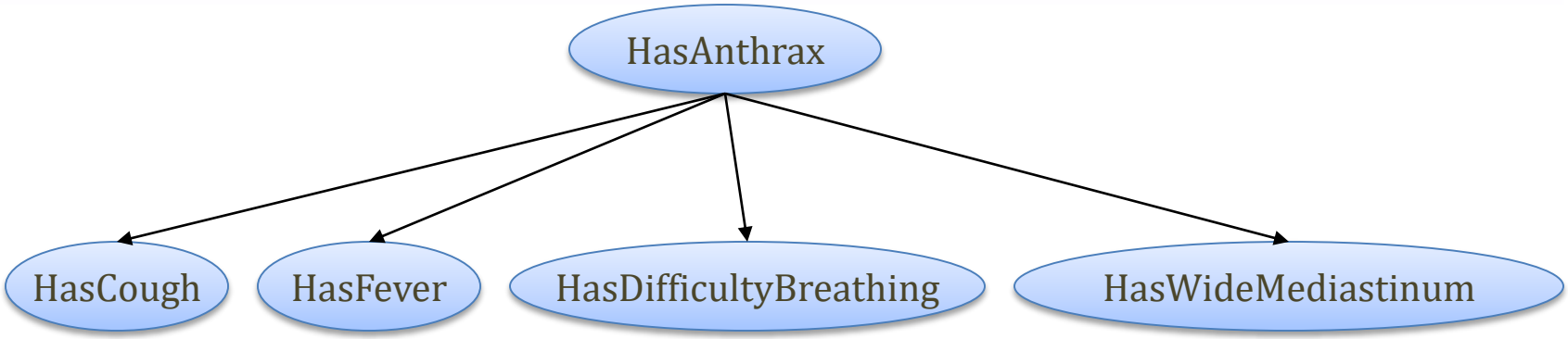
SHIRAZ UNIVERSITY
Computer Science and Engineering Department
Machine Learning Lab

Machine Learning

Bayesian Networks

Bayesian Networks



- *In the opinion of many AI researchers, Bayesian networks are the most significant contribution in AI in the last 10 years*
- *They are used in many applications e.g. spam filtering, speech recognition, robotics, diagnostic systems and …*

The Joint Probability Distribution

- *Joint probabilities can be between any number of variables*
  e.g. $P(A = true, B = true, C = true)$
- *For each combination of variables, we need to say how probable that combination is*
- *The probabilities of these combinations need to sum to 1*

A	B	C	$P(A, B, C)$
false	false	false	0.1
false	false	true	0.2
false	true	false	0.05
false	true	true	0.05
true	false	false	0.3
true	false	true	0.1
true	true	false	0.05
true	true	true	0.15

Sums to 1

The Joint Probability Distribution

- *if you have the joint probability distribution, you can calculate any probability involving A, B, and C*
- *Note: May need to use marginalization and Bayes rule*

A	B	C	$P(A,B,C)$
false	false	false	0.1
false	false	true	0.2
false	true	false	0.05
false	true	true	0.05
true	false	false	0.3
true	false	true	0.1
true	true	false	0.05
true	true	true	0.15

Examples of things you can compute:

- $P(A = true) = sum\ of\ P(A,B,C)\ in\ rows\ with\ A = true$
- $P(A = true, B = true \mid C = true)$

$$= \frac{P(A = true, B = true, C = true)}{P(C = true)}$$

The Problem with the Joint Distribution

- *Lots of entries in the table to fill up!*
- *For* k *Boolean random variables, you need a table of size* 2^k
- *How do we use fewer numbers? Need the concept of independence*

A	B	C	$P(A, B, C)$
false	false	false	0.1
false	false	true	0.2
false	true	false	0.05
false	true	true	0.05
true	false	false	0.3
true	false	true	0.1
true	true	false	0.05
true	true	true	0.15

Independence

Variables A and B are independent if any of the following hold:

- $P(A, B) = P(A)\,P(B)$
- $P(A \mid B) = P(A)$
- $P(B \mid A) = P(B)$

This says that knowing the outcome of A does not tell me anything new about the outcome of B.

Conditional Independence

Variables A and B are conditionally independent given C if any of the following hold:

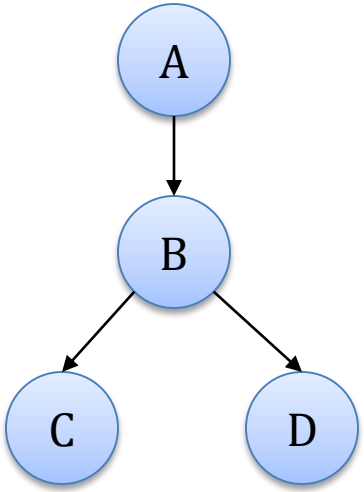
- $P(A, B \mid C) = P(A \mid C)\, P(B \mid C)$
- $P(A \mid B, C) = P(A \mid C)$
- $P(B \mid A, C) = P(B \mid C)$

Knowing C tells me everything about B. I don't gain anything by knowing A (either because A doesn't influence B or because knowing C provides all the information knowing A would give)

A Bayesian Network

A Bayesian network is made up of:

1. A Directed Acyclic Graph



2. A set of tables for each node in the graph

A	***P*(*A*)**
false	0.6
true	0.4

A	***B***	***P*(*B*|*A*)**
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	***D***	***P*(*D*|*B*)**
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

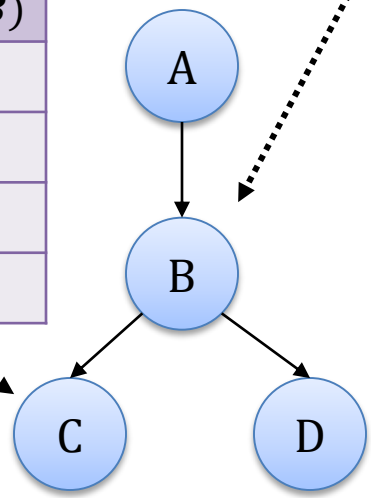
B	***C***	***P*(*C*|*B*)**
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1

A Set of Tables for Each Node

A	$P(A)$
false	0.6
true	0.4

A	B	$P(B\|A)$
false	false	0.01
false	true	0.99
true	false	0.7
true	true	0.3

B	C	$P(C\|B)$
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1



B	D	$P(D\|B)$
false	false	0.02
false	true	0.98
true	false	0.05
true	true	0.95

Each node X_i has a conditional probability distribution

$P(Xi \mid Parents(Xi))$ that quantifies the effect of the parents on the node

The parameters are the probabilities in these conditional probability tables (CPTs)

A Set of Tables for Each Node

Conditional Probability Distribution for C given B

B	C	$P(C\|B)$
false	false	0.4
false	true	0.6
true	false	0.9
true	true	0.1

For a given combination of values of the parents (B in this example), the entries for P(C=true | B) and P(C=false | B) must add up to 1

e.g. P(C=true | B=false) + P(C=false |B=false)=1

If you have a Boolean variable with k Boolean parents, this table has 2^{k+1} *probabilities (but only* 2^k *need to be stored)*

The Joint Probability Distribution

Due to the Markov condition, we can compute the joint probability distribution over all the variables $X_1, \dots, X_n$ in the Bayesian net using the formula:

$$P(X_1 = x_1, \dots, X_n = x_n) = \prod_{i=1}^{n} P(X_i = x_i | Parents(X_i))$$

Where $Parents(X_i)$ means the values of the Parents of the node X_i with respect to the graph

Chain Rule

A joint probability distribution can be expressed as a product of conditional probabilities:

$$P(X_1, X_2, \dots, X_n)$$

$$= P(X1)\ P(X_2, X_3, \dots, X_n \mid X_1)$$

$$= P(X_1)\, P(X_2|X_1)\ P(X_3, X_4, \dots, X_n|X_1, X_2)$$

$$= P(X_1)P(X_2|X_1)P(X_3|X_1, X_2)\, P(X_4, \dots, X_n|X_1, X_2, X_3) \dots \dots$$

$$= P(X_1)\, P(X_2|X_1)\, P(X_3|X_1, X_2) \dots P(X_n|X_1, \dots X_n - 1)$$

This has nothing to do with any independence assumption!

Using a Bayesian Network Example

Using the network in the example, suppose you want to calculate:

$$P(A = true, B = true, C = true, D = true)$$

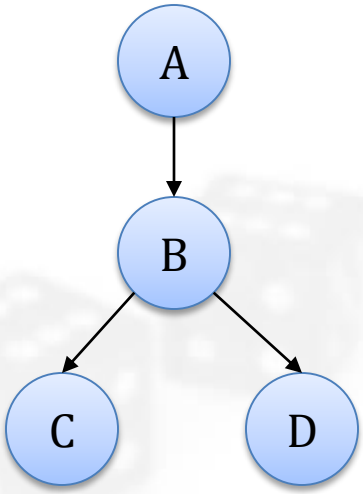
$$= P(A = true) * P(B = true \mid A = true) *$$

$$P(C = true \mid B = true)\, P(D = true \mid B = true)$$

This is from the graph structure

$$= (0.4) * (0.3) * (0.1) * (0.95)$$

These numbers are from the conditional probability tables



Inference

- *Using a Bayesian network to compute probabilities is called inference*

- *In general, inference involves queries of the form:*

$P(X \mid E)$

E = The evidence variable(s)

X = The query variable(s)

One last unresolved issue…

We still haven't said where we get the Bayesian network from. There are two options:

- *Get an expert to design it*
- *Learn it from data*

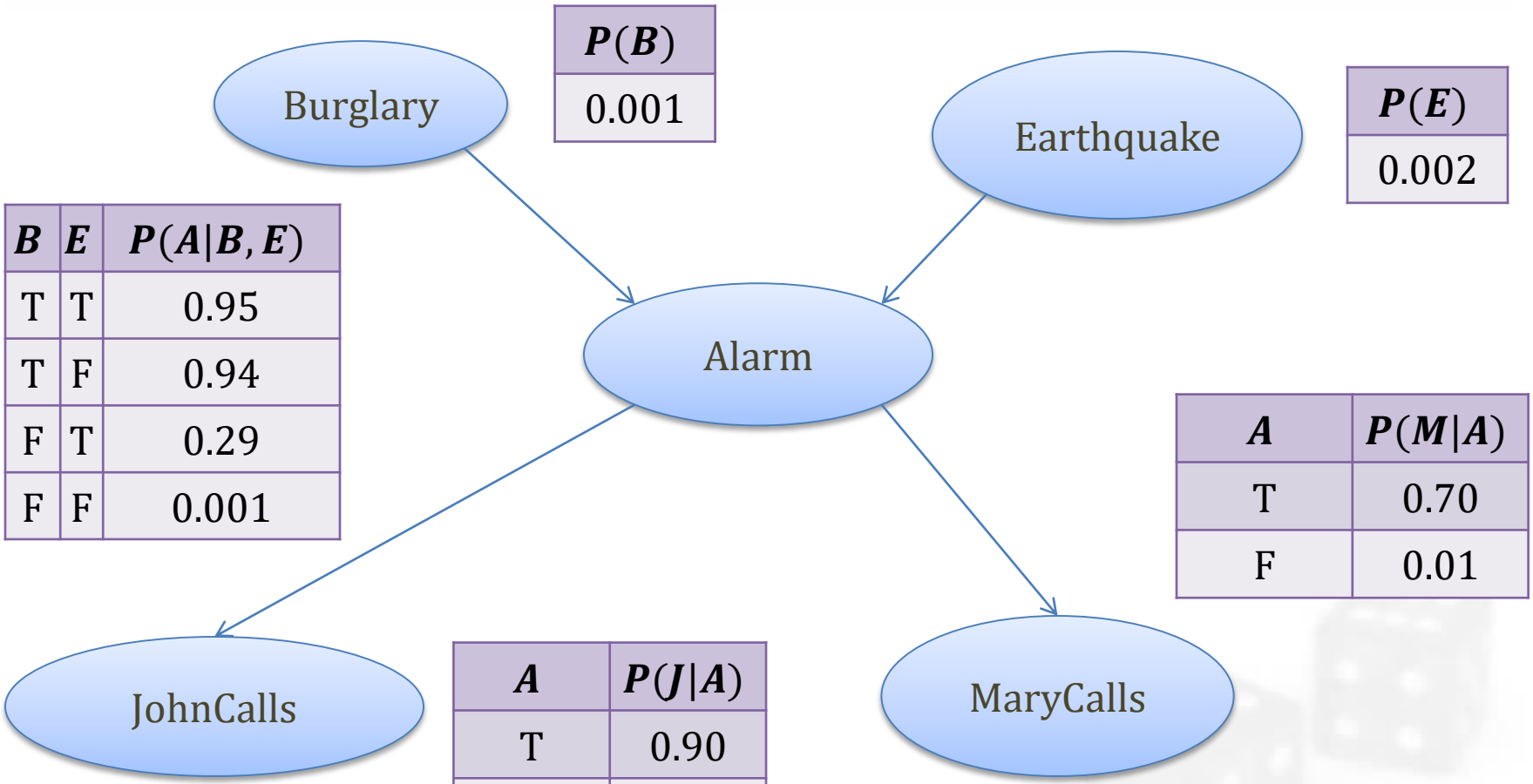
Example

- *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: Burglary, 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.



$P(B)$
0.001

$P(E)$
0.002

B	E	$P(A\|B,E)$
T	T	0.95
T	F	0.94
F	T	0.29
F	F	0.001

A	$P(M\|A)$
T	0.70
F	0.01

A	$P(J\|A)$
T	0.90
F	0.05

Constructing Bayesian networks

1. *Choose an ordering of variables* $X_1, \dots, X_n$
2. *For i = 1 to n*
   - *add* X_i *to the network*
   - *select parents from* $X_1, \dots, Xi_{\_1}$ *such that*

$$P\,(Xi \mid Parents(Xi)) \;=\; P\,(Xi \mid X_1, \dots Xi_{\_1})$$

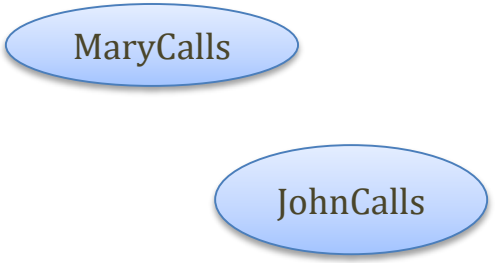
This choice of parents guarantees:

$$P\,(X_1, \dots, X_n) \;=\; \prod_{i=1}^{n} P\,(X_i \mid X_1, \dots, X_{i-1}) \qquad \text{(chain rule)}$$

$$=\; \prod_{i=1}^{n} P(X_i \mid Parents(X_i)) \qquad \text{(by construction)}$$

Example

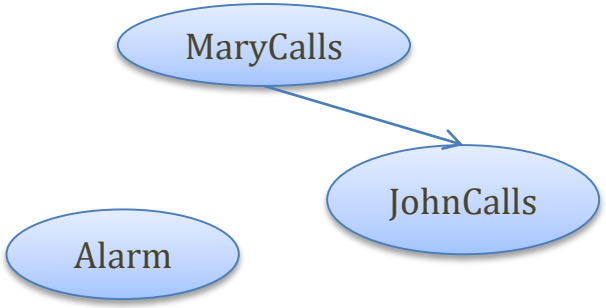
Suppose we choose the ordering M, J, A, B, E



$P(J \mid M) = P(J)$?

Example

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



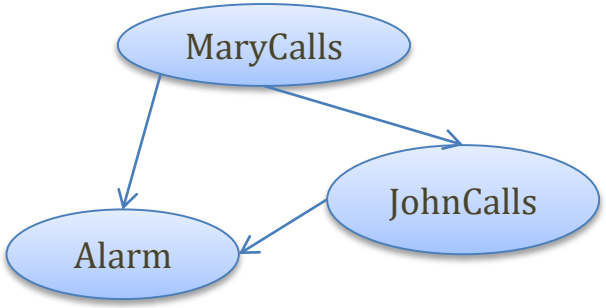
P(J | M) = P(J) ?

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

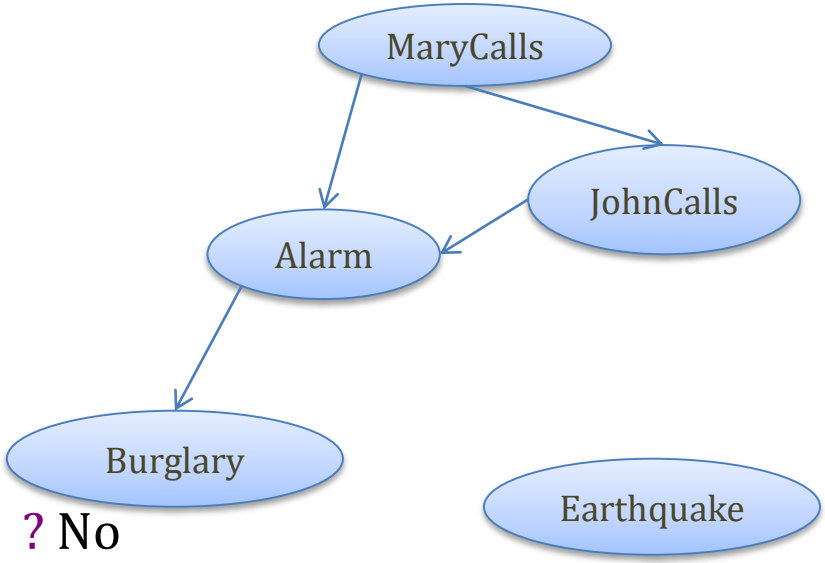
P(A | J, M) = P(A | J)? *P(A | J, M) = P(A)* ? No

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

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

Example

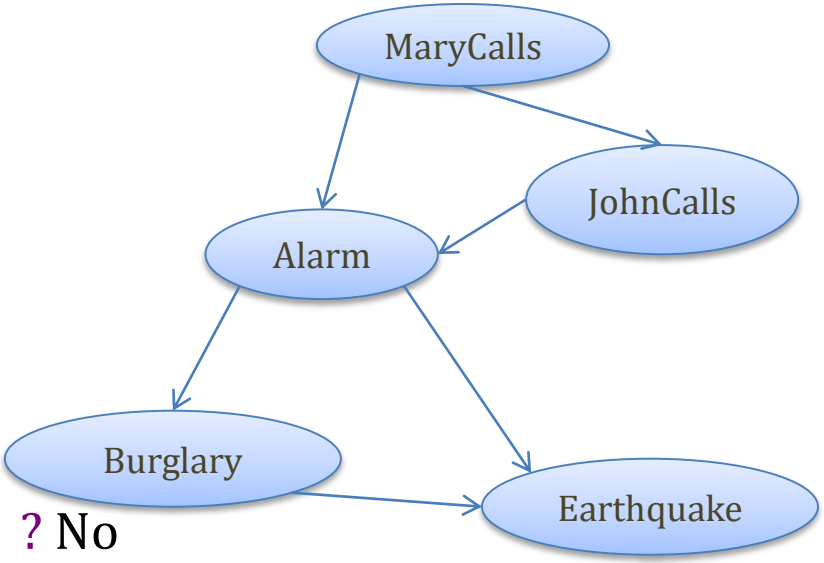
Suppose we choose the ordering M, J, A, B, E



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

Learning Bayesian Network

If structure is known and observe all variables

- *Then it's easy as training a Naïve Bayes classifier*

Suppose structure is known, variables are partially observable

- *Similar to training neural network with hidden units*
- *In fact, can learn network conditional probability tables using gradient ascent!*
- *Converge to network h that (locally) maximizes* $P(D|h)$

Gradient Ascent for Bayes Nets

Let w_{ijk} *denote one entry in the conditional probability table for variable* Y_i *in the network*

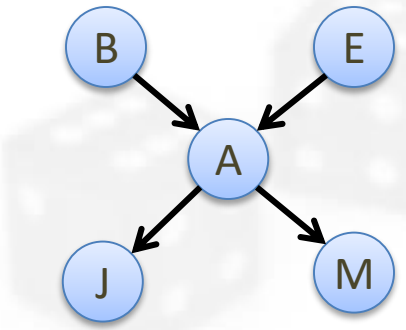
$$w_{ijk} = P\left(Y_i = y_{ij} \middle| Parents\ (Y_i) = the\ list\ u_{ik}\ of\ values\right)$$

Perform gradient ascent by repeatedly

1. *Update all* w_{ijk} *using training data* D

$$w_{ijk} \leftarrow w_{ijk} + \eta \sum_{d \in D} \frac{P_h\left(y_{ij}, u_{ik} \middle| d\right)}{w_{ijk}}$$

2. *Then , renormalize the* w_{ijk} *to assure*
   - $\sum_j w_{ijk} = 1$
   - $0 \leq w_{ijk} \leq 1$



Gradient Ascent for Bayes Nets

$$\frac{\partial \ln P_h(D)}{\partial w_{ijk}} = \frac{\partial}{\partial w_{ijk}} \ln \prod_{d \in D} P_h(d)$$

$$= \sum_{d \in D} \frac{\partial \ln P_h(d)}{\partial w_{ijk}}$$

$$= \sum_{d \in D} \frac{1}{P_h(d)} \frac{\partial P_n(d)}{\partial w_{ijk}}$$

This last step makes use of the general equality $\frac{\partial \ln f(x)}{\partial x} = \frac{1}{f(x)} \frac{\partial f(x)}{\partial x}$ *we can now introduce the values of the variables* Y_i *and* $U_i = Parents\ (Y_i)$ *by summing over*

$$\frac{\partial ln P(D)}{\partial w_{ijk}} = \sum_{d \in D} \frac{1}{P_h(d)} \frac{\partial}{\partial w_{ijk}} \sum_{a,b} P_h(d|y_{ia}, u_{ib}) P_h(y_{ia}, u_{ib})$$

$$= \sum_{d \in D} \frac{1}{P_h(d)} \frac{\partial}{\partial w_{ijk}} \sum_{a,b} P_h(d|y_{ia}, u_{ib}) P_h(y_{ia}|u_{ib}) P_h(u_{ib})$$

Gradient Ascent for Bayes Nets

Now Consider the rightmost sum in the final expression above.

Given that $w_{ijk} \equiv P_h(d|y_{ij}, u_{ik})$ *, the only term in this sum for which* $\frac{\partial}{\partial w_{ijk}}$ *is nonzero is the term for which* $a = j$ *and* $b = k$ *Therefore*

$$\frac{\partial ln\, P_h(d)}{\partial w_{ijk}} = \frac{1}{P_h(d)} \frac{\partial}{\partial w_{ijk}} P_h(d|y_{ij}, u_{ik}) P_h(y_{ij}|u_{ik}) P_h(u_{ik})$$

$$= \sum_{d \in D} \frac{1}{P_h(d)} \frac{\partial}{\partial w_{ijk}} P_h(d|y_{ij}, u_{ik}) w_{ijk} P_h(u_{ik})$$

$$= \sum_{d \in D} \frac{1}{P_h(d)} P_h(d|y_{ij}, u_{ik}) P_h(u_{ik})$$

Gradient Ascent for Bayes Nets

Applying Bayes theorem to rewrite $P_h(d|y_{ij},u_{ik})$*,we have:*

$$\frac{\partial P_h(D)}{\partial w_{ijk}} = \sum_{d\in D} \frac{1}{P_h(d)} \frac{P_h(y_{ij},u_{ik}|d)P_h(d)P_h(u_{ik})}{P_h(y_{ij},u_{ik})}$$

$$= \sum_{d\in D} \frac{P_h(y_{ij},u_{ik}|d)P_h(u_{ik})}{P_h(y_{ij},u_{ik})}$$

$$= \sum_{d\in D} \frac{P_h(y_{ij},u_{ik}|d)}{P_h(y_{ij}|u_{ik})}$$

$$= \sum_{d\in D} \frac{P_h(y_{ij},u_{ik}|d)}{w_{ijk}}$$

Gradient Ascent for Bayes Nets

- *In particular , we require as the weight w_{ijk} is updated, they must remain valid probabilities in the interval [0,1].*
- *We also require that* $\forall i,k \ \sum_j w_{ijk} = 1$

These constrains can be satisfied by updating weights in two-step process:

1. $w_{ijk} \leftarrow w_{ijk} + \eta \sum_{d \in D} \frac{P_h(y_{ij}, u_{ik}|d)}{w_{ijk}}$

2. *Normalize* w_{ijk} *to satisfy the above constraints.*

Inference in Bayesian networks

- *Exact inference by enumeration*
- *Exact inference by variable elimination*
- *Approximate inference by stochastic simulation*
- *Approximate inference by Markov Chain Monte Carlo*

Inference by enumeration

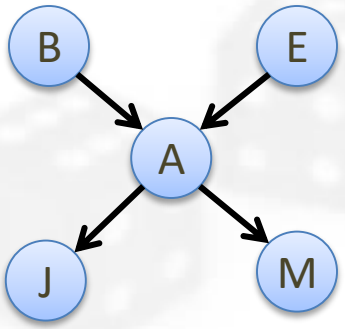
Simply query on the burglary network

$$P(B|j,m) = \frac{P(B,j,m)}{P(j,m)} = \alpha P(B,j,m) = \alpha \sum_e \sum_a P(B,e,a,j,m)$$

Rewrite full joint entries using product of CPT entries:

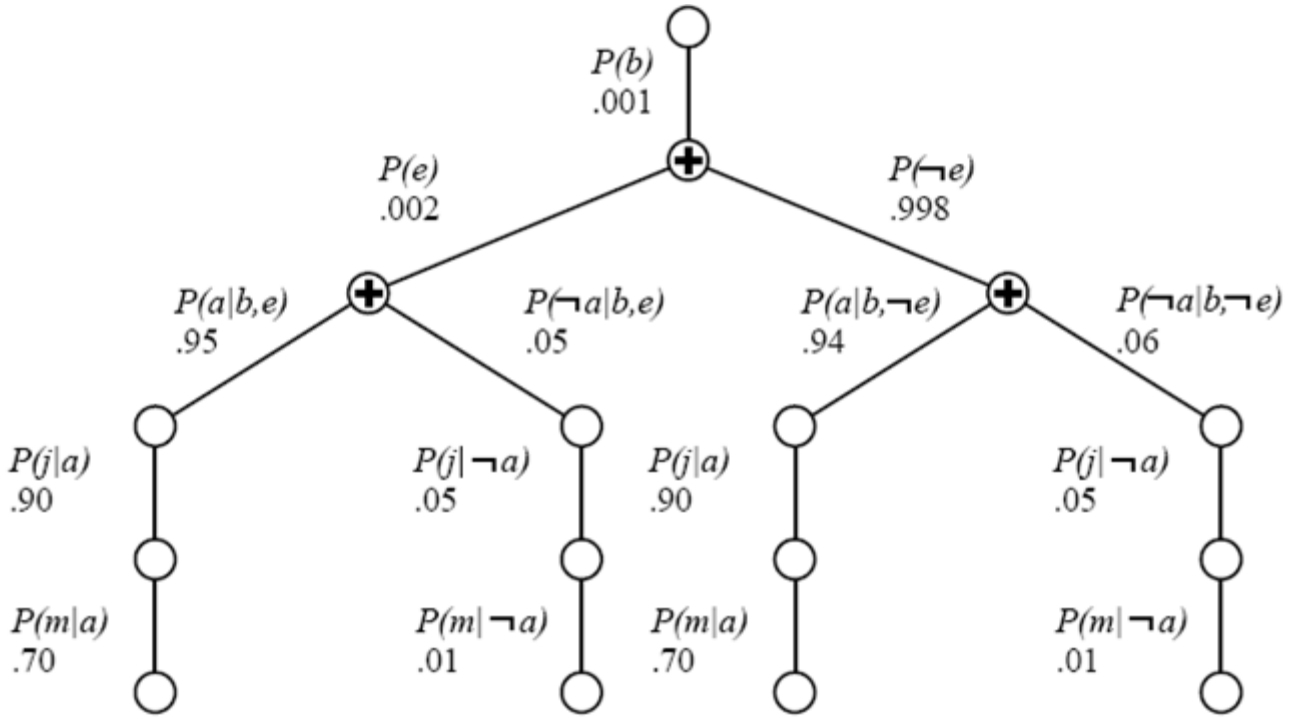
$$P(B|j,m) = \alpha \sum_e \sum_a P(B)P(e)P(a|B,e)P(j|a)P(m|a)$$

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



Evaluation tree

$$\alpha\, P(B) \sum_{e} P(e) \sum_{a} P(a|B,e) P(j|a) P(m|a)$$



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

Inference by variable elimination

$$P(B|j,m) = \alpha \underbrace{P(B)}_{B} \sum_e \underbrace{P(e)}_{E} \sum_a \underbrace{P(a|B,e)}_{A} \underbrace{P(j|a)}_{J} \underbrace{P(m|a)}_{M}$$

$$= \alpha P(B) \sum_e P(e) \sum_a P(a|B,e) P(j|a) f_M(a)$$

$$= \alpha P(B) \sum_e P(e) \sum_a P(a|B,e) f_J(a) f_M(a)$$

$$= \alpha P(B) \sum_e P(e) \sum_a f_A(a,b,e) f_J(a) f_M(a)$$

$$= \alpha P(B) \sum_e P(e) f_{\bar{A}JM}(b,e) \quad \textit{(sum out } A\textit{)}$$

$$= \alpha P(B) f_{\bar{E}\bar{A}JM}(b) \quad \textit{(sum out } E\textit{)}$$

$$= \alpha f_B(b) f_{\bar{E}\bar{A}JM}(b)$$

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

Irrelevant variables

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

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

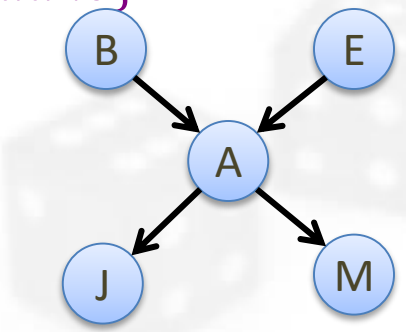
Sum over m *is identically 1;* M *is irrelevant to the query*

Thm 1: Y *is irrelevant unless* $Y \in Ancestors\ (\{X\} \cup E)$

Here $X = JohnCalls\ , E = \{Burglary\}$, *and*

$$Ancestors\ (\{X\} \cup E) = \{\ Alarm, Earthquake\}$$

So $MarryCalls$ *is irrelevant.*



Inference by stochastic simulation

Basic Idea

1) Draw N samples from a sampling distribution S

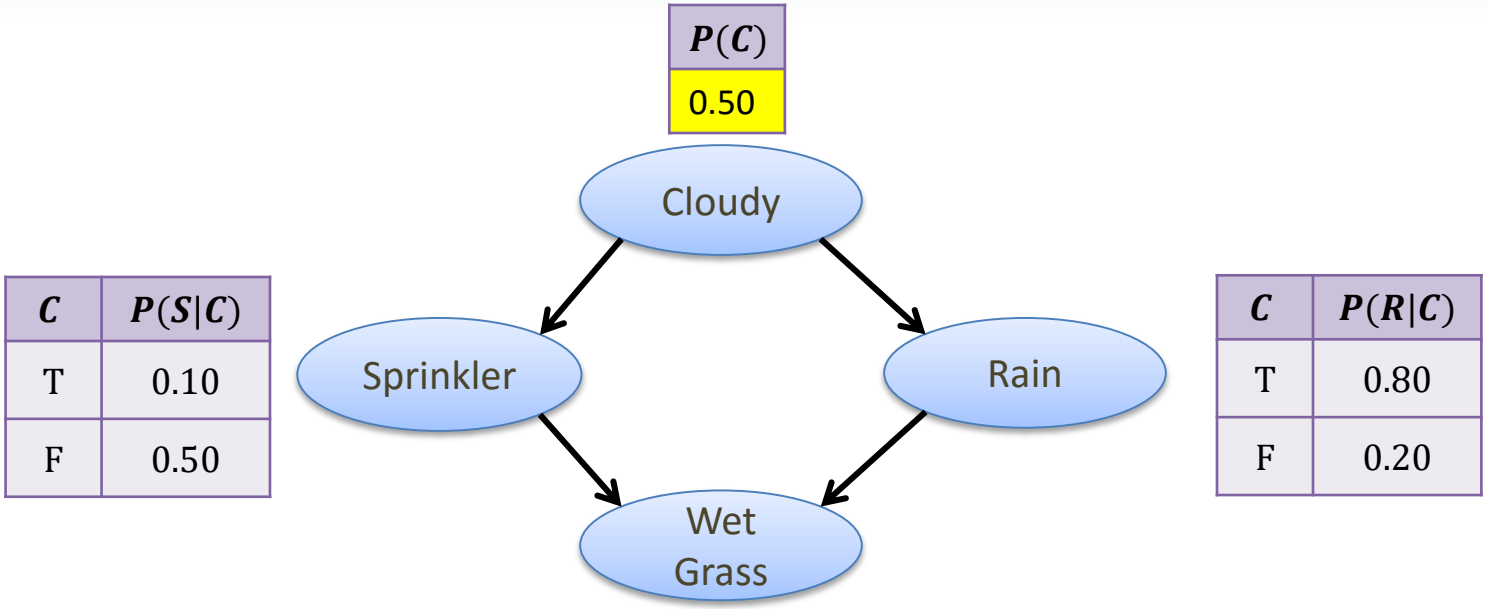
2) Compute an approximate posterior probability $\hat{P}$

3) Show this converges to the true probability P

Outlines

- *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*

Example

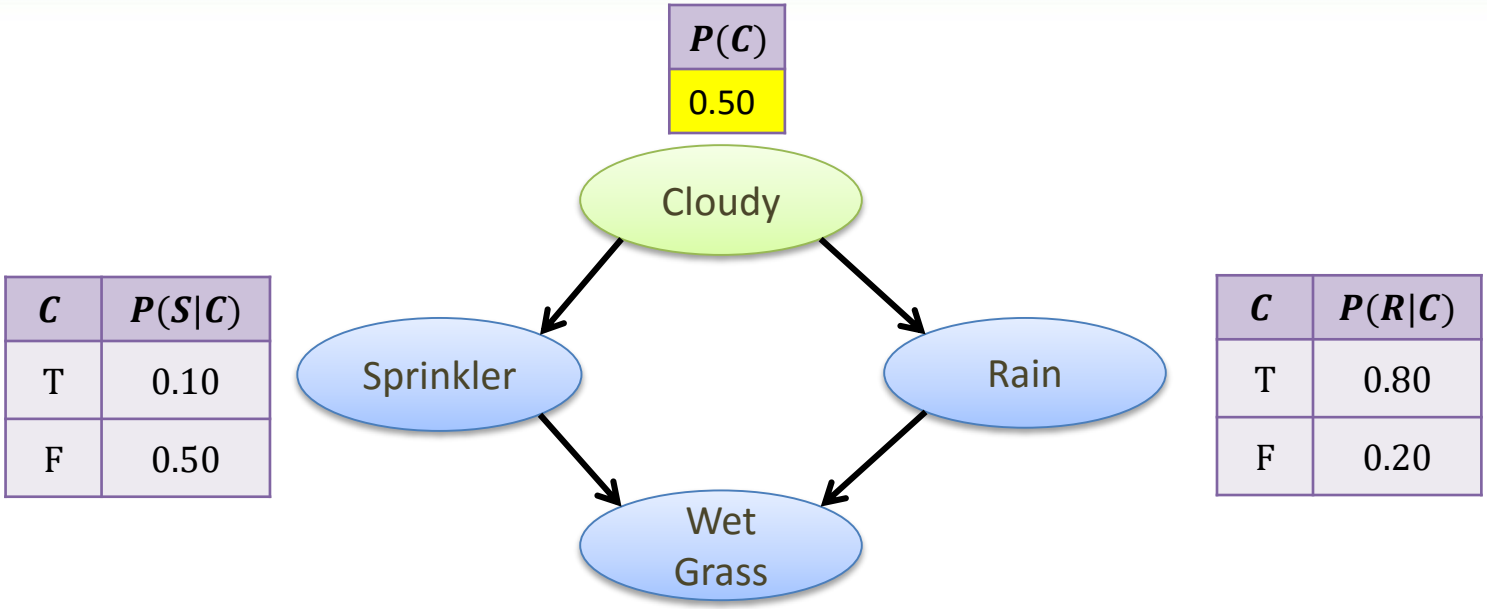


C	$P(S\|C)$
T	0.10
F	0.50

C	$P(R\|C)$
T	0.80
F	0.20

S	R	$P(W\|S,R)$
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.01

Example

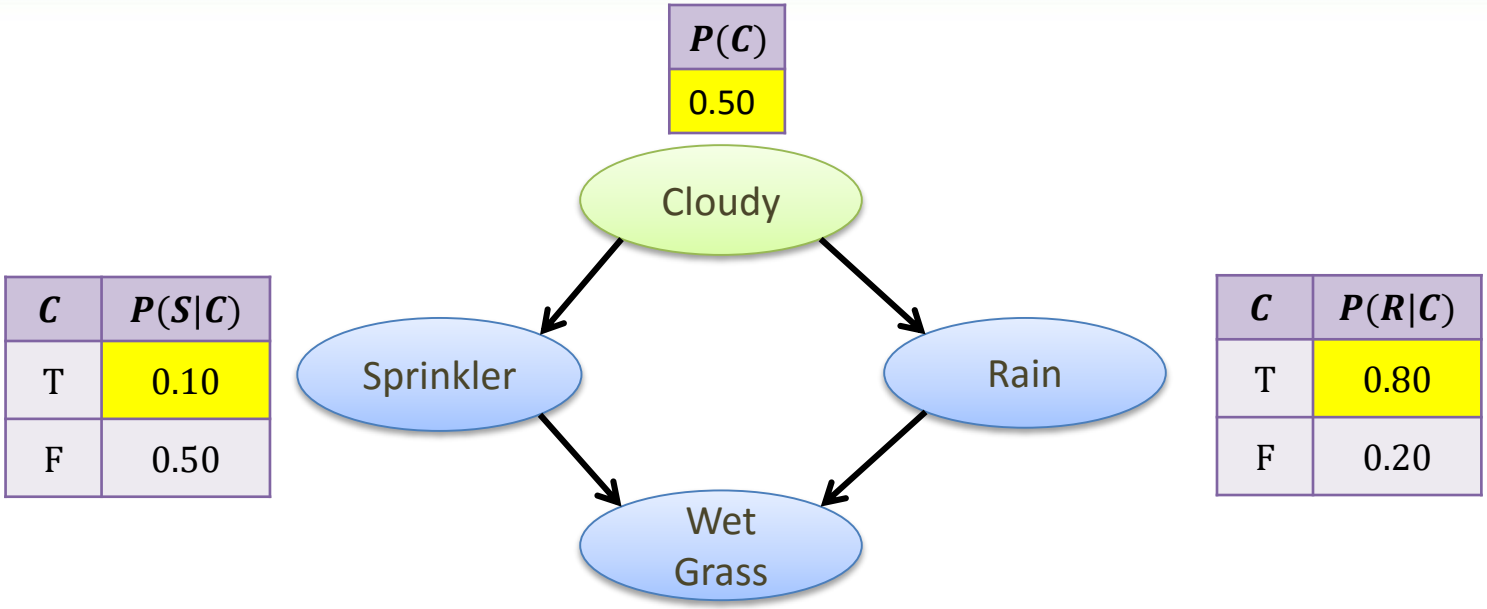


C	$P(S\|C)$
T	0.10
F	0.50

C	$P(R\|C)$
T	0.80
F	0.20

S	R	$P(W\|S,R)$
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.01

Example



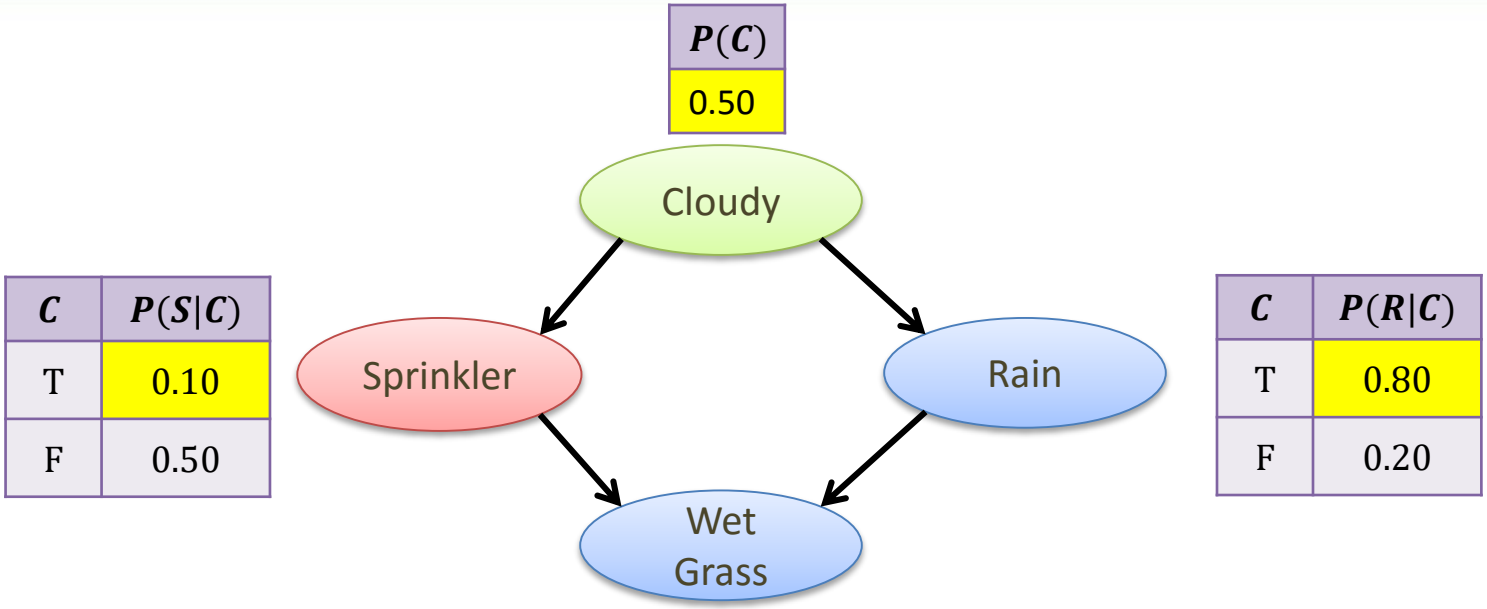
$P(C)$
0.50

C	$P(S\|C)$
T	0.10
F	0.50

C	$P(R\|C)$
T	0.80
F	0.20

S	R	$P(W\|S,R)$
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.01

Example



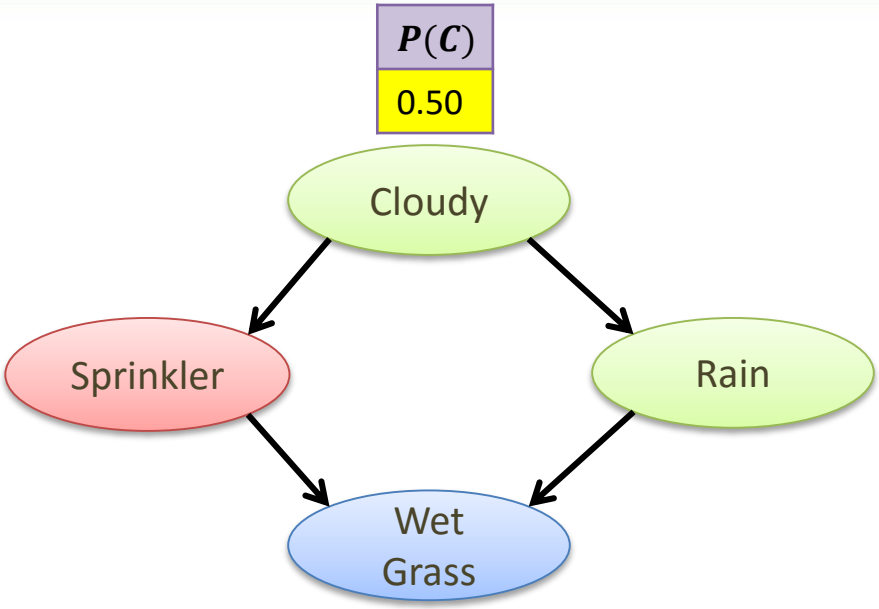
$P(C)$
0.50

C	$P(S\|C)$
T	0.10
F	0.50

C	$P(R\|C)$
T	0.80
F	0.20

S	R	$P(W\|S,R)$
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.01

Example



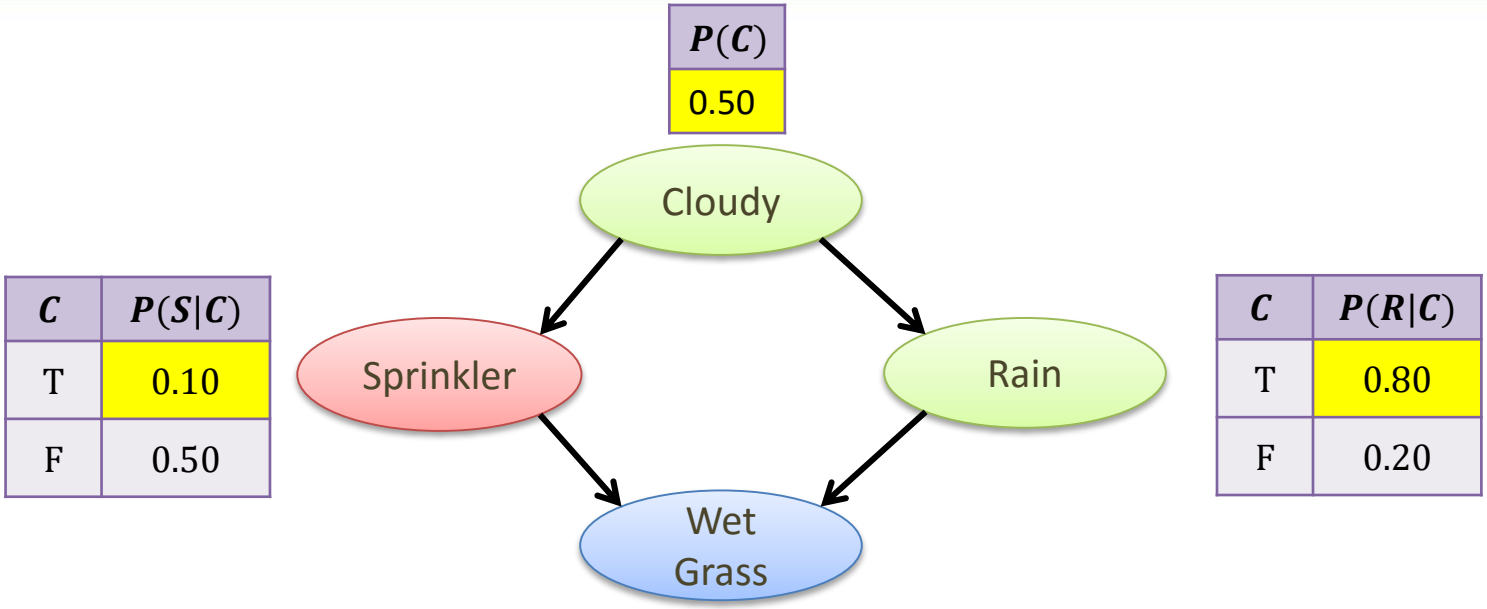
$P(C)$
0.50

C	$P(S\|C)$
T	0.10
F	0.50

C	$P(R\|C)$
T	0.80
F	0.20

S	R	$P(W\|S,R)$
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.01

Example

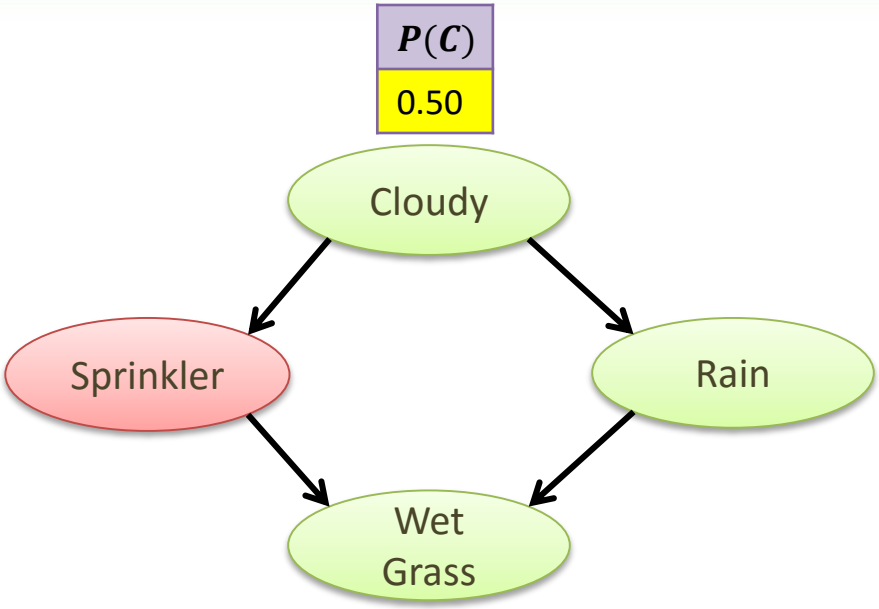


C	$P(S\|C)$
T	0.10
F	0.50

C	$P(R\|C)$
T	0.80
F	0.20

S	R	$P(W\|S,R)$
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.01

Example



$P(C)$
0.50

C	$P(S\|C)$
T	0.10
F	0.50

C	$P(R\|C)$
T	0.80
F	0.20

S	R	$P(W\|S,R)$
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.01

Rejection sampling

$\hat{P}(x|e)$ estimate from samples agreeing with 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 $P(Rain|Sprinkler = true)$ *using 100 samples*

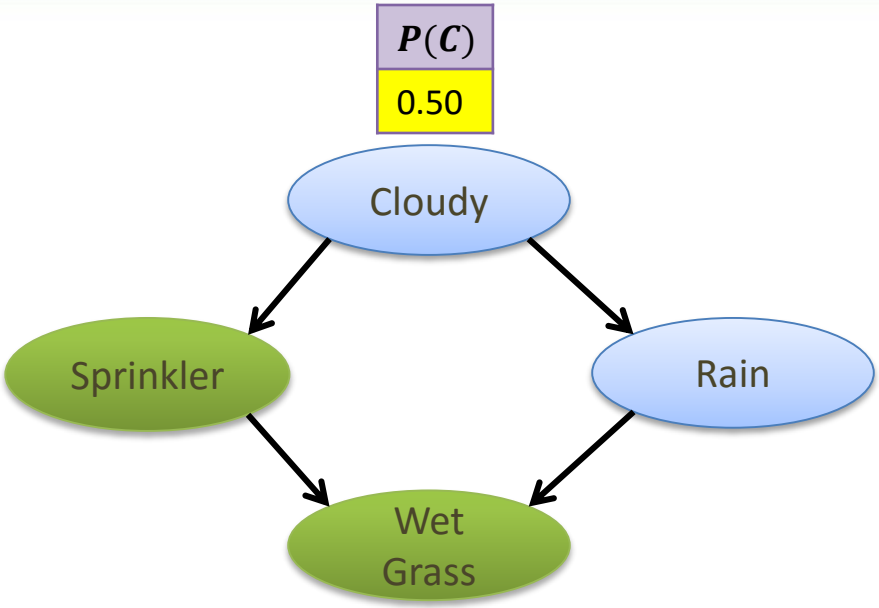
27 samples have $Sprinkler = true$

of these, 8 have $Rain = true$ *and 19 have* $Rain = false$.

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

Similar to a basic real-world empirical estimation procedure

Example



$P(C)$
0.50

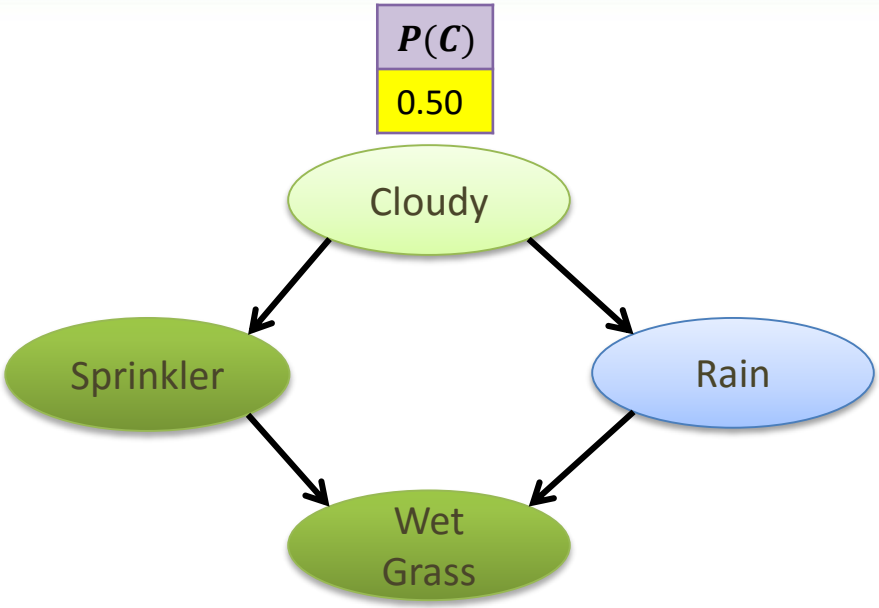
C	$P(S\|C)$
T	0.10
F	0.50

C	$P(R\|C)$
T	0.80
F	0.20

S	R	$P(W\|S,R)$
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.01

$w = 1.0$

Example



$P(C)$
0.50

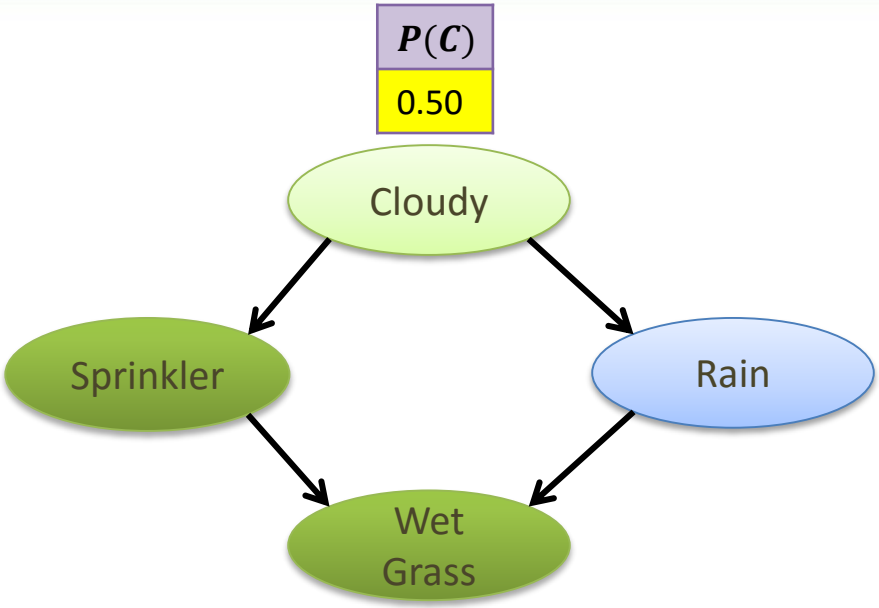
C	$P(S\|C)$
T	0.10
F	0.50

C	$P(R\|C)$
T	0.80
F	0.20

S	R	$P(W\|S,R)$
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.01

$w = 1.0$

Example



$P(C)$
0.50

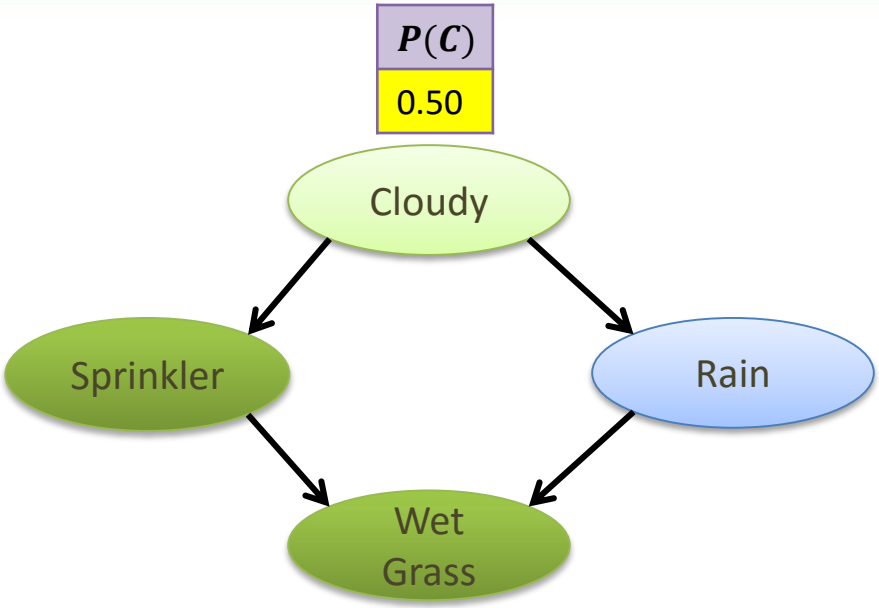
C	$P(S\|C)$
T	0.10
F	0.50

C	$P(R\|C)$
T	0.80
F	0.20

S	R	$P(W\|S,R)$
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.01

$w = 1.0$

Example



$P(C)$
0.50

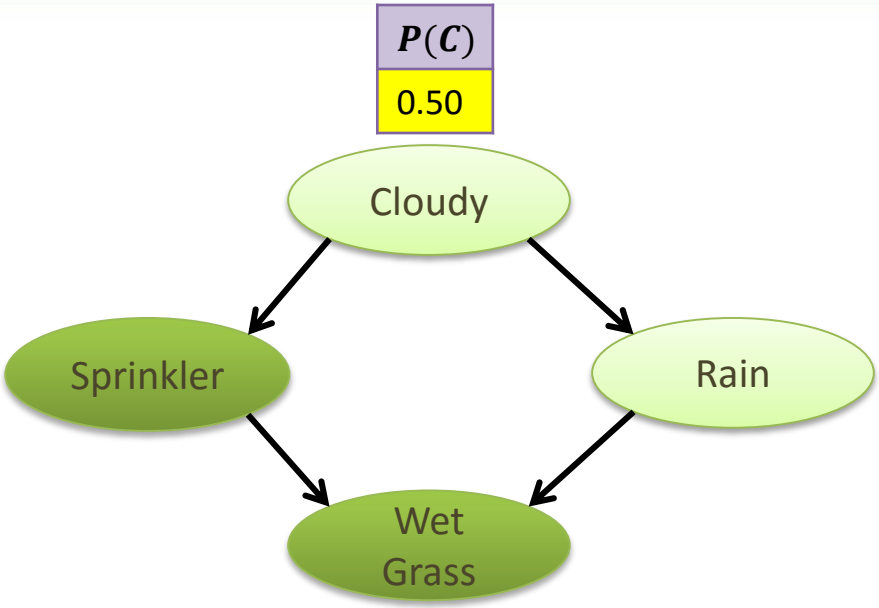
C	$P(S\|C)$
T	0.10
F	0.50

C	$P(R\|C)$
T	0.80
F	0.20

S	R	$P(W\|S,R)$
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.01

$w = 1.0 * 0.1$

Example



$P(C)$
0.50

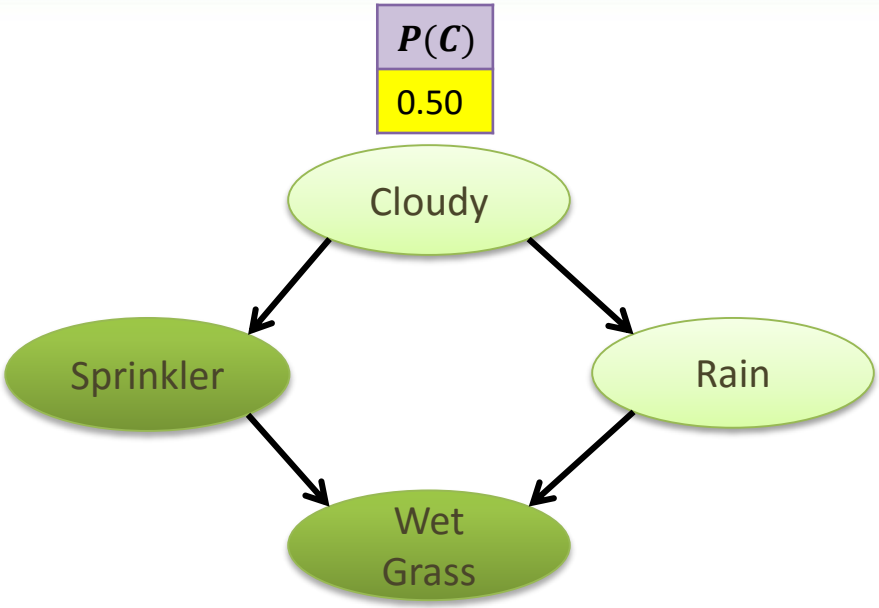
C	$P(S\|C)$
T	0.10
F	0.50

C	$P(R\|C)$
T	0.80
F	0.20

S	R	$P(W\|S,R)$
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.01

$w = 1.0 * 0.1$

Example



$P(C)$
0.50

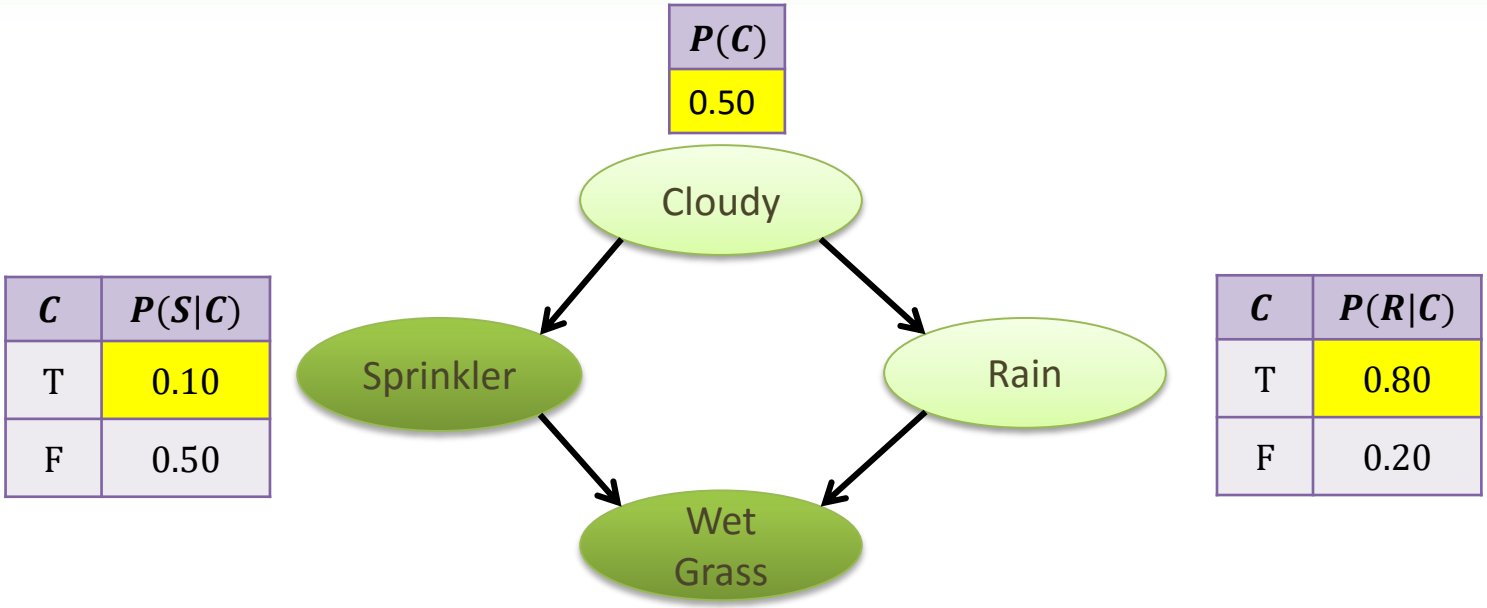
C	$P(S\|C)$
T	0.10
F	0.50

C	$P(R\|C)$
T	0.80
F	0.20

S	R	$P(W\|S,R)$
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.01

$w = 1.0 * 0.1$

Example



C	$P(S\|C)$
T	0.10
F	0.50

C	$P(R\|C)$
T	0.80
F	0.20

S	R	$P(W\|S,R)$
T	T	0.99
T	F	0.90
F	T	0.90
F	F	0.01

$$w = 1.0 * 0.1 * 0.99 = 0.099$$

Approximate inference using MCMC

Unlike previous sampling approach, MCMC generate each sample by making a random change to preceding sample.

Actually, the sampling method is used, is one of famous MCMC sampling approaches: ***Gibbs sampling****.*

We want to sample from $f_X(x_1, x_2, x_3)$ *but we have problem to get sample directly, but we can sample from* $\forall i\ f_X(x_i|x_{-i})$ *easily.*

We follow this procedure:

1- we begin with some initial value (a, b, c)
2-Doing this to obtain the number of sample you want:

- *1- sample from* $f_X(x_1|x_2 = b, x_3 = c) \rightarrow (d, b, c)$
- *2- sample from* $f_X(x_2|x_1 = d, x_3 = c) \rightarrow (d, e, c)$
- *3- sample from* $f_X(x_3|x_1 = d, x_2 = e) \rightarrow (d, e, g)$

Approximate inference using MCMC

Here our goal is sample from $P(ne|e)$. By using Gibbs sampling and Markov Blanket we can sample efficiently.

Generate next state by sampling one variable given Markov blanket.

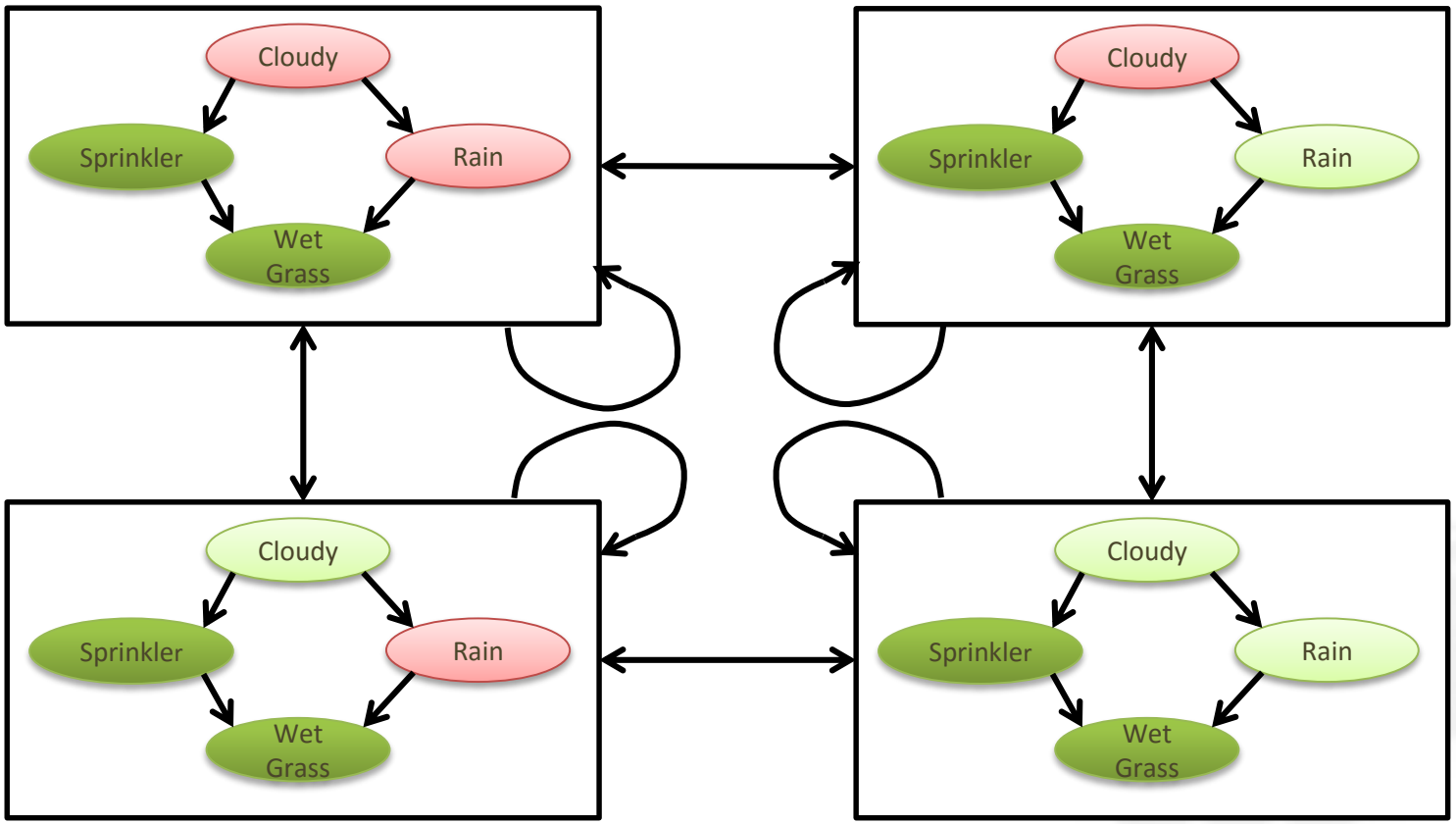
Sample each variable in turn, keeping evidence fixed

```
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 variable in bn
                    x, the current state of the network, initially
                     copied from e

   initialize x with random values for the variables
   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])
```

The Markov chain

With $Sprinkler = true$ *,* $WetGrass = true$*, there are four state:*



MCMC example contd.

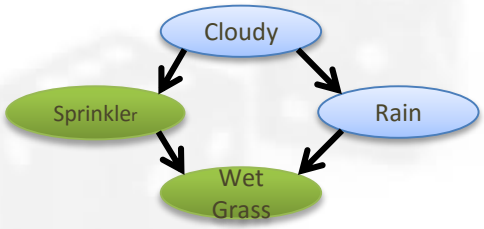
Estimate $P(Rain|Sprinkler = true, WetGrass = true)$

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 state
31 have $Rain = true$, *69 have* $Rain = false$

$\hat{P}(Rain|Sprinkle = true, WetGrass = true)$
$= \text{NORMALIZE}\left((31,69)\right) = (0.31,0.69)$

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



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

Approximate inference using MCMC

How can we sample from $P(X|mb(X))$ *?*

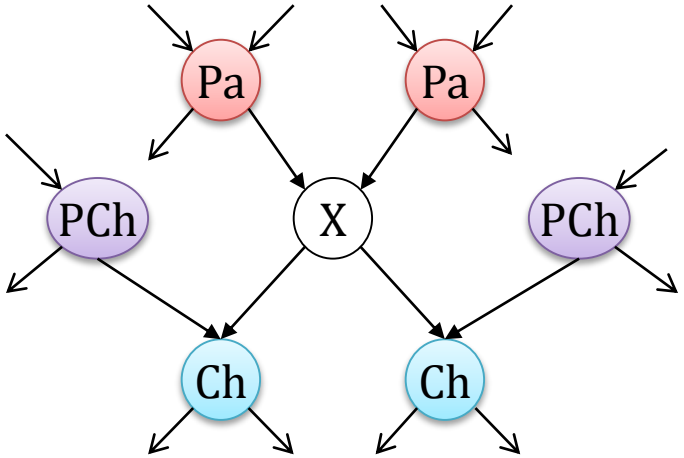
$$P(X|Pa,Ch,PCh) = \alpha P(X,Pa,Ch,PCh)$$

$$= \alpha P(Ch|X,Pa,PCh)P(X,Pa,PCh)$$

$$= \alpha P(Ch|X,PCh)P(X|PCh,Pa)P(PCh,Pa)$$

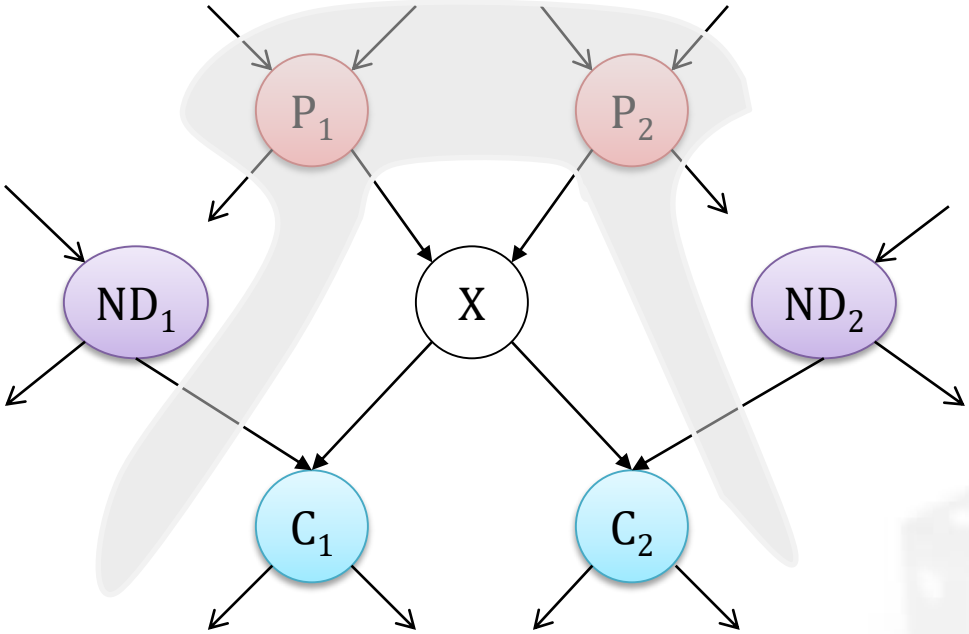
$$= \beta P(Ch|X,PCh)P(X|Pa)$$

$$= \boldsymbol{\beta P(X|Parent(X))\prod_i P(Ch_i|Parent(Ch_i))}$$



Local Semantics

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



Markov Blanket

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