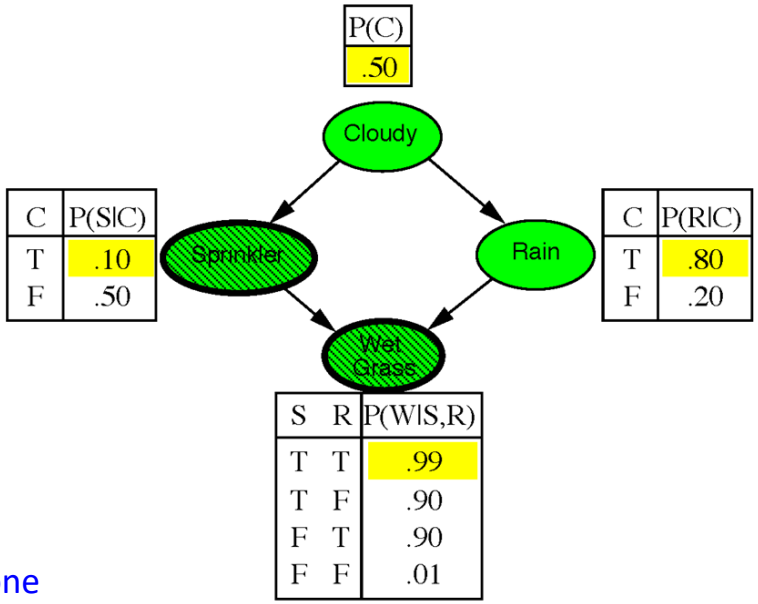
Likelihood weighting example

Rain=true now given for the remainder of this sample



$w = 1.0 \times 0.1$

Likelihood weighting example



Wet Grass=true was given.

So, update w with
P(Wet Grass=true|parents) =
P(Wet Grass=true|
Sprinkler=true,Rain=true) = 0.99

All variable assigned or sampled: done for this sample.

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

Return event (true, true, true, true) with weight 0.099

Likelihood Weighting Analysis

Sampling probability for WEIGHTEDSAMPLE is

$$S_{WS}(\mathbf{y}, \mathbf{e}) = \prod_{i=1}^{l} P(y_i | Parents(Y_i))$$

Note: pays attention to evidence in $ancestors$ only

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

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

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

Weighted sampling probability is

$$\begin{aligned} & S_{WS}(\mathbf{y}, \mathbf{e}) w(\mathbf{y}, \mathbf{e}) \\ & = \prod_{i=1}^{l} P(y_i | Parents(Y_i)) \prod_{i=1}^{m} P(e_i | Parents(E_i)) \\ & = P(\mathbf{y}, \mathbf{e}) \text{ (by standard global semantics of network)} \end{aligned}$$

Hence likelihood weighting returns consistent estimates
but performance still degrades with many evidence variables

Approximate Inference by Markov Chain Monte Carlo (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 approximation to $P(X|\mathbf{e})$
  local variables: $\mathbf{N}[X]$, a vector of counts over X, initially zero
    $\mathbf{Y}$, the nonevidence variables in bn
    $\mathbf{x}$, the current state of the network, initially copied from **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 Y_i in $\mathbf{Y}$ **do**
      sample the value of Y_i in $\mathbf{x}$ from $\mathbf{P}(Y_i|MB(Y_i))$ given the values of $MB(Y_i)$ in $\mathbf{x}$
  return NORMALIZE($\mathbf{N}[X]$)

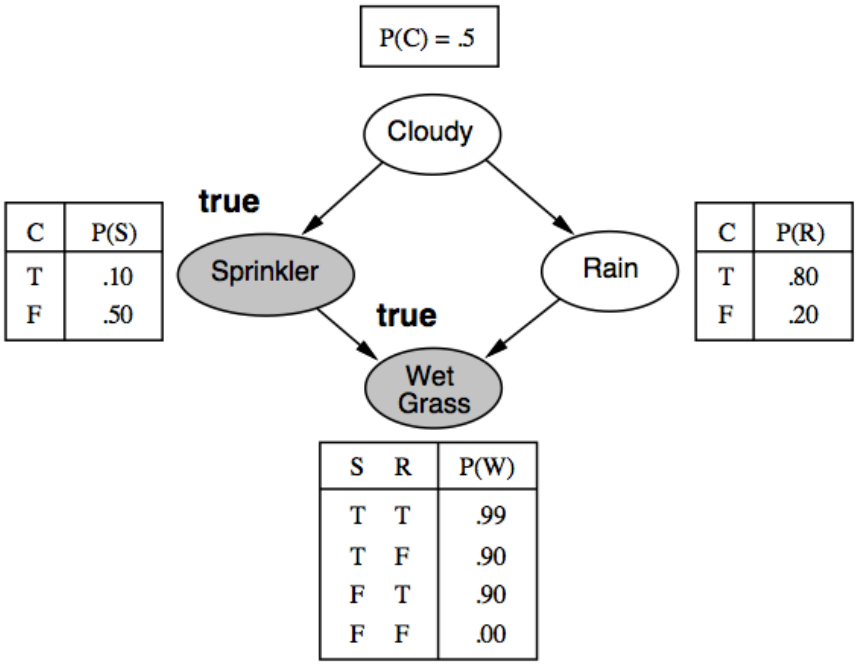
Approaches stationary distribution: long-run fraction of time spent in each state is exactly proportional to its posterior probability

MCMC Example

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

Sample $Cloudy$ then $Rain$, repeat.
Count number of times $Rain$ is true and false in the samples.

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

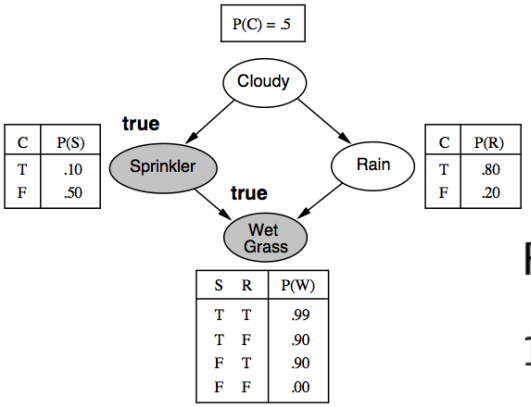


MCMC Example

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

Sample $Cloudy$ then $Rain$, repeat.
Count number of times $Rain$ is true and false in the samples.

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



Random initial state: $Cloudy = true$ and $Rain = false$

1. $\mathbf{P}(Cloudy|MB(Cloudy)) = \mathbf{P}(Cloudy|Sprinkler, \neg Rain)$
   sample $\rightarrow false$

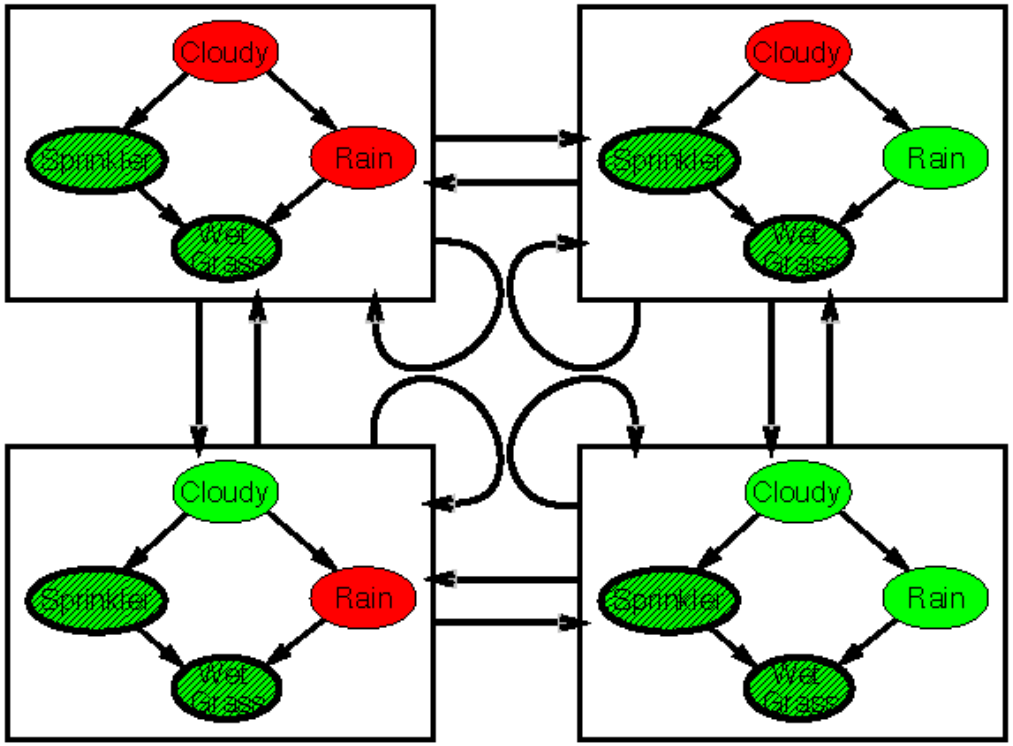
2. $\mathbf{P}(Rain|MB(Rain)) = \mathbf{P}(Rain|\neg Cloudy, Sprinkler, WetGrass)$
   sample $\rightarrow true$

Visit 100 states
31 have $Rain = true$, 69 have $Rain = false$

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

The Markov chain

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



Wander about for a while, average what you see

MCMC Analysis

Transition probability $q(\mathbf{y} \rightarrow \mathbf{y}')$

Occupancy probability $\pi_t(\mathbf{y})$ at time t

Equilibrium condition on π_t defines stationary distribution $\pi(\mathbf{y})$
Note: stationary distribution depends on choice of $q(\mathbf{y} \rightarrow \mathbf{y}')$

Pairwise detailed balance on states guarantees equilibrium

Gibbs sampling transition probability:
sample each variable given current values of all others
$\Rightarrow$ detailed balance with the true posterior

For Bayesian networks, Gibbs sampling reduces to
sampling conditioned on each variable's Markov blanket

Stationary Distribution

$\pi_t(\mathbf{y})$ = probability in state $\mathbf{y}$ at time t
$\pi_{t+1}(\mathbf{y}')$ = probability in state $\mathbf{y}'$ at time $t+1$

π_{t+1} in terms of π_t and $q(\mathbf{y} \to \mathbf{y}')$

$$\pi_{t+1}(\mathbf{y}') = \Sigma_{\mathbf{y}} \pi_t(\mathbf{y}) q(\mathbf{y} \to \mathbf{y}')$$

Stationary distribution: $\pi_t = \pi_{t+1} = \pi$

$$\pi(\mathbf{y}') = \Sigma_{\mathbf{y}} \pi(\mathbf{y}) q(\mathbf{y} \to \mathbf{y}') \qquad \text{for all } \mathbf{y}'$$

If π exists, it is unique (specific to $q(\mathbf{y} \to \mathbf{y}')$)

In equilibrium, expected "outflow" = expected "inflow"

Detailed Balance

"Outflow" = "inflow" for each pair of states:

$$\pi(\mathbf{y})q(\mathbf{y} \to \mathbf{y}') = \pi(\mathbf{y}')q(\mathbf{y}' \to \mathbf{y}) \qquad \text{for all } \mathbf{y},\ \mathbf{y}'$$

Detailed balance $\Rightarrow$ stationarity:

$$\begin{aligned} \Sigma_{\mathbf{y}}\pi(\mathbf{y})q(\mathbf{y} \to \mathbf{y}') &= \Sigma_{\mathbf{y}}\pi(\mathbf{y}')q(\mathbf{y}' \to \mathbf{y}) \\ &= \pi(\mathbf{y}')\Sigma_{\mathbf{y}}q(\mathbf{y}' \to \mathbf{y}) \\ &= \pi(\mathbf{y}') \end{aligned}$$

MCMC algorithms typically constructed by designing a transition probability q that is in detailed balance with desired π

Gibbs Sampling

Sample each variable in turn, given *all other variables*

Sampling Y_i, let $\bar{\mathbf{Y}}_i$ be all other nonevidence variables
Current values are y_i and $\bar{\mathbf{y}}_i$; $\mathbf{e}$ is fixed
Transition probability is given by

$$q(\mathbf{y} \rightarrow \mathbf{y}') = q(y_i, \bar{\mathbf{y}}_i \rightarrow y_i', \bar{\mathbf{y}}_i) = P(y_i'|\bar{\mathbf{y}}_i, \mathbf{e})$$

This gives detailed balance with true posterior $P(\mathbf{y}|\mathbf{e})$:

$$\begin{aligned}
\pi(\mathbf{y})q(\mathbf{y} \rightarrow \mathbf{y}') &= P(\mathbf{y}|\mathbf{e})P(y_i'|\bar{\mathbf{y}}_i, \mathbf{e}) = P(y_i, \bar{\mathbf{y}}_i|\mathbf{e})P(y_i'|\bar{\mathbf{y}}_i, \mathbf{e}) \\
&= P(y_i|\bar{\mathbf{y}}_i, \mathbf{e})P(\bar{\mathbf{y}}_i|\mathbf{e})P(y_i'|\bar{\mathbf{y}}_i, \mathbf{e}) \quad \text{(chain rule)} \\
&= P(y_i|\bar{\mathbf{y}}_i, \mathbf{e})P(y_i', \bar{\mathbf{y}}_i|\mathbf{e}) \qquad \text{(chain rule backwards)} \\
&= q(\mathbf{y}' \rightarrow \mathbf{y})\pi(\mathbf{y}') = \pi(\mathbf{y}')q(\mathbf{y}' \rightarrow \mathbf{y})
\end{aligned}$$

Markov Blanket Sampling

A variable is independent of all others given its Markov blanket:

$$P(y_i'|\bar{\mathbf{y}}_i, \mathbf{e}) = P(y_i'|MB(Y_i))$$

Probability given the Markov blanket is calculated as follows:

$$P(y_i'|MB(Y_i)) = P(y_i'|Parents(Y_i))\prod_{Z_j \in Children(Y_i)} P(z_j|Parents(Z_j))$$

Hence computing the sampling distribution over Y_i for each flip requires just cd multiplications if Y_i has c children and d values; can cache it if c not too large.

Main computational problems:

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

Performance of Approximation Algorithms

Absolute approximation: $|P(X|\mathbf{e}) - \hat{P}(X|\mathbf{e})| \leq \epsilon$

Relative approximation: $\frac{|P(X|\mathbf{e}) - \hat{P}(X|\mathbf{e})|}{P(X|\mathbf{e})} \leq \epsilon$

Relative $\Rightarrow$ absolute since $0 \leq P \leq 1$ (may be $O(2^{-n})$)

Randomized algorithms may fail with probability at most δ

Polytime approximation: poly$(n, \epsilon^{-1}, \log \delta^{-1})$

Theorem (Dagum and Luby, 1993): both absolute and relative approximation for either deterministic or randomized algorithms are NP-hard for any $\epsilon, \delta < 0.5$

(Absolute approximation polytime with no evidence—Chernoff bounds)

Summary

- Bayesian Networks
  - Use Conditional Independence to efficiently represent full probability distribution
  - Can be constructed by ordering variables and check dependencies
  - Inference methods, including
    - Causal or evidential reasoning
    - Exact inferences (NP-hard)
    - Approximation Inferences