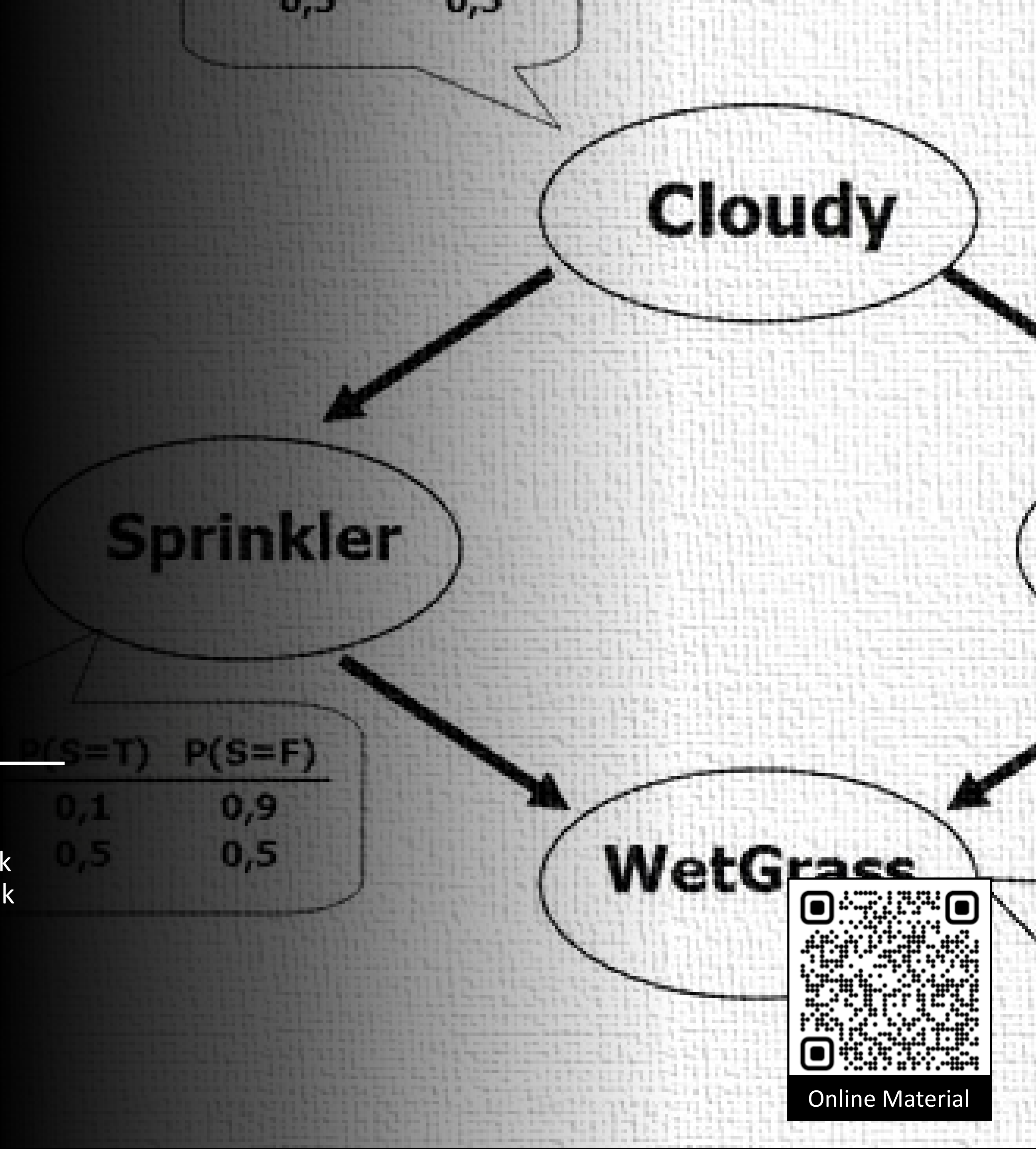
CS 5/7320 Artificial Intelligence

Probabilistic Reasoning:

Bayesian Networks

AIMA Chapter 13

Slides by Michael Hahsler
based on slides by Svetlana Lazepnik
with figures from the AIMA textbook



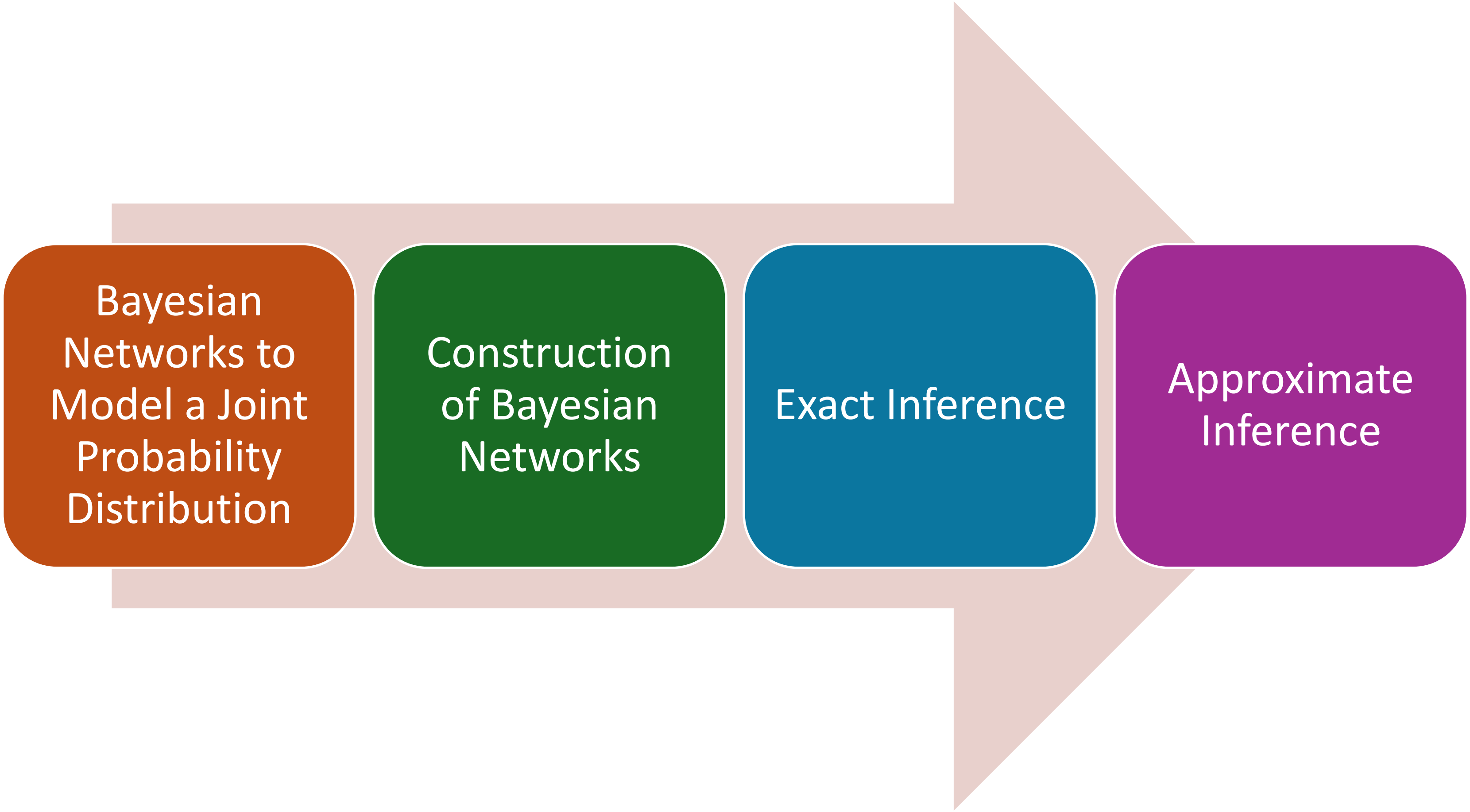
Online Material

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

Probability Theory Recap

- Notation: Prob. of an event $P(X = x) = P_X(x) = P(x)$
  Prob. distribution $\boldsymbol{P}(X) = \langle P(X = x_1), P(X = x_2), \ldots, P(X = x_n)\rangle$
- Product rule $P(x, y) = P(x|y)P(y)$
- Chain rule $\boldsymbol{P}(\mathrm{X}_1, \mathrm{X}_2, \ldots, \mathrm{X}_\mathrm{n}) = \mathbf{P}(\mathrm{X}_1)\mathbf{P}(\mathrm{X}_2|\mathrm{X}_1)\mathbf{P}(\mathrm{X}_3|\mathrm{X}_1, \mathrm{X}_2) \ldots$
  $= \prod_{i=1}^{n} \boldsymbol{P}(X\_i|X_1, \ldots, X_{i-1})$
- Conditional probability $P(x|y) = \frac{P(x,y)}{P(y)} = \alpha P(x, y)$
- Marginal distribution given $\boldsymbol{P}(X, Y)$
  $\boldsymbol{P}(X) = \sum_y \boldsymbol{P}(X, y)$ (called marginalizing out Y)
- Independence
  - $X \perp\!\!\!\perp Y$: X, Y are independent (written as $X \perp\!\!\!\perp Y$) if and only if:
    $\forall x, y: P(x, y) = P(x)P(y)$
  - $X \perp\!\!\!\perp Y|Z$: X and Y are conditionally independent given Z if and only if:
    $\forall x, y, z: P(x, y|z) = P(x|z)P(y|z)$

Contents

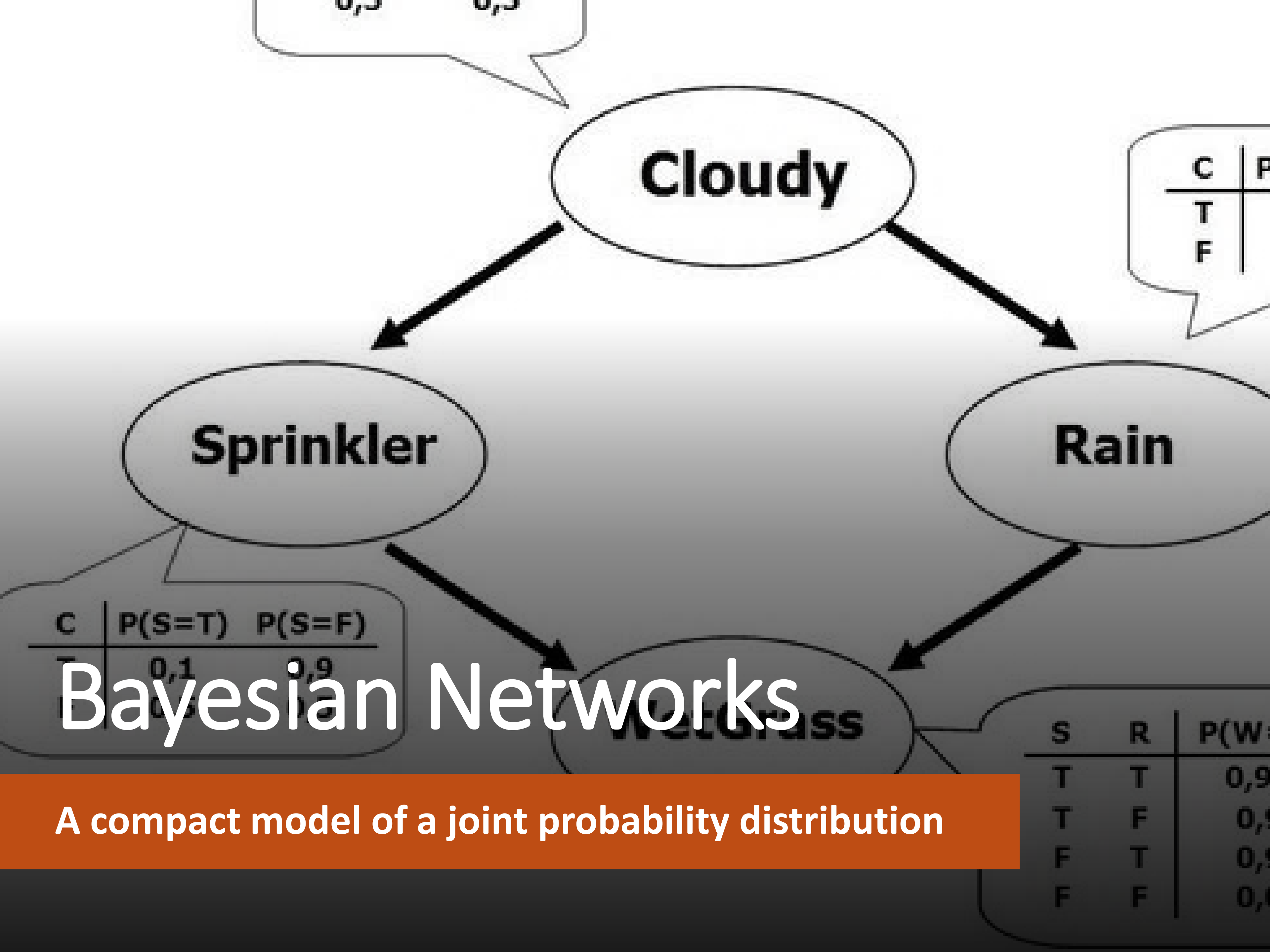


Bayesian Networks to Model a Joint Probability Distribution

Construction of Bayesian Networks

Exact Inference

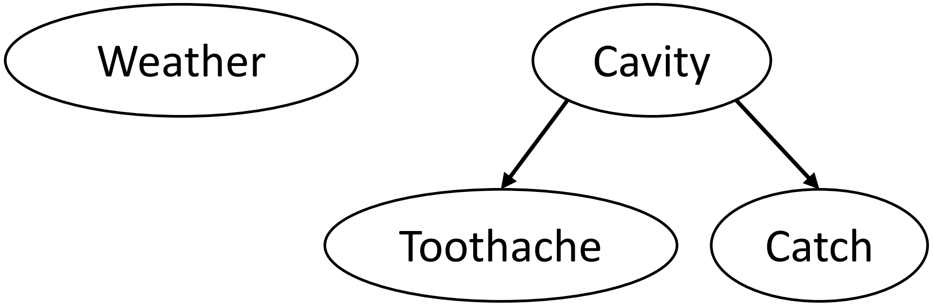
Approximate Inference



Bayesian Networks

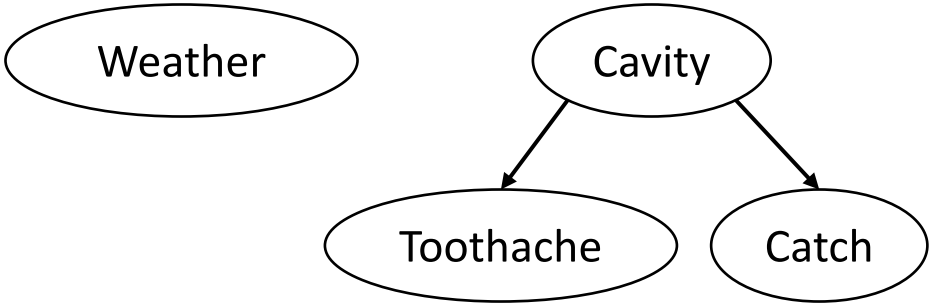
A compact model of a joint probability distribution

Bayesian Networks
(aka Belief Networks)



- A type of graphical model.
- A way to specify dependence between random variables.
- A compact specification of a full joint probability distribution.
- A general and important model to reason with uncertainty in AI.

Structure of Bayesian Networks



Nodes: Random variables

- Can be assigned (observed) or unassigned (unobserved)

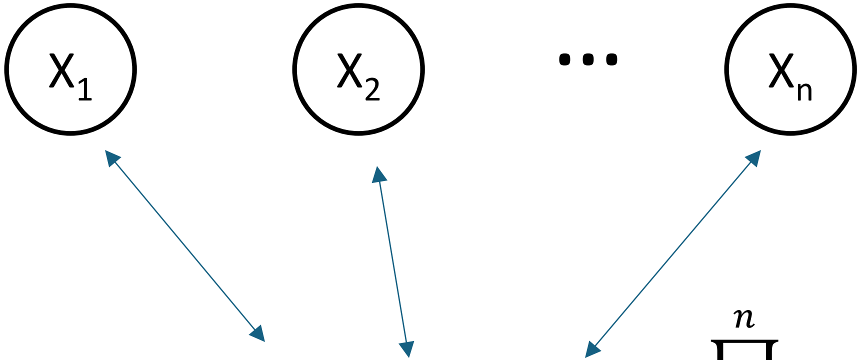
Edges: Dependencies

- An arrow from one variable to another indicates **direct influence**.
- Show independence
  - *Weather* is **independent** of the other variables (no connection).
  - *Toothache* and *Catch* are **conditionally independent** given *Cavity* (directed arc).
- Must form a **directed acyclic graph** (DAG).

Relationship to states in AI: A network can be seen as a factored state representation. If we assign a value to all random variables, then we have a complete state of the system.

Example: N Independent Coin Flips

Complete independence: no interactions between variables representing the coin flips.



$$P(X_1, X_2, \dots, X_n) = P(X_1)P(X_2)\dots P(X_n) = \prod_{i=1}^{n} P(X_i)$$

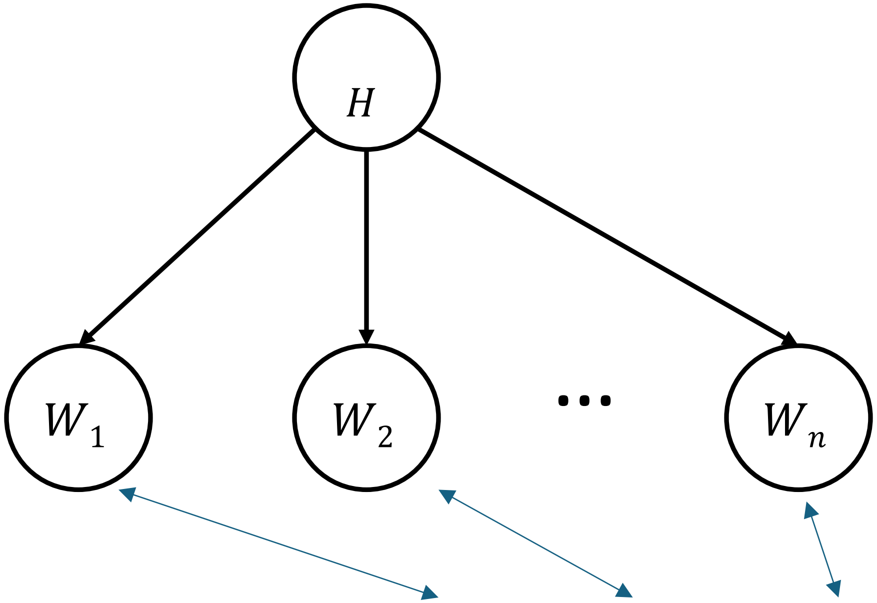
Joint probability distribution

Example: Naïve Bayes Spam Filter

Random variables:

- H: message class (spam or not spam)
- $W_1, \ldots, W_n$: presence or absence of words comprising the message

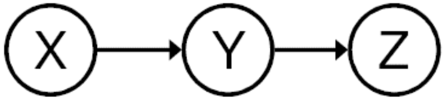
Words depend on the class, but they are modeled conditional independent of each other given the class (= no direct connection between words).

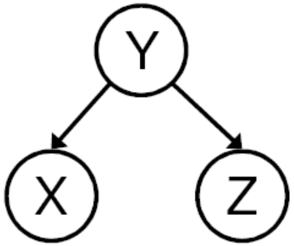


$$P(W_1, W_2, \ldots, W_n | H) = P(W_1|H)P(W_2|H) \ldots P(W_n|H)$$

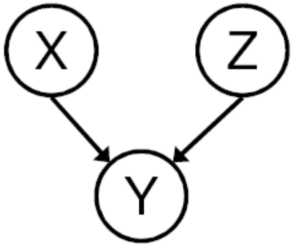
Common Structures

1. Causal Chains





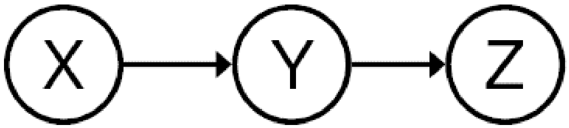
2. Common Cause



1. Common Effect

Causal Chains: Dependence

- Example: *causal chain*



X: Low pressure

Y: Rain

Z: Traffic

- Are X and Z independent?

1. Conditioning: $P(X,Y,Z) = P(X)P(Y|X)P(Z|Y)$
2. Marginalize over y: $P(X,Z) = \sum_y P(X)P(y|X)P(Z|y)$

$$= P(X) \sum_y P(Z|y)P(y|X) \neq P(X)P(Z)$$

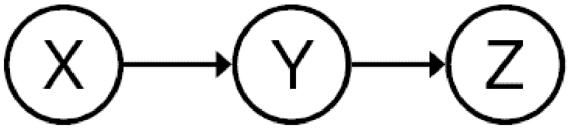
We are not interested in y.

X and Z are **not** independent!

Note: most Ps refer to probability distributions, but for convenience, we omit setting using a bold **P**!

Causal Chains: Conditional Independence

- Example: *causal chain*



X: Low pressure

Y: Rain

Z: Traffic

- Is Z independent of X given Y?

1. Conditioning: $P(X,Z|Y) = \frac{P(X,Y,Z)}{P(Y)} = \frac{\mathrm{P(X)P(Y|X)P(Z|Y)}}{P(Y)}$

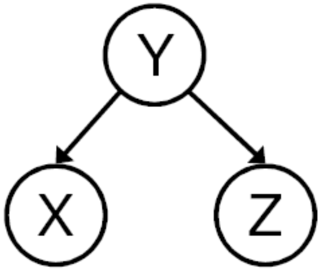
2. Bayes' rule: $= \frac{P(X)\frac{P(X|Y)P(Y)}{P(X)}P(Z|Y)}{P(Y)} = \mathrm{P(X|Y)P(Z|Y)}$

= Definition of conditional independence

X and Z are conditionally independent given Y

Common Cause vs. Common Effect

- *Common cause*



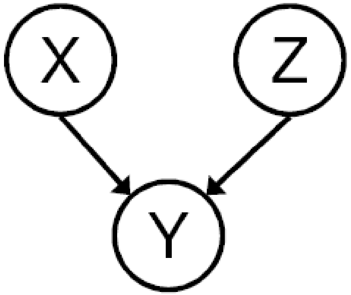
Y: Project due

X: Newsgroup busy

Z: Lab full

- Are X and Z independent?
  - No. If Y is unknown, then it will influence both in the same way.
- Are they conditionally independent given Y?
  - Yes, the only variation comes from other, not shared, causes.

- *Common effect*



X: Raining

Z: Ballgame

Y: Traffic

- Are X and Z independent?
  - Yes, they do not share a parent.
- Are they conditionally independent given Y?
  - No. The observed Y can now be seen as a common cause for X and Z resulting in dependence.

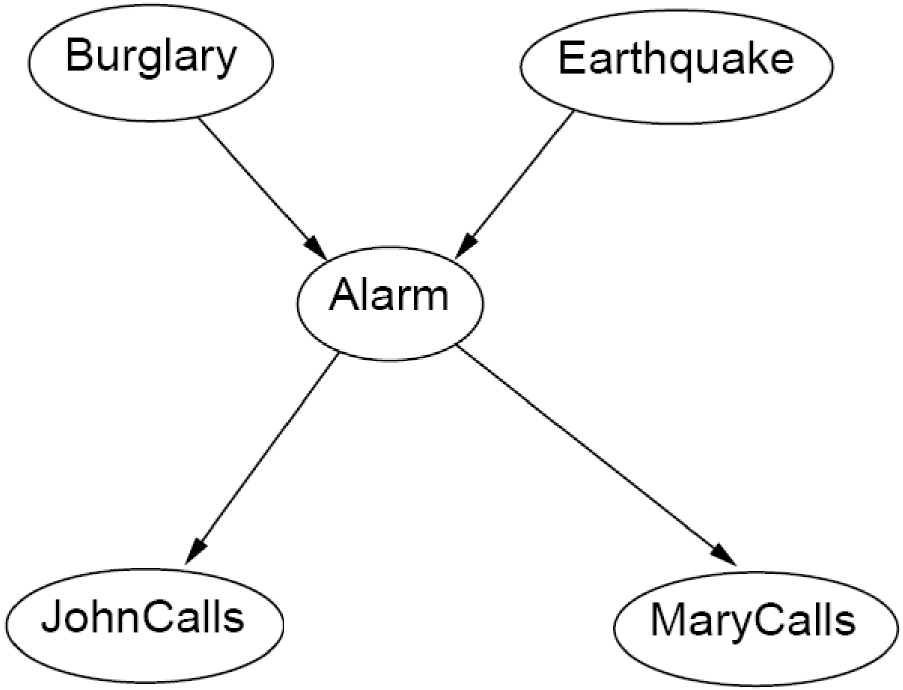
Example: Burglar Alarm

- **Description**: I have a **burglar alarm** that is sometimes set off by minor **earthquakes**. My two neighbors, **John** and **Mary**, promised to call me at work if they hear the alarm.
- Example inference task: Suppose Mary calls, and John doesn't call. What is the probability of a burglary?

- What are the random variables?
  - Burglary, Earthquake, Alarm, JohnCalls, MaryCalls

- What are the direct influence relationships?
  - A burglar can set off the alarm
  - An earthquake can set off the alarm
  - The alarm can cause Mary to call
  - The alarm can cause John to call

Example: Burglar Alarm as a Network



Direct influence relationships:

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

What are the probabilities?

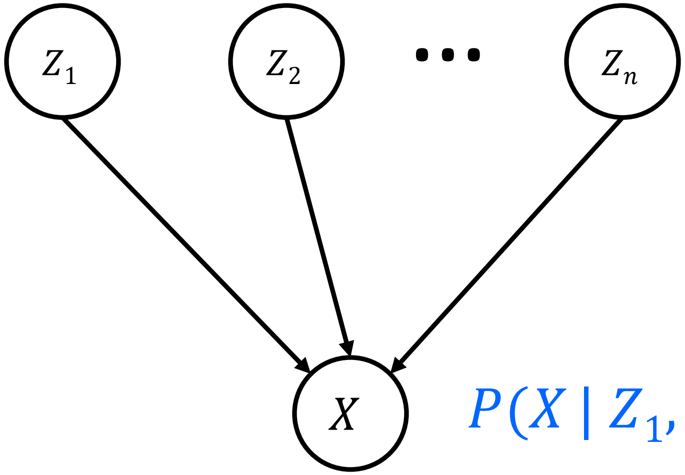
Model Parameters: Conditional Probability Tables (CPTs)

The full joint distribution, can be broken down into *conditional* distribution for each node given its parents:

$$P(X \mid Parents(X))$$

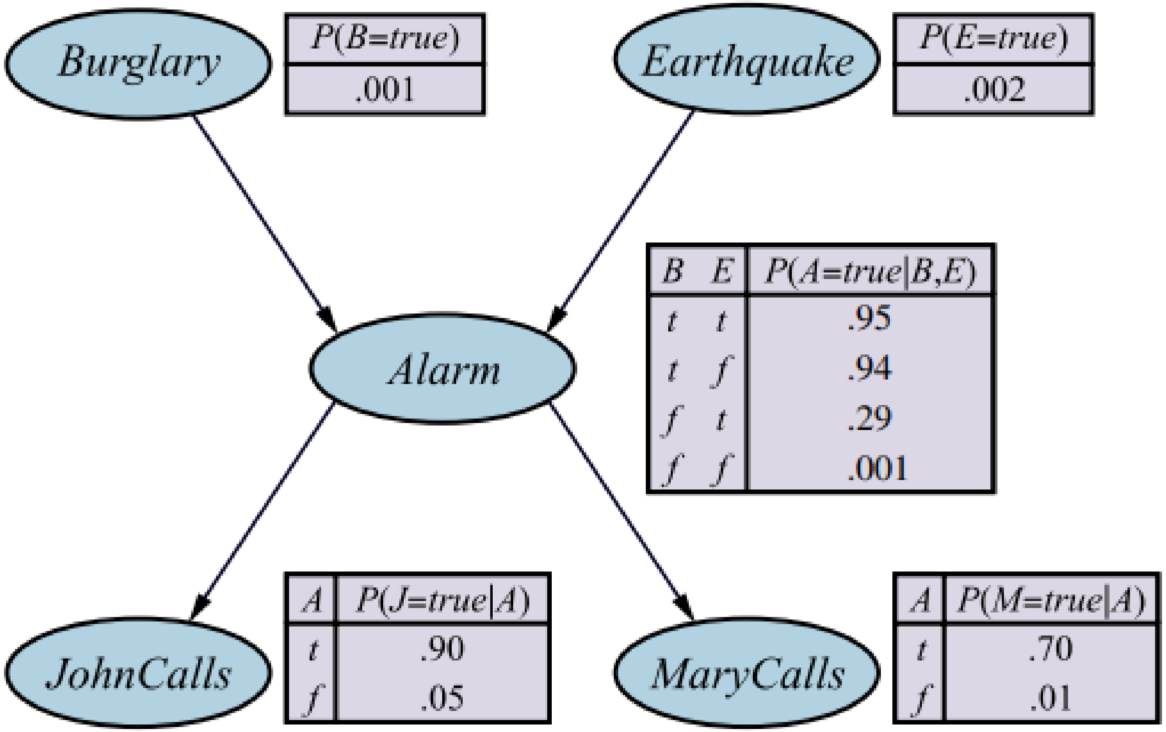
These distributions are stored in conditional probability tables (CPTs)

Example:



$P(X \mid Z_1, \ldots, Zn)$

Example: Burglar Alarm with CPTs



P(B=true)
.001

P(E=true)
.002

No parents

B	E	P(A=true|B,E)
t	t	.95
t	f	.94
f	t	.29
f	f	.001

2 parents

A	P(J=true|A)
t	.90
f	.05

A	P(M=true|A)
t	.70
f	.01

1 parent

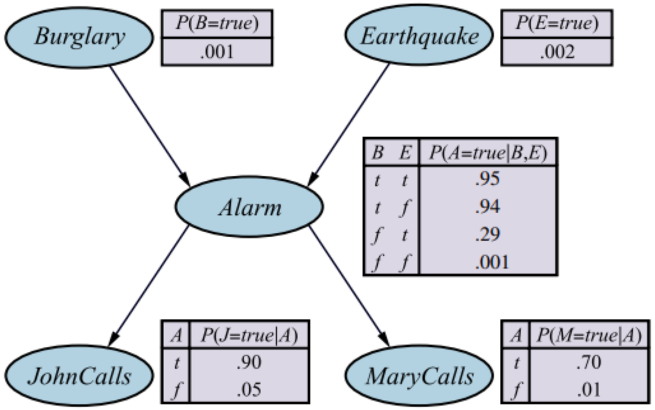
Extracting the Joint Probability Distribution

- For each node X_i, we know the CPT $P(X_i \mid Parents(X_i))$
- How do we get the full joint distribution $P(X_1, \ldots, Xn)$?

- Using chain rule, but only depends on parents:

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

Example:



P(B=true)
.001

P(E=true)
.002

B	E	P(A=true|B,E)
t	t	.95
t	f	.94
f	t	.29
f	f	.001

A	P(J=true|A)
t	.90
f	.05

A	P(M=true|A)
t	.70
f	.01

Construct following arrows

$$P(J, M, A, B, E) = P(B)\, P(E)\, P(A \mid B, E)\, P(J \mid A)\, P(M \mid A)$$

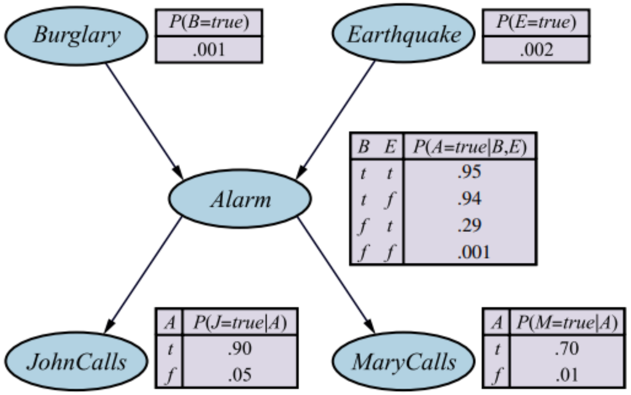
Compactness

- For a network with n boolean variables, the full joint distribution requires $O(2^n)$ probabilities.
  Example: Burglary network
  Joint probability: $2^5 - 1 = \mathbf{31}$ entries

- If each variable X_i has at most k boolean parents, then each conditional probability table (CPT) has at most 2^k rows. The CPTs for all n nodes contain then at most $O(n \times 2^k) = O(n)$ probabilities.
  Example: Burglary network
  Using CPTs: $1 + 1 + 4 + 2 + 2 = \mathbf{10}$

- **This reduces the space complexity from $\boldsymbol{O(2^n)}$ to $\boldsymbol{O(n)}$ and lets us store very large networks!**

- **Note**: The Bayesian network stores all information needed for the complete joint probability. It let's us make optimal Bayesian decisions!



P(B=true)
.001

P(E=true)
.002

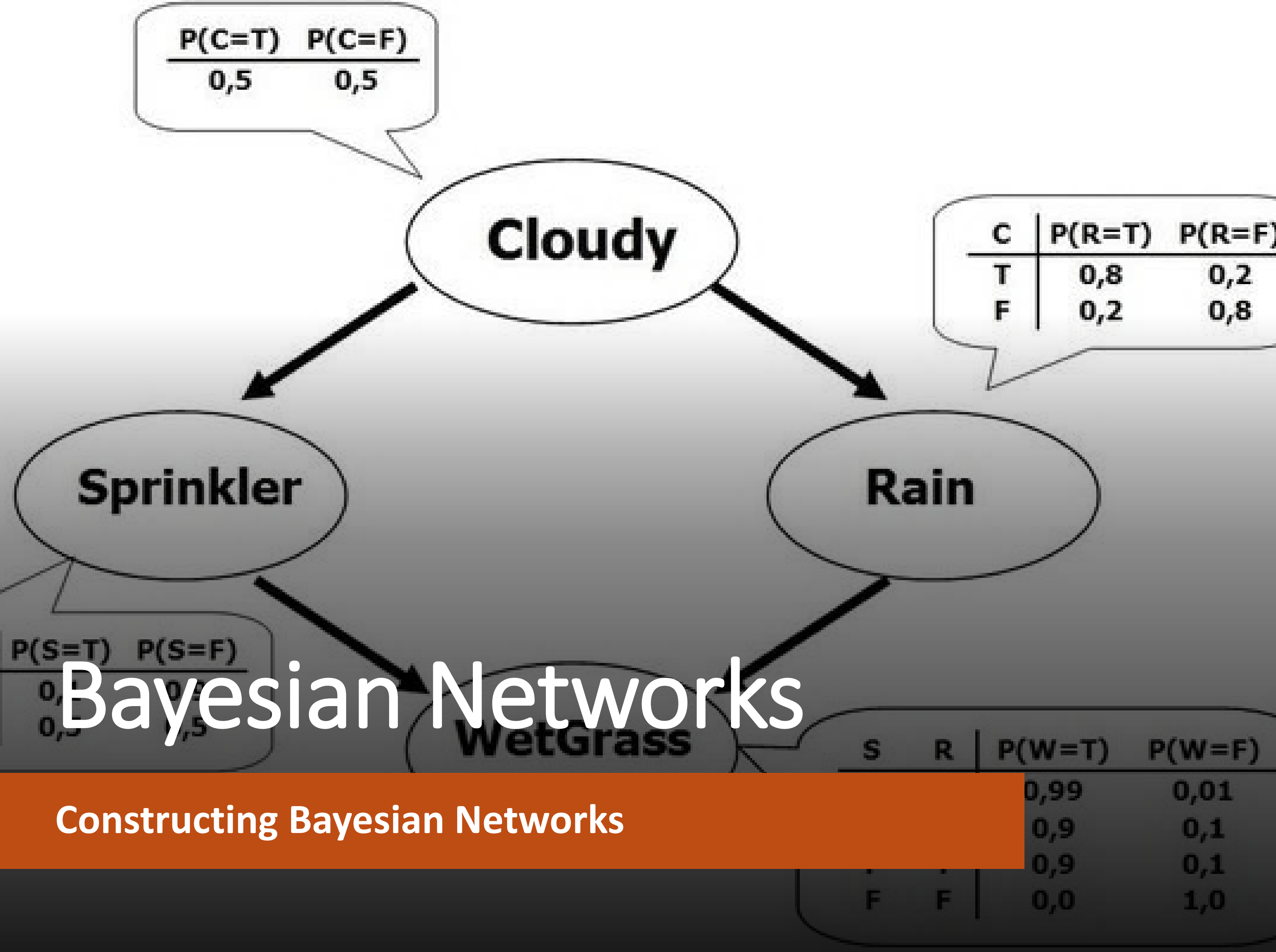
B	E	P(A=true|B,E)
t	t	.95
t	f	.94
f	t	.29
f	f	.001

A	P(J=true|A)
t	.90
f	.05

A	P(M=true|A)
t	.70
f	.01

Bayesian Networks

Constructing Bayesian Networks



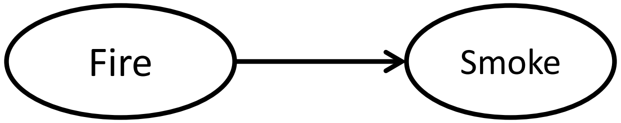
Constructing Bayesian Networks

Once we know what variables to use, we need to create the structure of the network and create the conditional probability tables (CPTs).

Algorithm

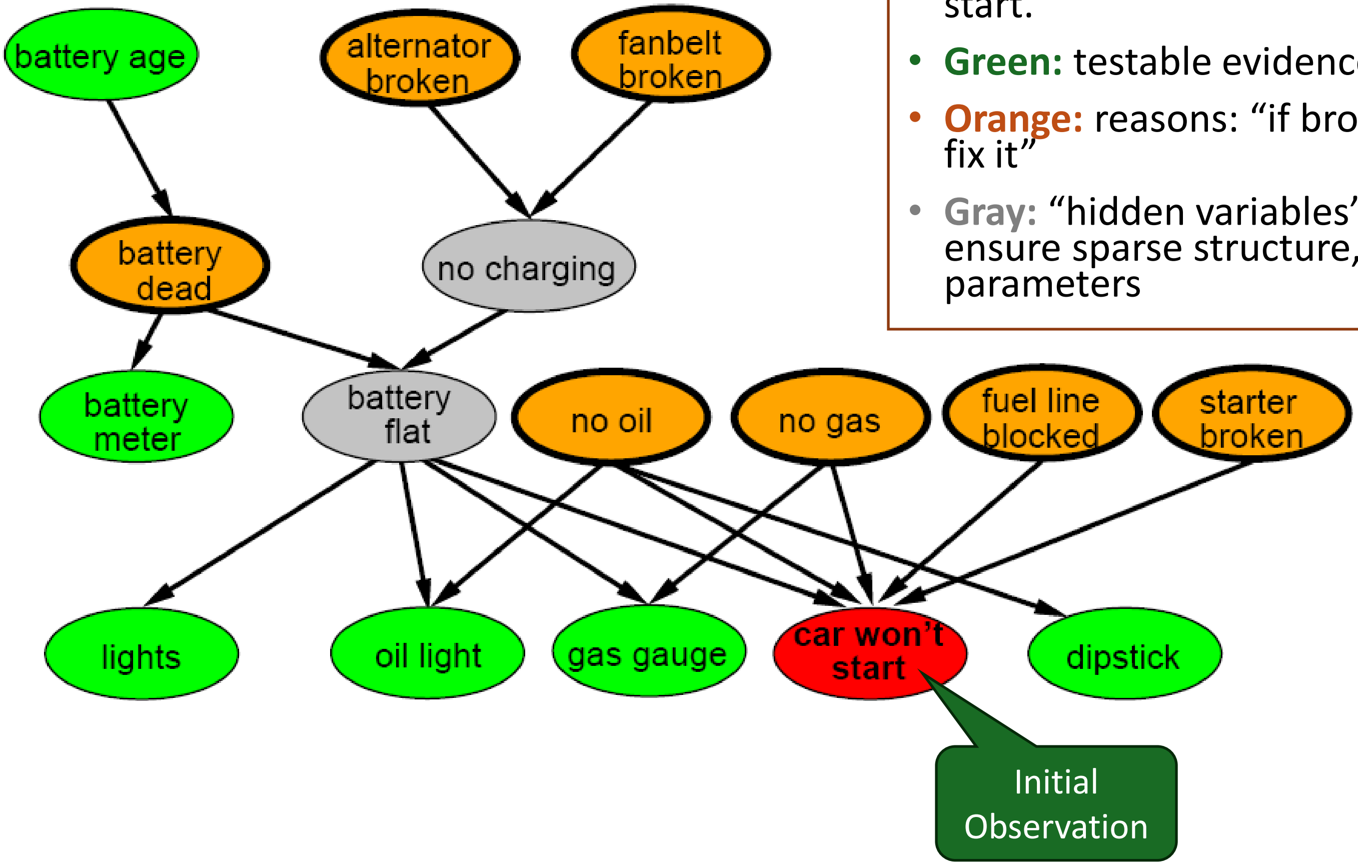
```
1.  Choose an ordering of variables $X_1, \ldots, X_n$
2.  For $i = 1$ to $n$
      •  add $X_i$ to the network
      •  select parents from $X_1, \ldots, X_{i-1}$ such that
           $P(X_i \mid Parents(X_i)) = P(X_i \mid X_1, \ldots X_{i-1})$
         that is, add a connection only from nodes it directly depends on.
```

Note: There are many ways to order the variables. Networks are typically constructed by domain experts with causality in mind. E.g., Fire causes Smoke:



The network resulting from causal ordering is typically sparse and conditional probabilities are easier to judge because they represent causal relationships.

A Larger Bayes Network: Car diagnosis



- **Initial observation**: car won’t start.
- **Green:** testable evidence.
- **Orange:** reasons: “if broken, then fix it”
- **Gray:** “hidden variables” to ensure sparse structure, reduce parameters

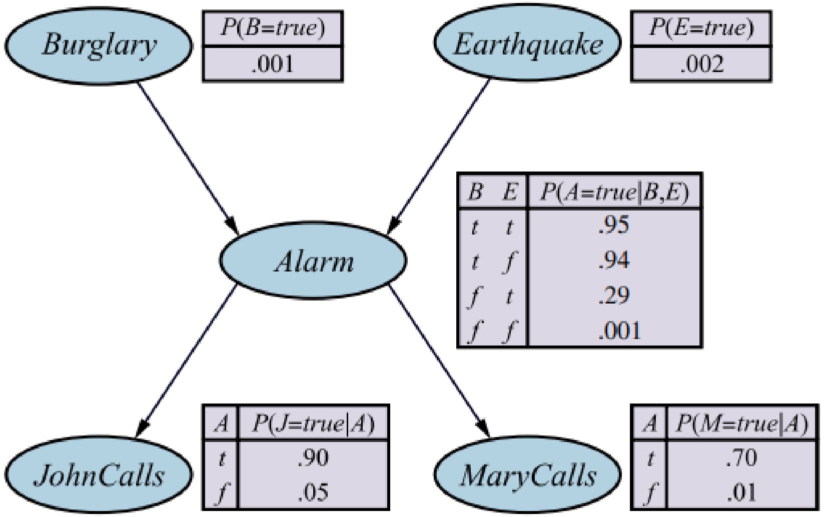
- The network was constructed by an expert with causality in mind.
- The CPTs specify the joint probability distribution.
- We can compute conditional probabilities like $P(\text{battery dead} \mid \text{car won't start})$ vs. $P(\text{starter broken} \mid \text{car won't start})$

Summary

- Bayesian networks provide a **natural representation for joint probabilities** used to calculate conditional probabilities needed for inference (prediction).
- **Conditional independence** (induced by causality) reduces the number of needed parameters and creates a compact network.
- Bayesian networks let us make optimal decisions as long as independence assumptions hold.

- Representation
  - Topology (nodes and edges)
  - Conditional probability tables (CPTs)
- Construction
  - Typically easy for domain experts
  - to construct thinking about causality.
- Use
  - Calculating conditional probabilities for decision making is called inference.

$P(B, E, A, J, M)$ is defined by



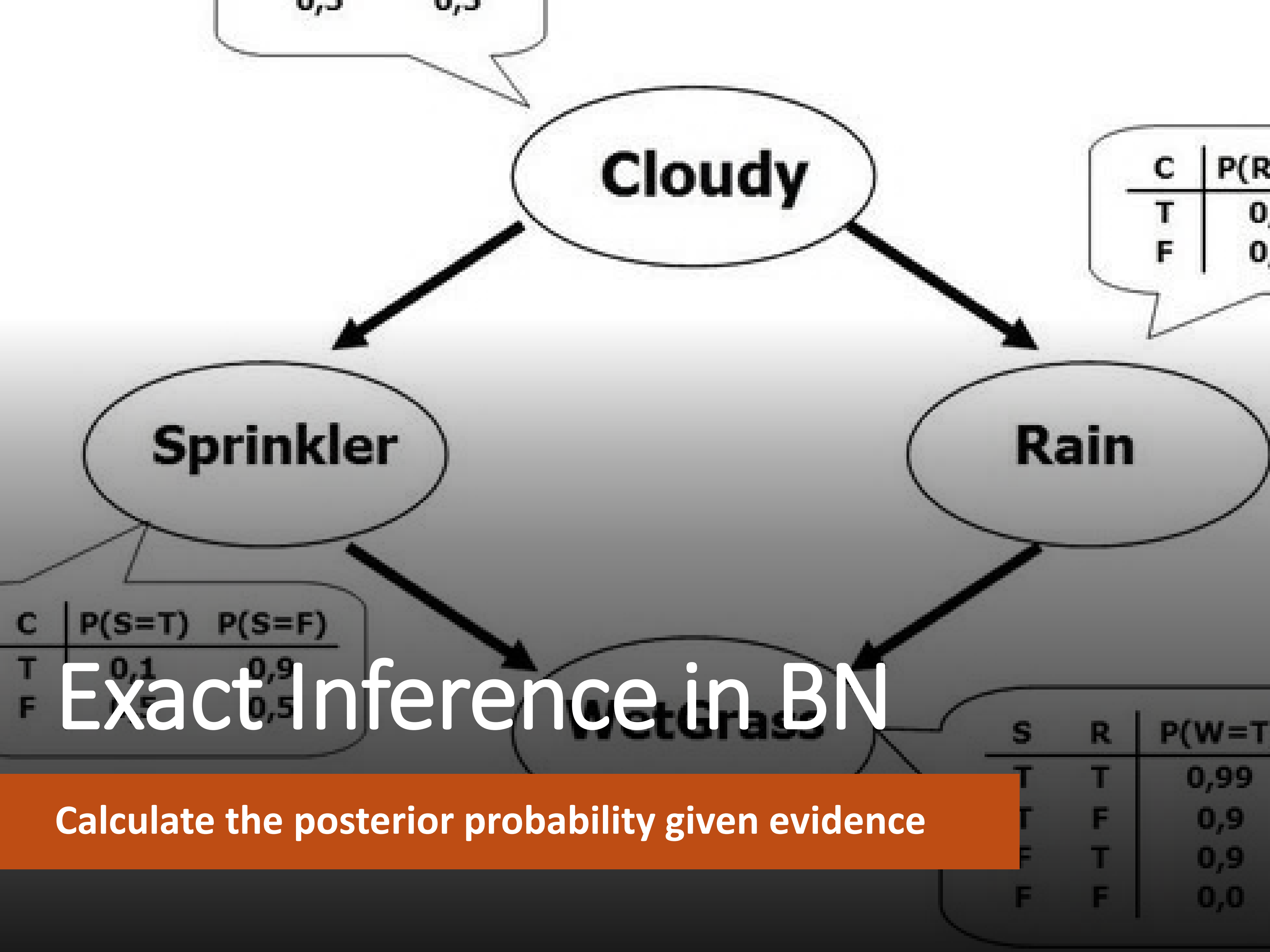
P(B=true)
.001

P(E=true)
.002

B	E	P(A=true|B,E)
t	t	.95
t	f	.94
f	t	.29
f	f	.001

A	P(J=true|A)
t	.90
f	.05

A	P(M=true|A)
t	.70
f	.01



Exact Inference in BN

Calculate the posterior probability given evidence

Exact Inference: Calculating Conditional Distributions

Goal

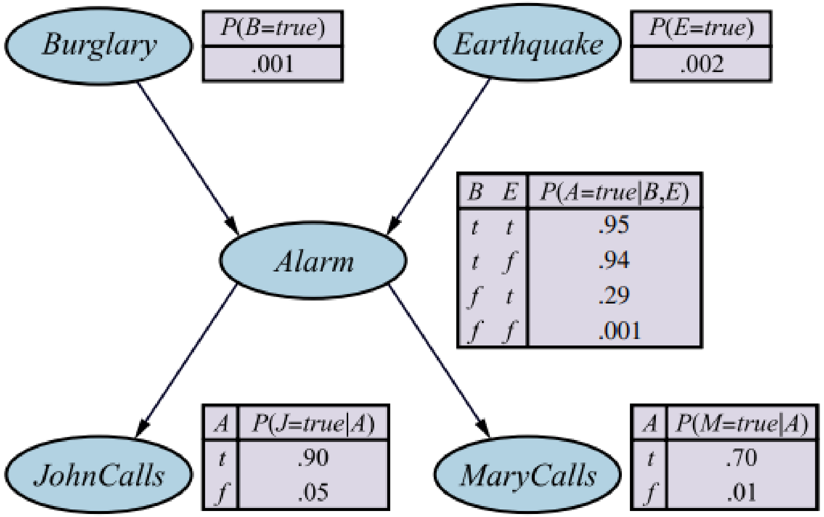
- Query *variables:* X
- *Evidence* (*observed*) event: $E = e$
- *Set of unobserved* variables: Y
- Calculate the probability of X given e.

If we know the full joint distribution $P(X, E, Y)$, we can infer X by:

$$\boldsymbol{P}(X|E=e) = \frac{\boldsymbol{P}(X,e)}{P(e)} = \frac{\sum_y \boldsymbol{P}(X,e,y)}{P(e)} \propto \sum_y \boldsymbol{P}(X,e,y)$$

Sum over values of unobservable variables = marginalizing them out.

Exact inference: Example – Calculation



P(B=true)
.001

P(E=true)
.002

B	E	P(A=true|B,E)
t	t	.95
t	f	.94
f	t	.29
f	f	.001

A	P(J=true|A)
t	.90
f	.05

A	P(M=true|A)
t	.70
f	.01

Assume we can observe being called and the two variables have the values j and m.
We want to know the probability of a burglary.

Query: $P(B \mid j, m)$ with unobservable variables: Earthquake E, Alarm A

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

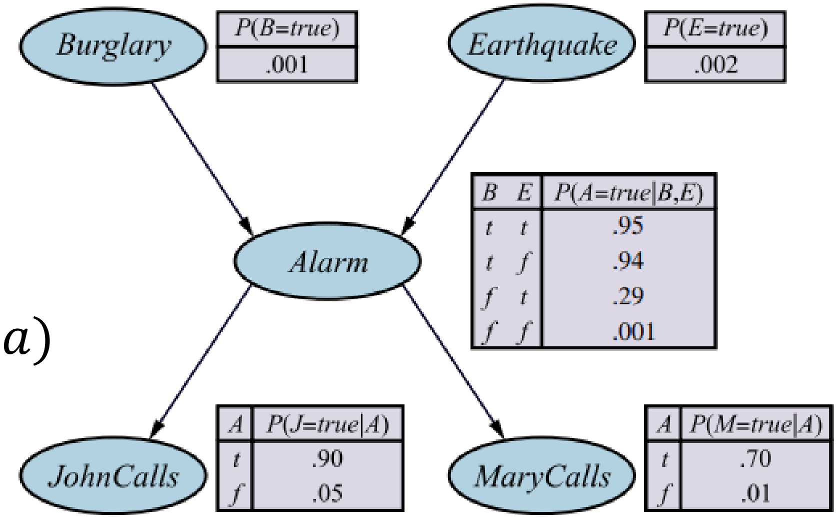
$$= \sum_e \sum_a P(b)P(e)P(a|b,e)P(j|a)P(m|a)$$

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

Full joint probability and marginalize over E and A

Exact inference:
Example – Evaluation Tree

$$P(b|j,m)$$
$$\propto P(b)\sum_{e} P(e)\sum_{a} P(a|b,e)\,P(j|a)P(m|a)$$



P(B=true)
.001

P(E=true)
.002

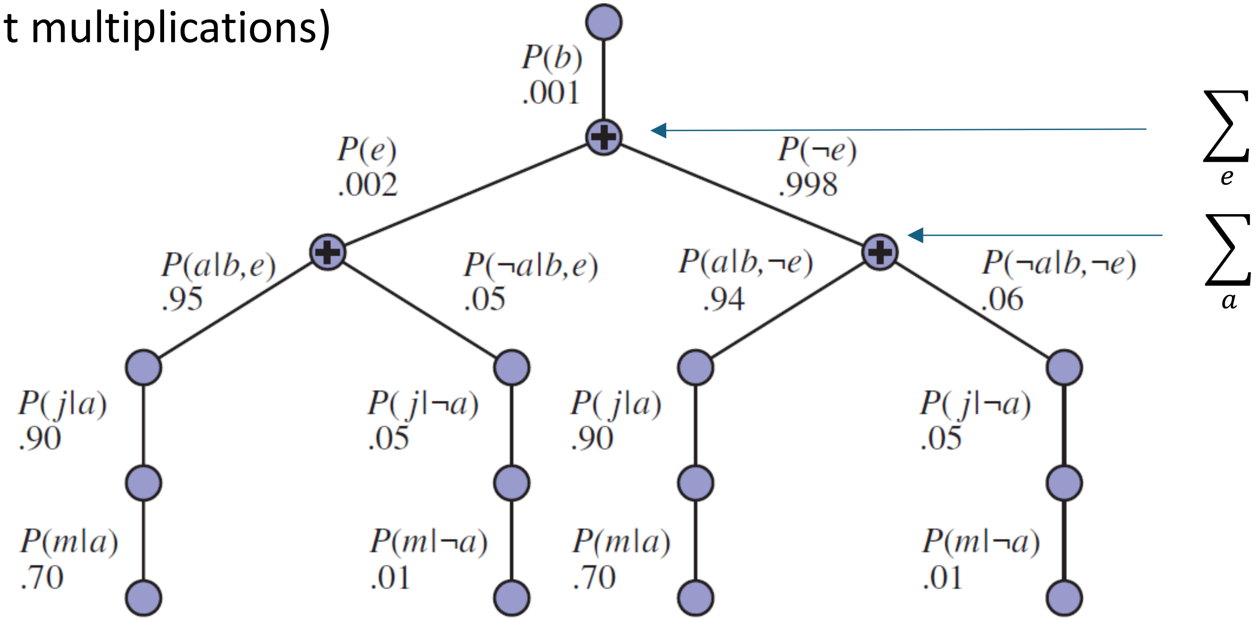
B	E	P(A=true|B,E)
t	t	.95
t	f	.94
f	t	.29
f	f	.001

A	P(J=true|A)
t	.90
f	.05

A	P(M=true|A)
t	.70
f	.01

Implement as a traversal of the evaluation tree

(lines represent multiplications)



Issues with Exact Inference in AI

$$\boldsymbol{P}(X|E=e) = \frac{\boldsymbol{P}(X,e)}{P(e)} \propto \boxed{\sum_{y}} \boxed{\boldsymbol{P}(X,e,y)}$$

Issues

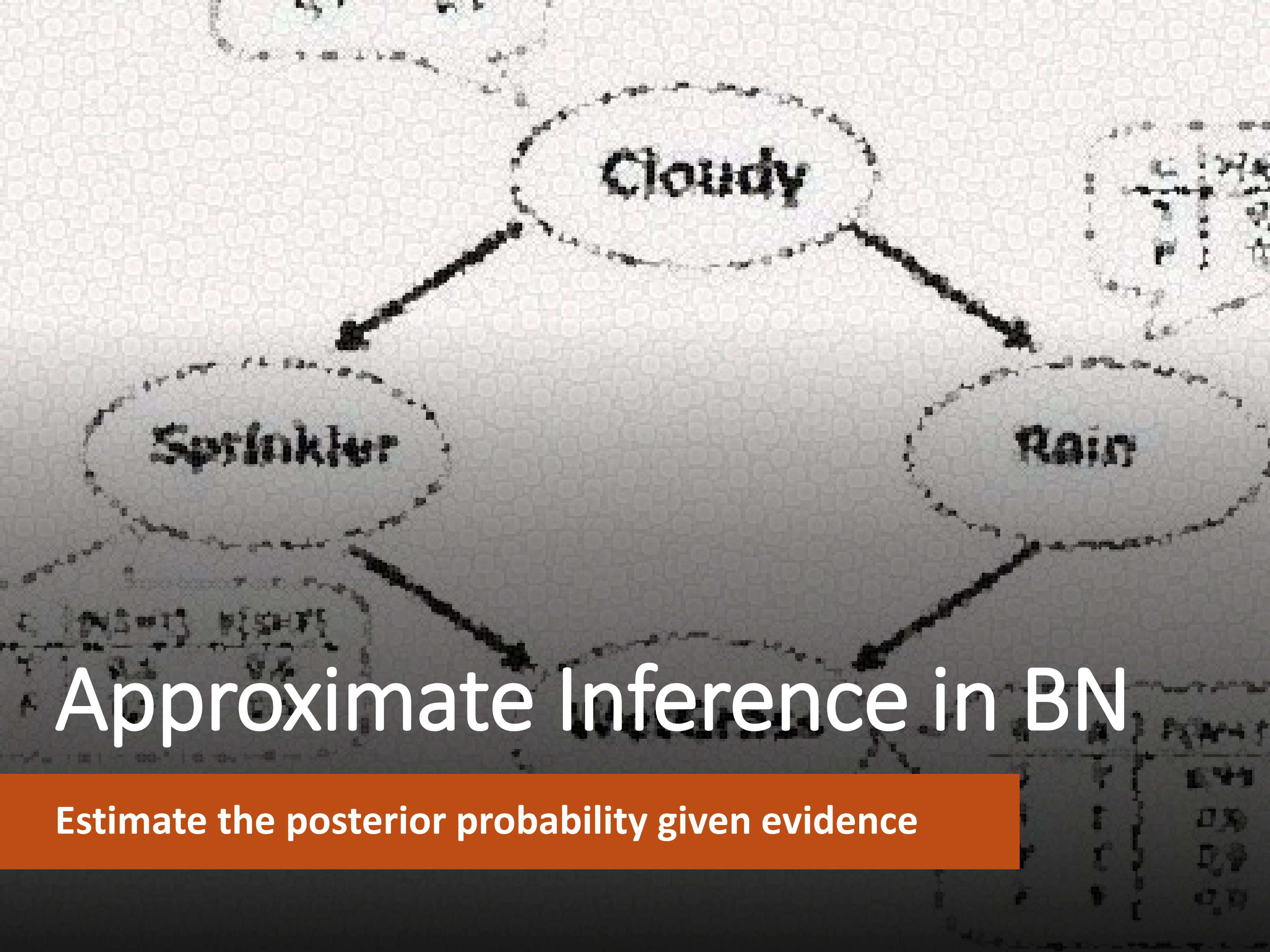
1. The **full joint distribution is too large** to store.

   Bayes nets provide significant savings for representing the conditional probability structure using CPTs.

2. Marginalizing out many unobservable variables Y often involves **too many summation terms**. I.e., the evaluation tree becomes too large for 1000s of random variables.

This summation is called **exact inference by enumeration**. Unfortunately, it does not scale well (#p-hard).

In practice, **approximate inference by sampling** is used.



Approximate Inference in BN

Estimate the posterior probability given evidence

Bayesian Networks as a Generative Models

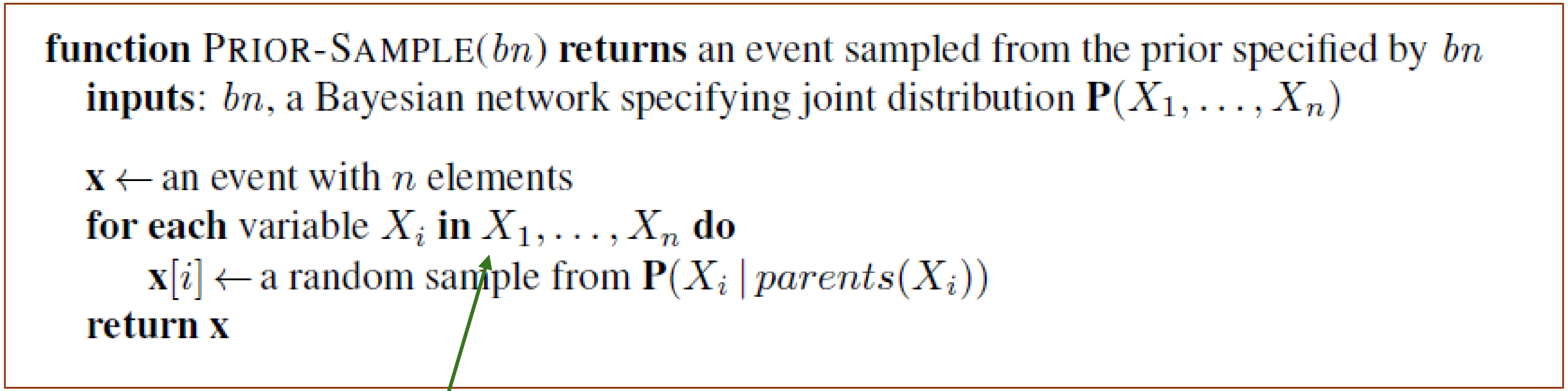
Bayesian networks can be used as ***generative models***.

Generative models allow us to efficiently **generate samples** from the joint distribution.

Idea: Generate samples from the network to estimate joint and conditional probability distributions using **Monte Carlo simulation** methods.

Estimating **Joint and Marginal Probability Distributions** from BNs

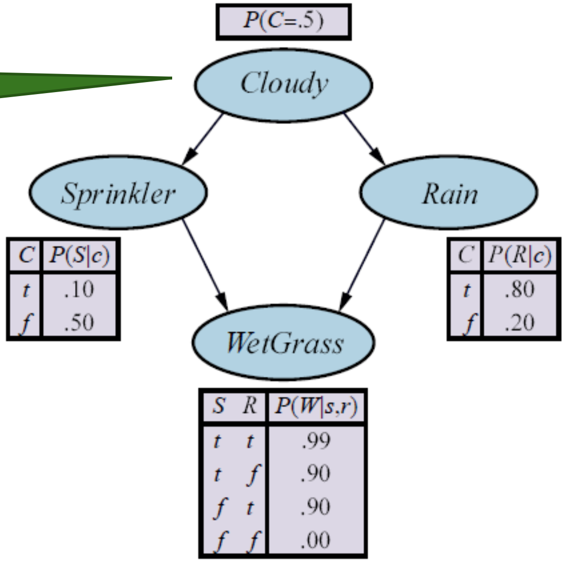
Prior-Sample Algorithm



```
function PRIOR-SAMPLE($bn$) returns an event sampled from the prior specified by $bn$
  inputs: $bn$, a Bayesian network specifying joint distribution $\mathbf{P}(X_1, \ldots, X_n)$

  $\mathbf{x} \leftarrow$ an event with $n$ elements
  for each variable $X_i$ in $X_1, \ldots, X_n$ do
      $\mathbf{x}[i] \leftarrow$ a random sample from $\mathbf{P}(X_i \mid parents(X_i))$
  return $\mathbf{x}$
```

Order is important! We need to start with the random variables that have no parents.



C	P(S|c)
t	.10
f	.50

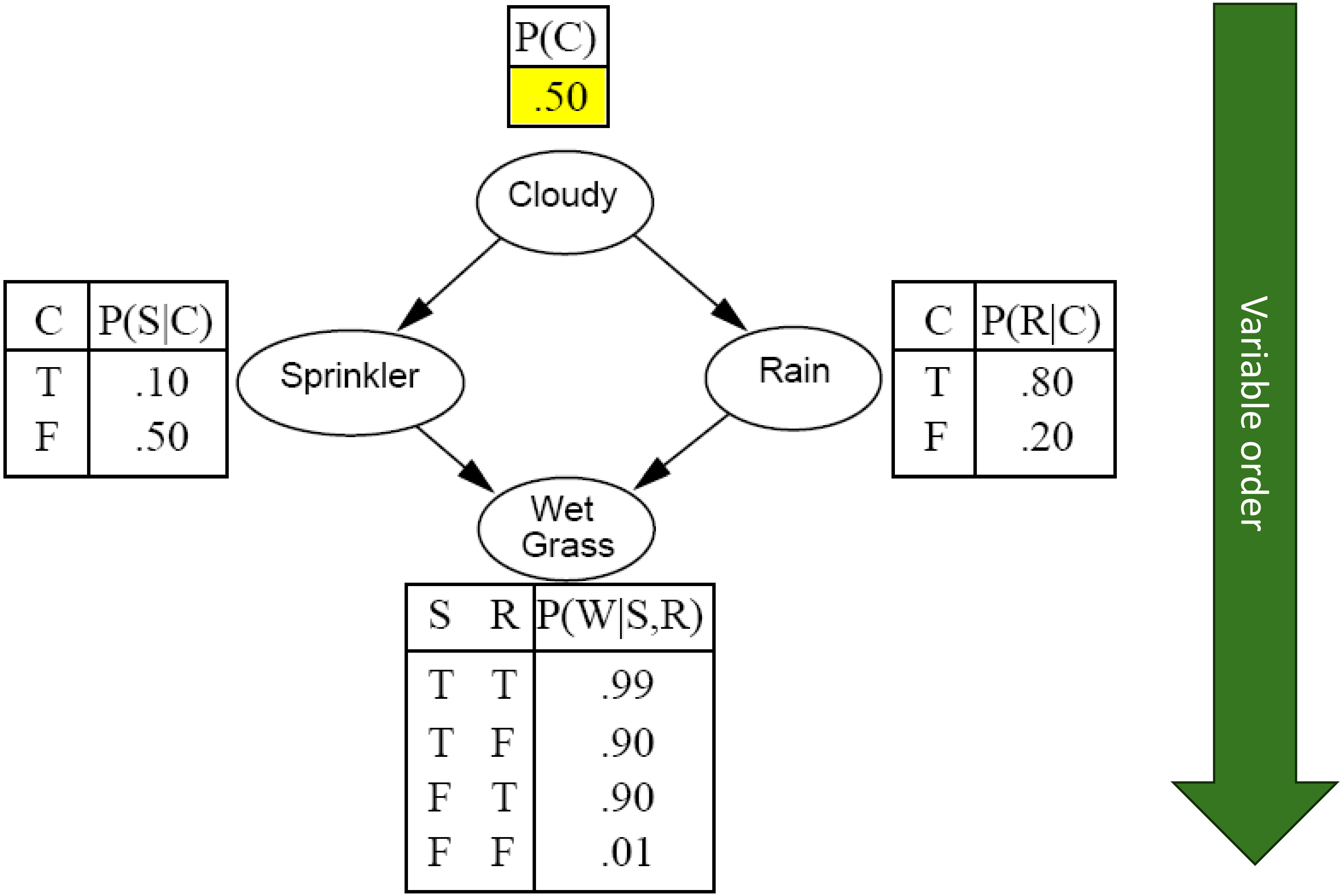
C	P(R|c)
t	.80
f	.20

S	R	P(W|s,r)
t	t	.99
t	f	.90
f	t	.90
f	f	.00

Variable Order

Note: This looks like playouts in games!

Example: Sampling from a Bayesian Network



P(C)
.50

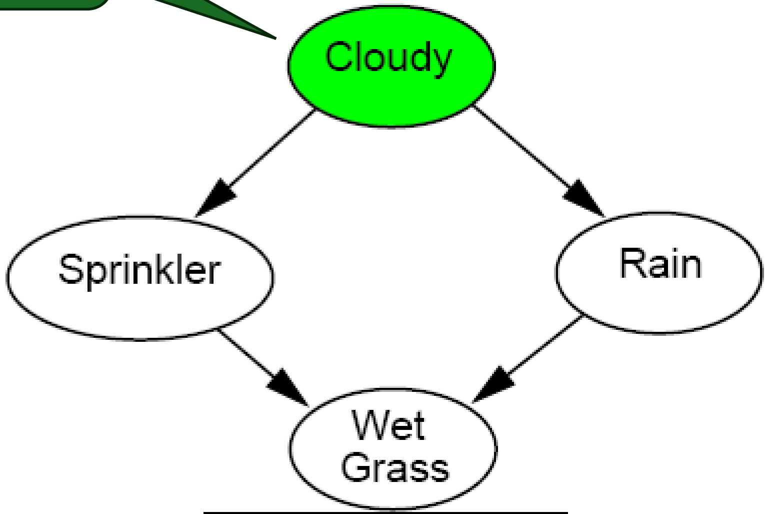
C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example: Sampling from a Bayesian Network

Draw a random value using the probability.



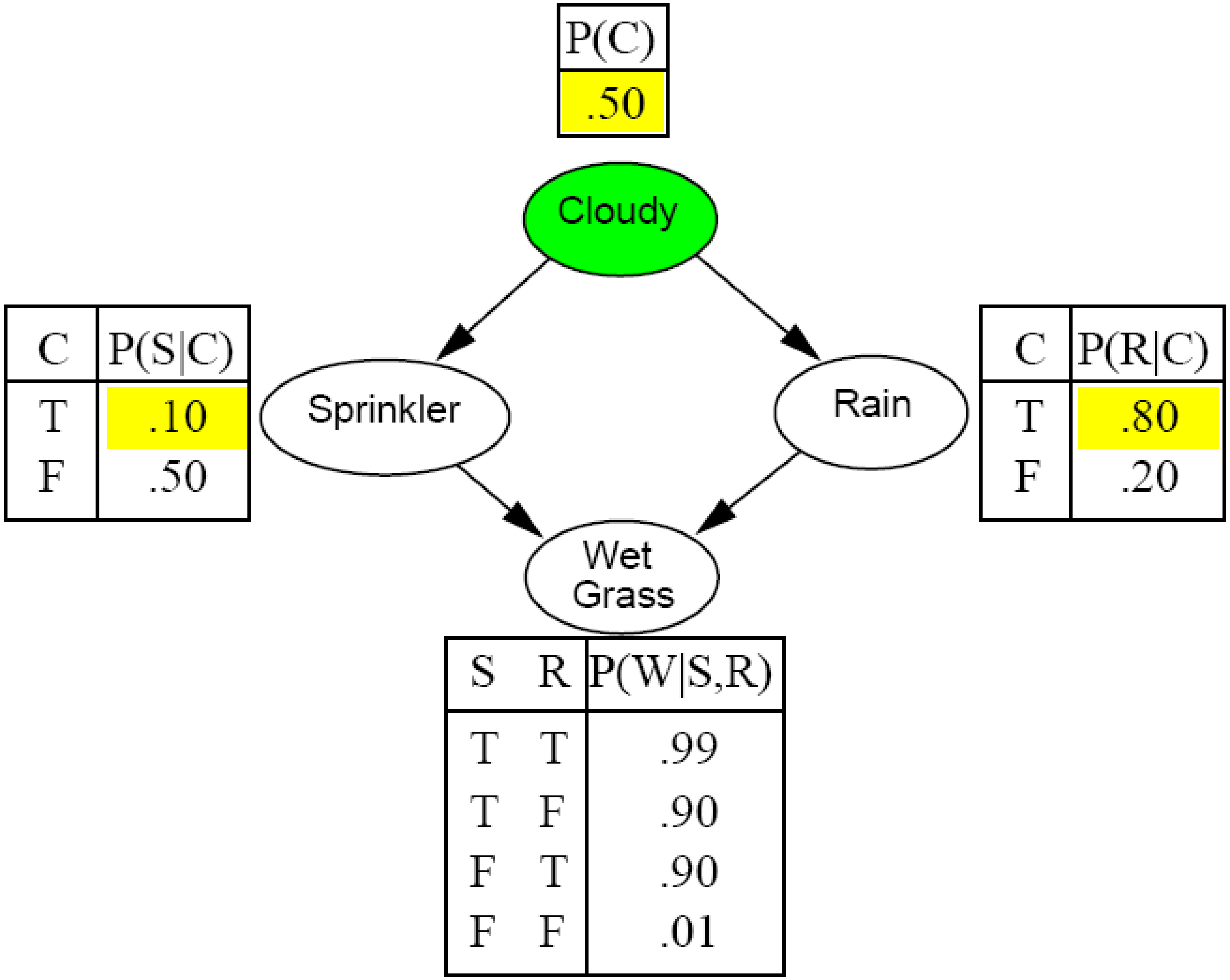
P(C)
.50

C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example: Sampling from a Bayesian Network



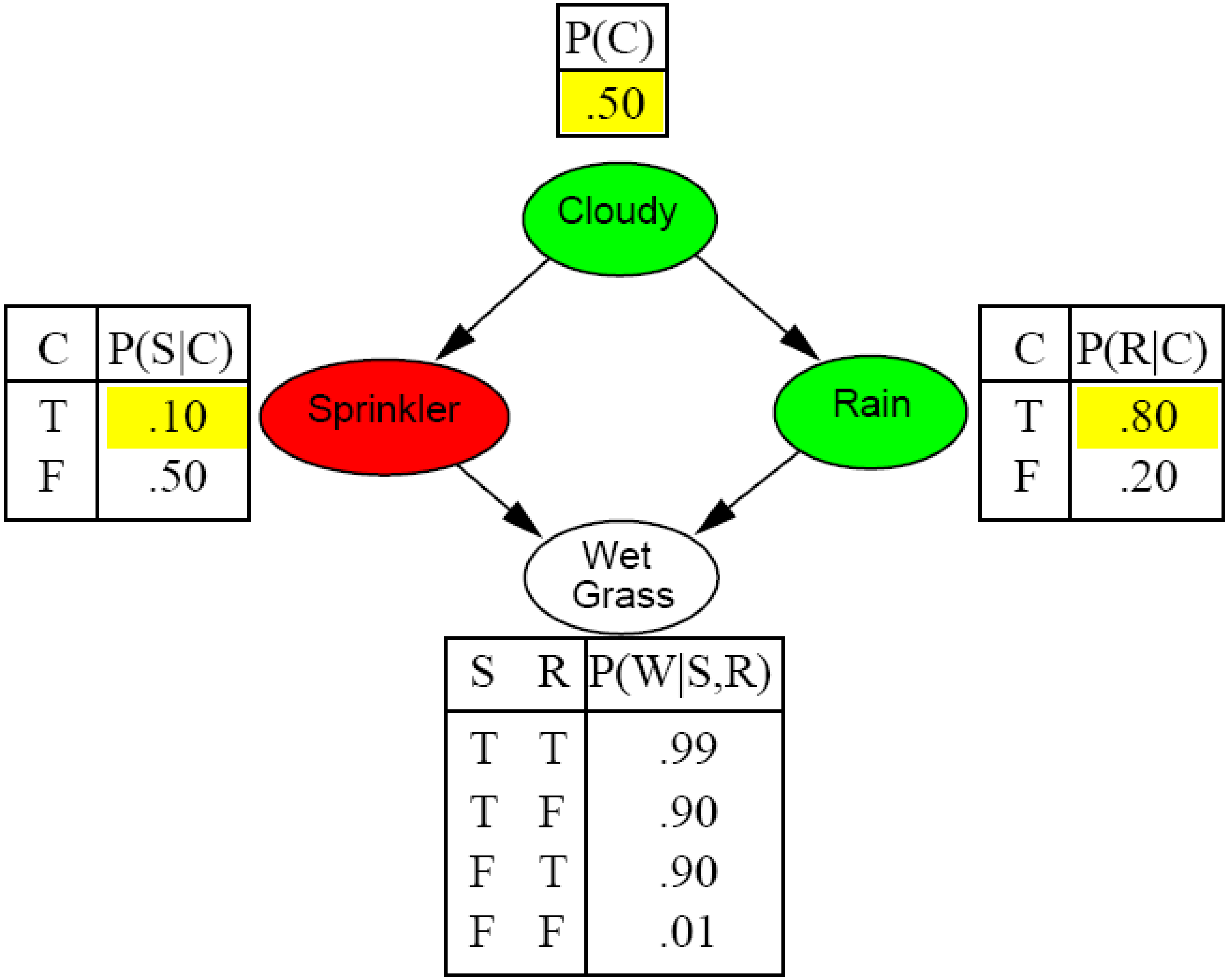
P(C)
.50

C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example: Sampling from a Bayesian Network



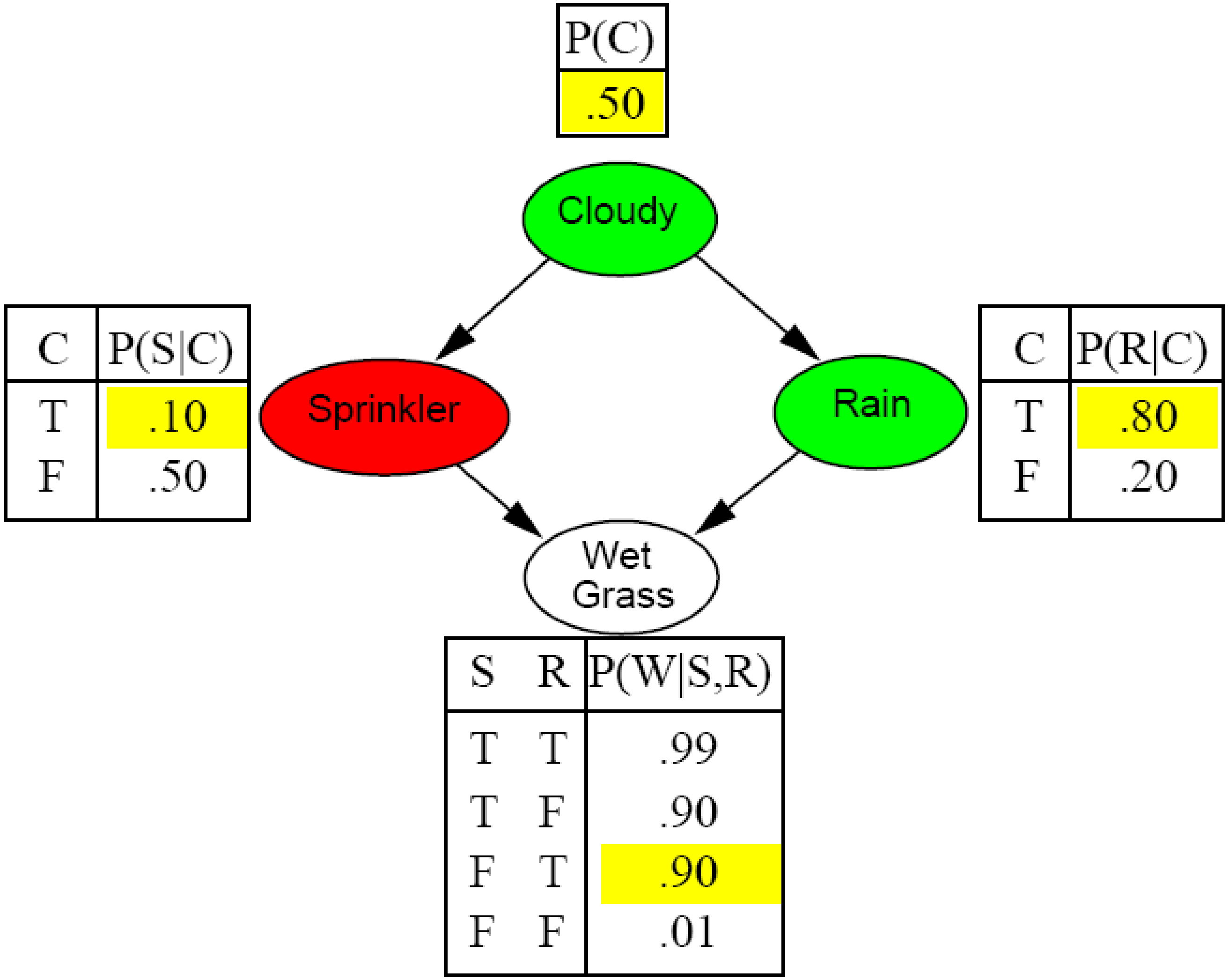
P(C)
.50

C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example: Sampling from a Bayesian Network



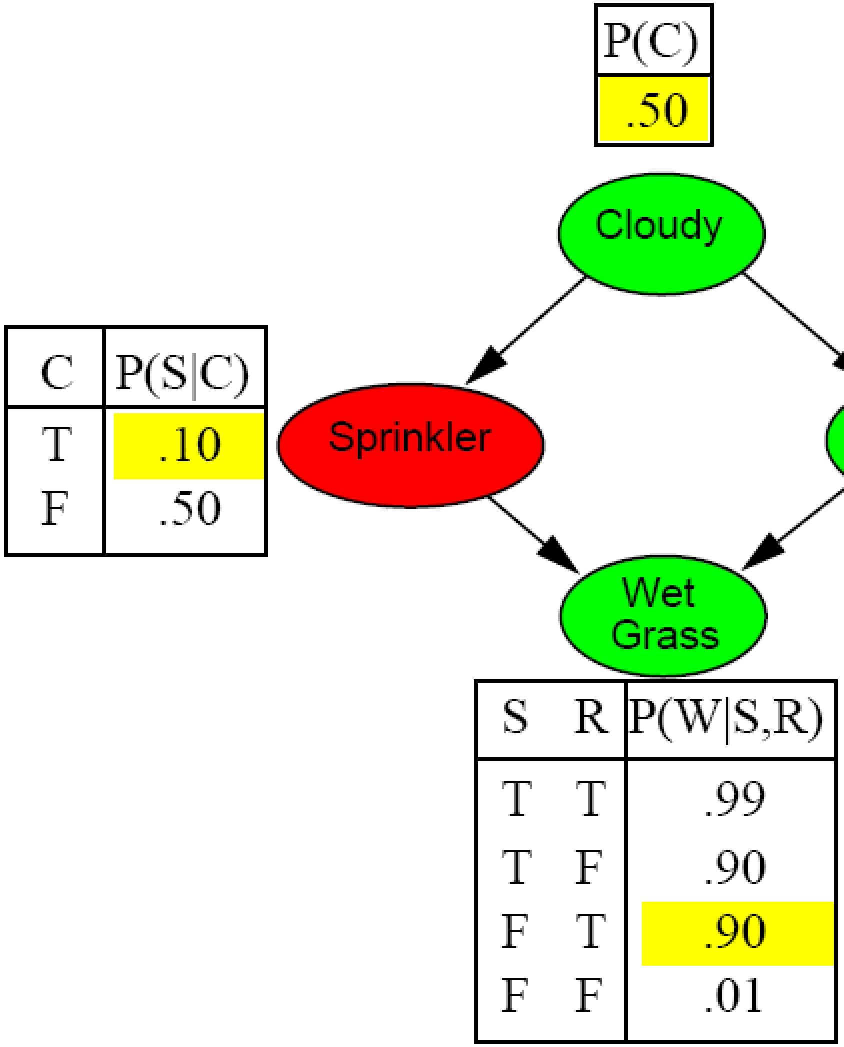
P(C)
.50

C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example: Sampling from a Bayesian Network



P(C)
.50

C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Prior Sample returns the event:

[C = True, S = False, R = True, W = True]

Estimating the Joint and Marginal Probability Distributions from Individual Samples

Joint Probability Distribution

Sample *N* times and determine $N_{PS}(x_1, x_2, \dots, x_n)$, the count of how many times Prior-Sample produces event $(x_1, x_2, \dots, x_n)$.

$$\hat{P}(x_1, x_2, \dots, x_n) = \frac{N_{PS}(x_1, x_2, \dots, x_n)}{N}$$

Marginal Probability Distributions

The marginal probability of a partially specified event (some x values are known) can also be calculated using the same samples. E.g.,

$$\hat{P}(x_1) = \frac{N_{PS}(x_1)}{N}$$

Estimating **Conditional Probability Distributions** from BNs

Estimate posterior probabilities for decision making

Estimating Conditional Probabilities: **Rejection Sampling**

To react to observed evidence, we want to estimate conditional probabilities. At this point, the **evidence** variables are no longer random but **fixed**.

Issue: Prior samples ignore the relationship to the fixed evidence.

Idea: Sample N times and **ignore the samples that are not consistent with the evidence e**.

$$\hat{\mathbf{P}}(X|\mathbf{e}) = \frac{\mathbf{N}_{PS}(X,\mathbf{e})}{N_{PS}(\mathbf{e})} = \alpha \mathbf{N}_{PS}(X,\mathbf{e})$$

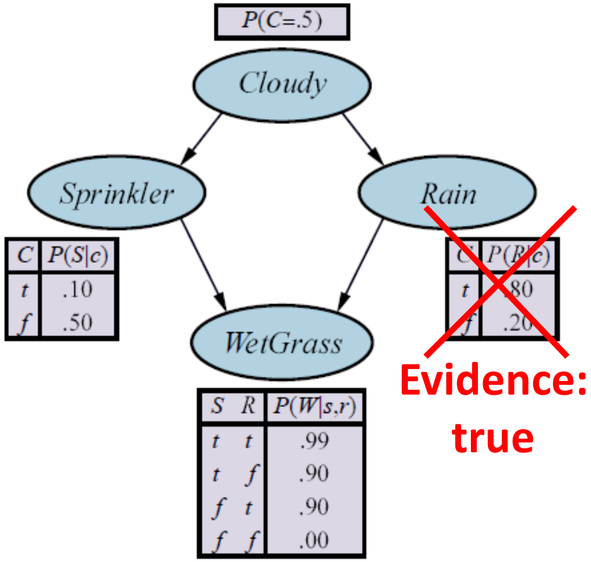
Normalization trick

Shortcoming:

- Throwing away samples is not very efficient.
- What if e is a rare event? Example: burglary ∧ earthquake
  Rejection sampling ends up throwing away most of the samples. This is extremely inefficient!

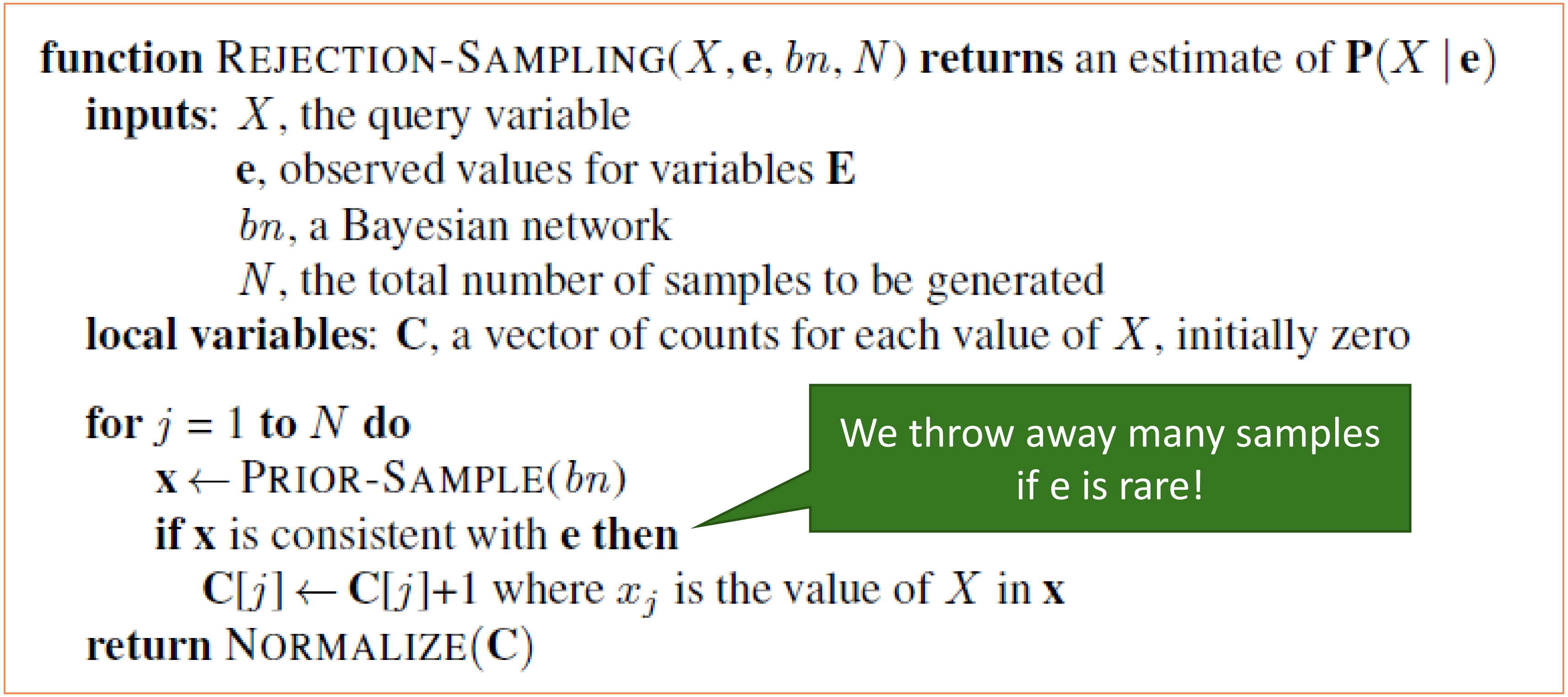
Example: We observe rain and wonder about

$$\boldsymbol{P}(WetGrass\ |Rain = true)$$



Evidence: true

Estimating Conditional Probabilities: **Rejection Sampling**



```
function REJECTION-SAMPLING($X$, $\mathbf{e}$, $bn$, $N$) returns an estimate of $\mathbf{P}(X \mid \mathbf{e})$
  inputs: $X$, the query variable
          $\mathbf{e}$, observed values for variables $\mathbf{E}$
          $bn$, a Bayesian network
          $N$, the total number of samples to be generated
  local variables: $\mathbf{C}$, a vector of counts for each value of $X$, initially zero

  for $j$ = 1 to $N$ do
      $\mathbf{x} \leftarrow$ PRIOR-SAMPLE($bn$)
      if $\mathbf{x}$ is consistent with $\mathbf{e}$ then
         $\mathbf{C}[j] \leftarrow \mathbf{C}[j]+1$ where $x_j$ is the value of $X$ in $\mathbf{x}$
  return NORMALIZE($\mathbf{C}$)
```

We throw away many samples if e is rare!

Estimating Conditional Probabilities: **Importance Sampling** (likelihood weighting)

Goal: Avoid throwing out samples like in rejection sampling.

1. Fix the evidence $\boldsymbol{E = e}$ for sampling and estimate the probability for the non-evidence variables using prior-sampling. We call this probability

$$Q_{WS}(x)$$

Note: Fixing the evidence breaks the dependence between the evidence variable and the evidence parents!

2. **Correct the probabilities** using weights

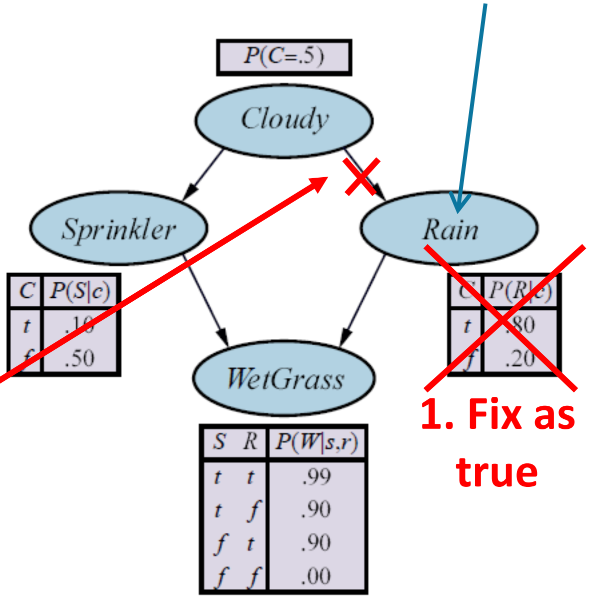
$$P(x|e) = w(x)Q_{WS}(x)$$

The right weight to fix the broken dependence is the chance that we see the evidence given its parent.

$$w(x) = \frac{1}{P(e)} \prod_{i=1}^{m} P(e_i|parents(E_i))$$

This gives samples with unlikely parents values a very low weight which is similar to rejecting them!

Example: Evidence = it rains

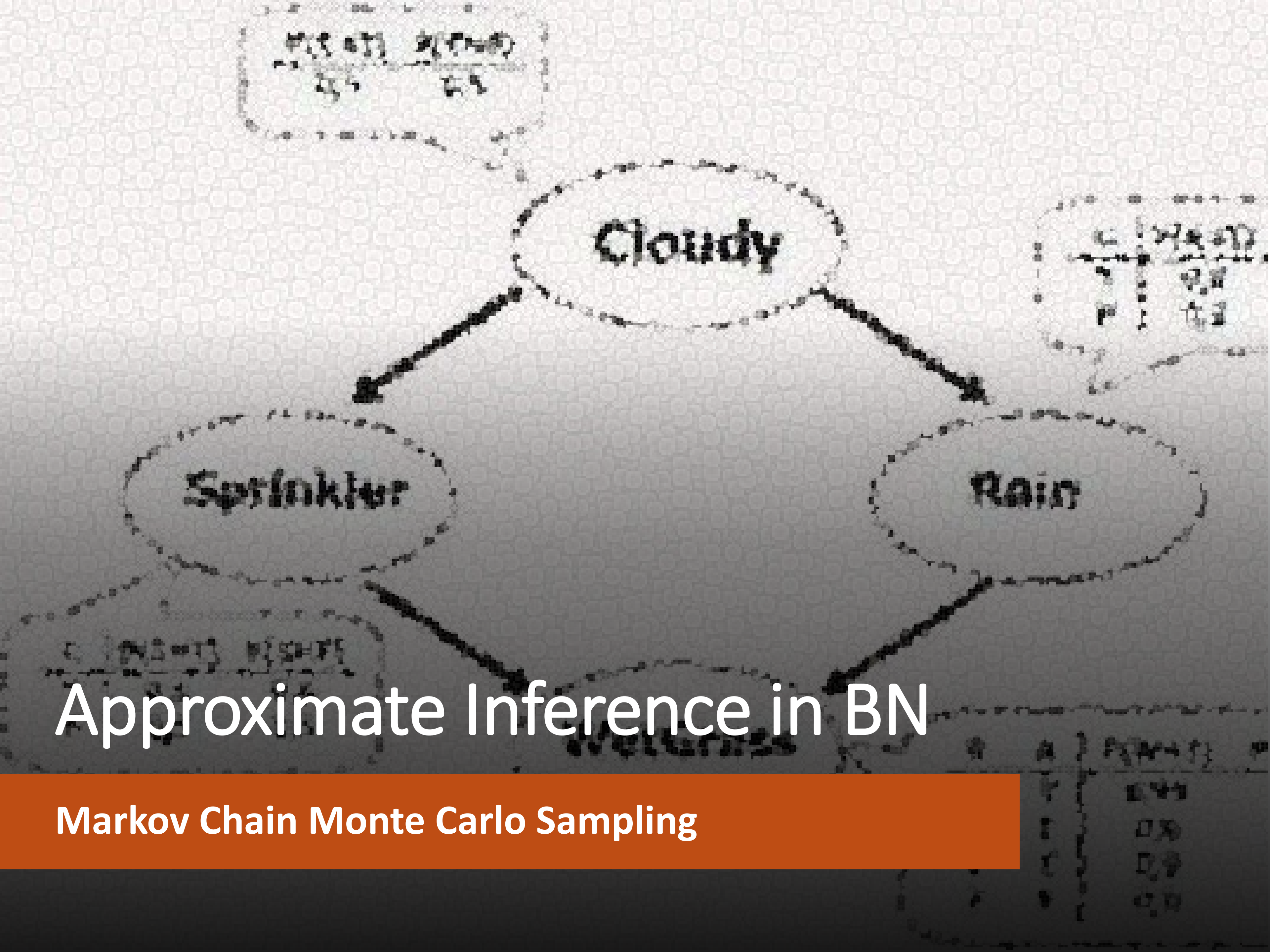


C	P(S|c)
t	.10
f	.50

C	P(R|c)
t	.80
f	.20

S	R	P(W|s,r)
t	t	.99
t	f	.90
f	t	.90
f	f	.00

1. Fix as true



Approximate Inference in BN

Markov Chain Monte Carlo Sampling

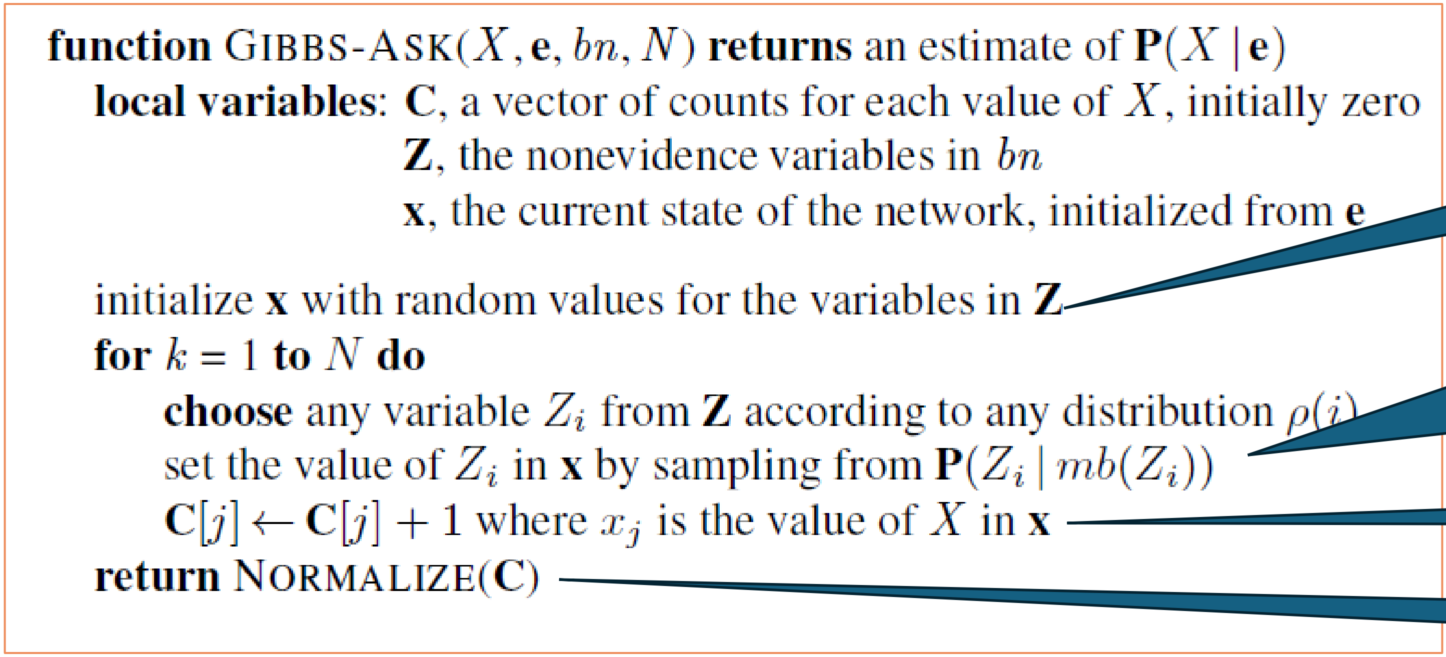
Estimating Conditional Probabilities:
Markov Chain Monte Carlo Sampling (MCMC)

- **Idea**: Instead of creating each sample individually from scratch, **generate a sequence of samples**.
- The next sample in the sequence is created by making random changes to the current sample. Changes are controlled by a **Markov Chain** (MC) that is specifically created to have the desired probability distribution as its stationary distribution.
- The stationary distribution of a MC can be estimated using **Monte Carlo** simulation by counting how often each state (=sample) is reached in a random walk through the MC.
- Algorithms:
  1. **Gibbs sampling** works well for BNs since it needs conditional probabilities and we have CPTs.
  2. **Metropolis-Hastings** sampling is more general.

Notes:

- MCMC with Gibbs sampling is the most popular inference method.
- Simulated annealing local search is related to MCMC algorithms.

Gibbs Sampling in Bayes Networks



```
function GIBBS-ASK($X$, $\mathbf{e}$, $bn$, $N$) returns an estimate of $\mathbf{P}(X \mid \mathbf{e})$
    local variables: $\mathbf{C}$, a vector of counts for each value of $X$, initially zero
                     $\mathbf{Z}$, the nonevidence variables in $bn$
                     $\mathbf{x}$, the current state of the network, initialized from $\mathbf{e}$

    initialize $\mathbf{x}$ with random values for the variables in $\mathbf{Z}$
    for $k$ = 1 to $N$ do
        choose any variable $Z_i$ from $\mathbf{Z}$ according to any distribution $\rho(i)$
        set the value of $Z_i$ in $\mathbf{x}$ by sampling from $\mathbf{P}(Z_i \mid mb(Z_i))$
        $\mathbf{C}[j] \leftarrow \mathbf{C}[j] + 1$ where $x_j$ is the value of $X$ in $\mathbf{x}$
    return NORMALIZE($\mathbf{C}$)
```

Start with a random state

Change one non-evidence variable at a time

Count

Convert to probabilities

$mb(Z_i)$ is the Markov blanket of random variable Z_i. The Markov blanket of a variable consists of all variables it can be dependent on:

- common cause: parents
- common effects: parents of children

$$P(z_i|mb(Z_i)) = \alpha P(z_i|parents(Z_i)) \prod_{Y_i \in children(X_i)} P(y_i|parents(Y_j))$$

It makes sure that the sampled value is consistent with all other values.

Gibbs Sampling: Example

Example: We observe sprinkler and wet grass. Find

$$P(Rain\ |Sprinkler\ = true, WetGrass\ = true).$$

The evidence is now fixed.

1. **Construct a Markov Chain** with states defined by all non-fixed variables. Transition probabilities for changing one variable are calculated from $P(z_i|mb(Z_i))$. Using the Markov blanket repairs all dependencies broken by fixing the evidence.

2. **Generate a sequence** by a random walk through this Markov chain and counting state visits. This will produce the stationary distribution over the states. For example, it is most likely that it is not cloudy and it does not rain.

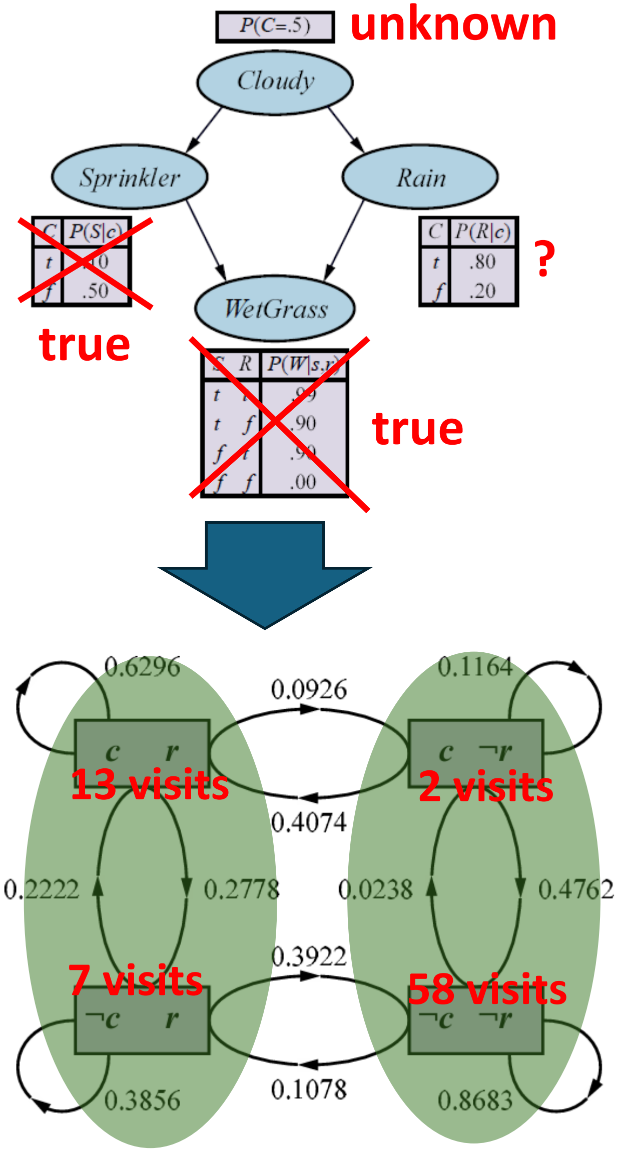
3. **Group by the query variable**. We observe 13+7=20 states with $Rain = true$ and 2+58=60 with $Rain = false$.

4. **Normalize counts** to produce a distribution:

$$NORMALIZE(\langle 13+7, 2+58\rangle) = \langle 0.25, 0.75\rangle$$

Estimate:

$$P(Rain\ |Sprinkler\ = true, WetGrass\ = true) \approx 0.25$$



Note the self-loops: the state stays the same when the resampled value is the same.

Conclusion

- Bayesian networks provide an efficient way to **store a complete probabilistic model for an AI problem** by exploiting (conditional) independence between variables.
- **Inference** means querying the model for a conditional probability given some evidence.
- Exact inference is difficult for all but tiny models.
- The state-of-the-art is to use **approximate inference** by Markov Chain Monte Carlo sampling from the model.
- Any **software libraries** provide general inference engines.