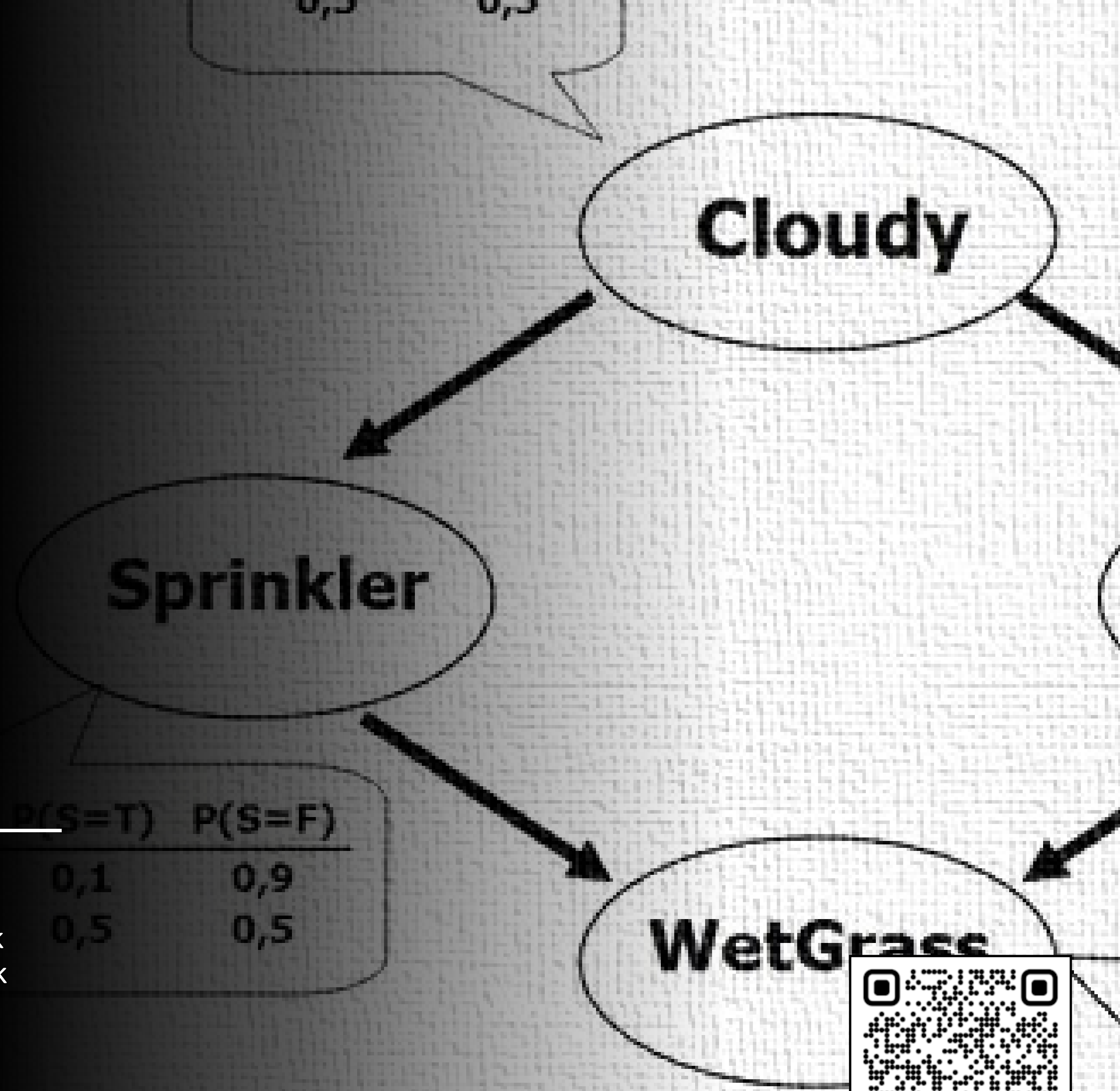
CS 5/7320 Artificial Intelligence

Probabilistic Reasoning

AIMA Chapter 13

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



Online Material

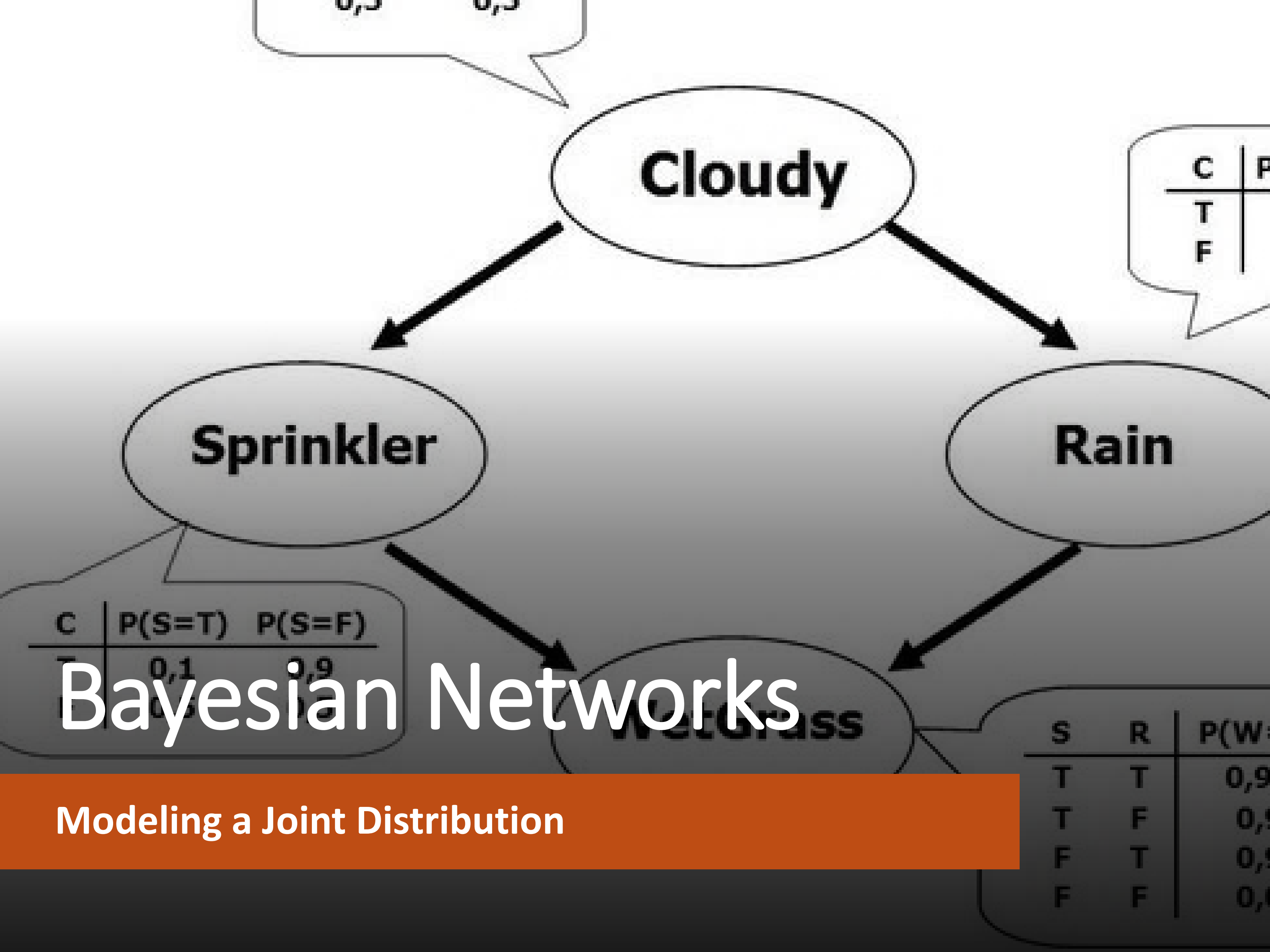
Probability Theory Recap

- Notation: Prob. of an event $P(X = x) = P(x)$
  Prob. distribution $\boldsymbol{P}(X) = \langle P(X = x_1), P(X = x_2), \dots, P(X = x_n) \rangle$

- Product rule $P(x, y) = P(x|y)P(y)$
- Chain rule $\boldsymbol{P}(\mathrm{X}_1, \mathrm{X}_2, \dots, \mathrm{X}_\mathrm{n}) = \mathbf{P}(\mathrm{X}_1)\mathbf{P}(\mathrm{X}_2|\mathrm{X}_1)\mathbf{P}(\mathrm{X}_3|\mathrm{X}_1, \mathrm{X}_2) \dots$
  $= \prod_{i=1}^{n} \boldsymbol{P}(X\_i | X_1, \dots, X_{i-1})$
- 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

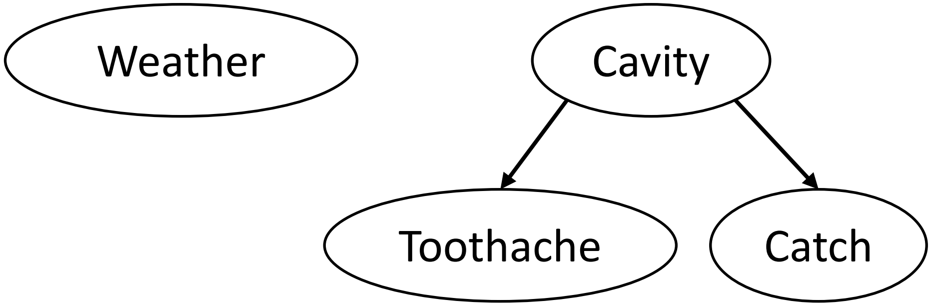
- Bayesian Networks to Specify Dependence
- Exact Inference
- Approximate Inference



Bayesian Networks

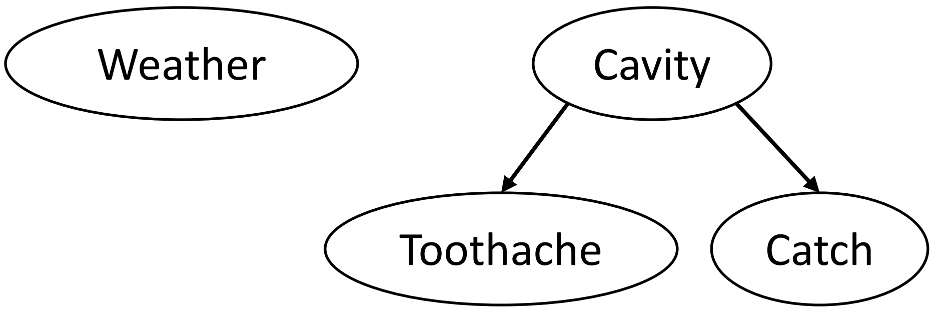
Modeling a Joint Distribution

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)

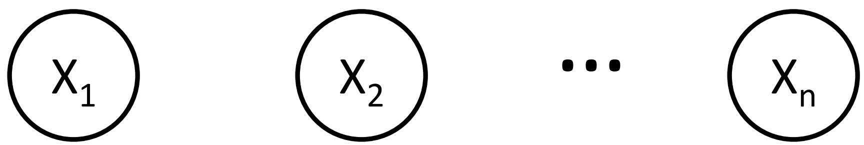
Arcs: Dependencies

- 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)

A network with all random variables assigned represents a **state of the system**.

Example: N independent coin flips

Complete independence: no interactions between coin flips



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

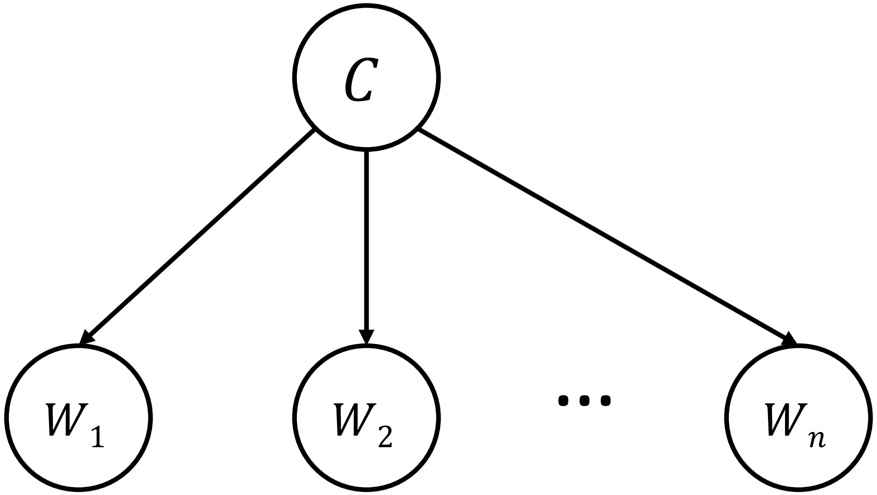
Joint probability distribution

Example: Naïve Bayes spam filter

Random variables:

- C: message class (spam or not spam)
- $W_1, \dots, W_n$: 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).



$$P(W_1, W_2, \dots, W_n | C) = P(W_1|C)P(W_2|C) \dots P(W_n|C)$$

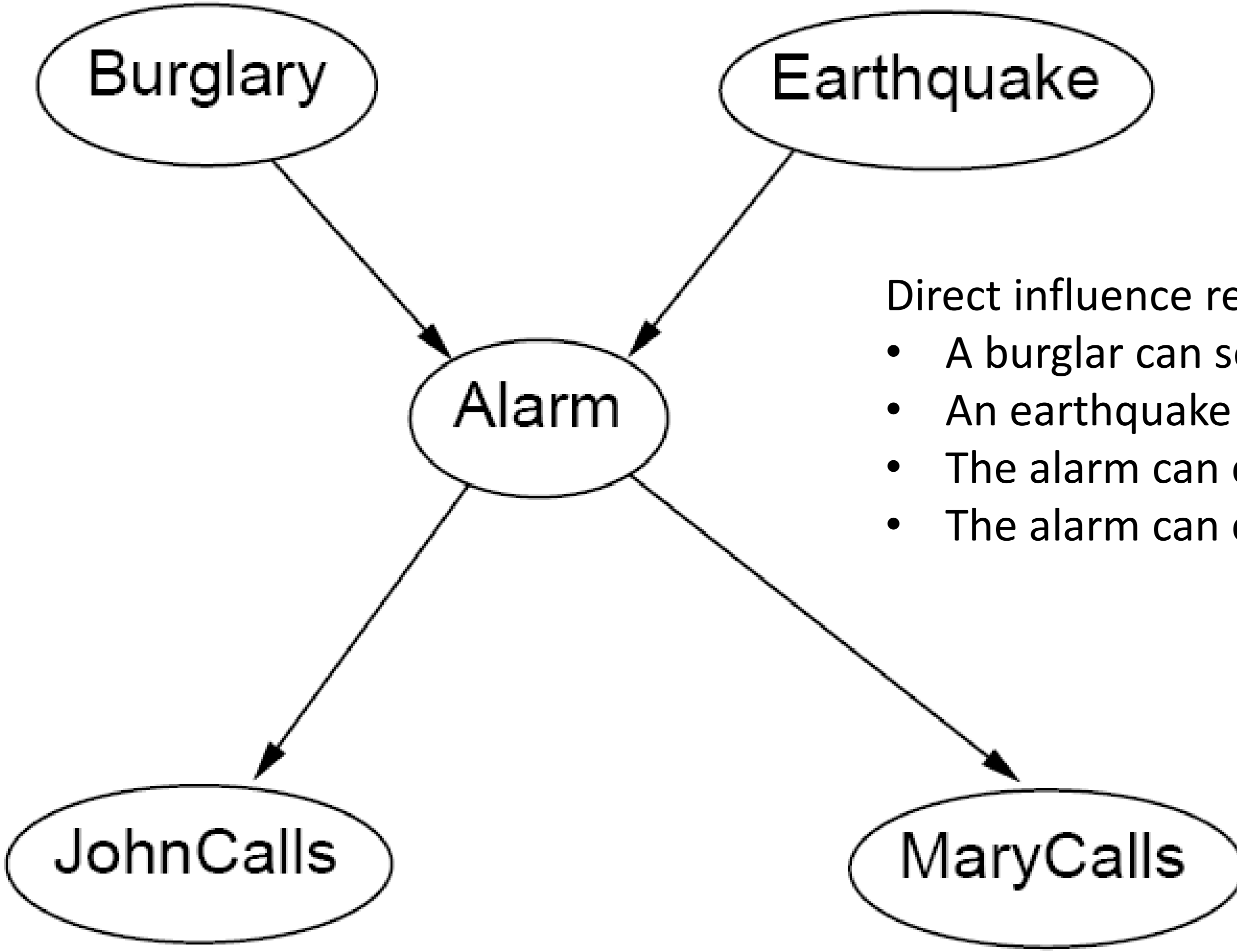
Example: Burglar Alarm

- **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

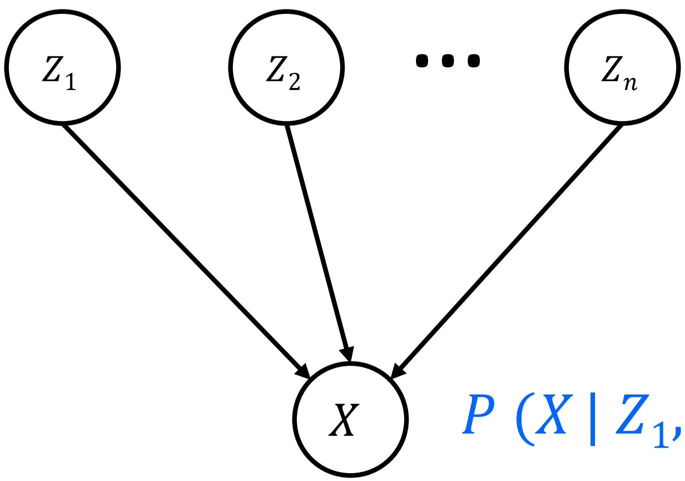
Parameters: Conditional probability tables

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

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

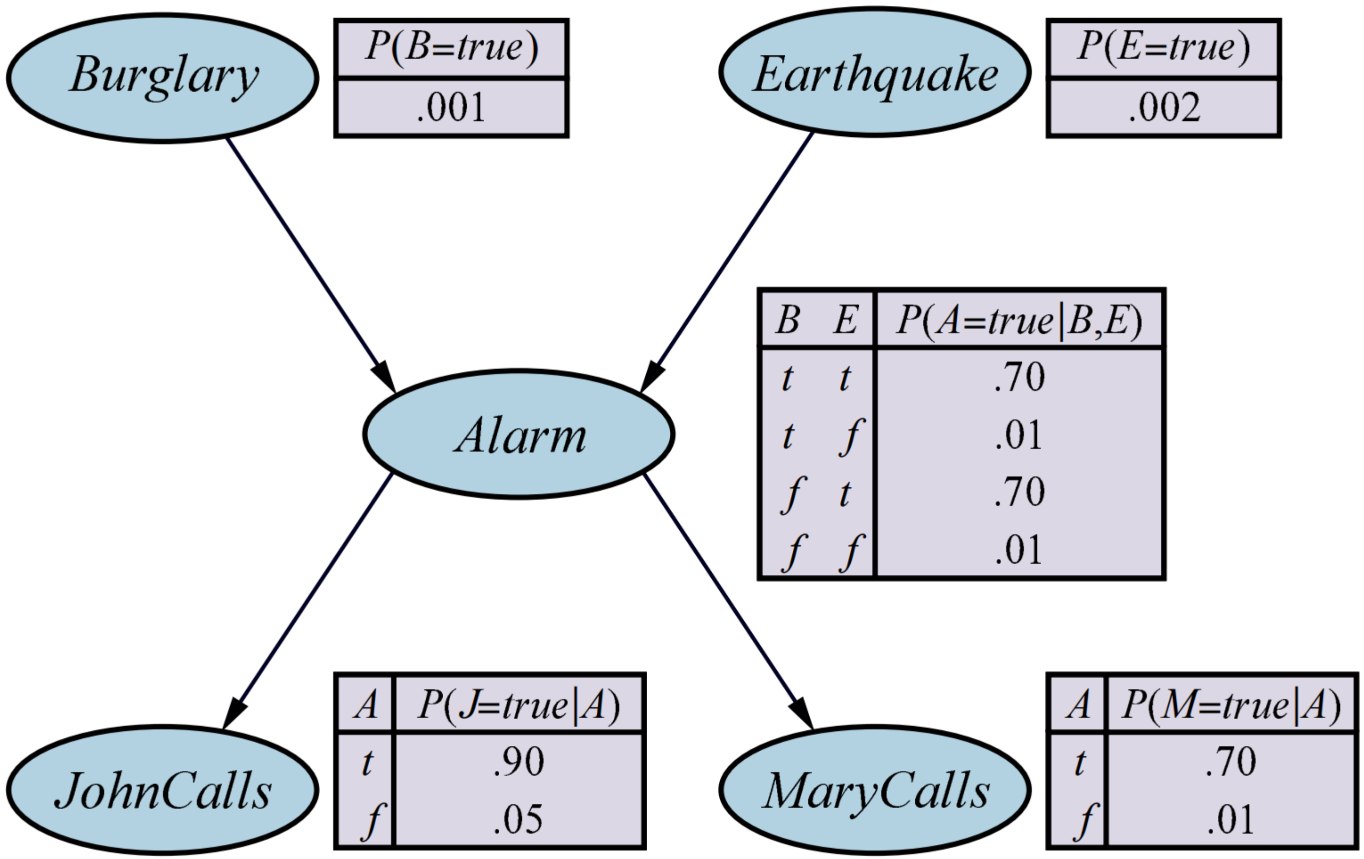
These distributions are stored in conditional probability tables (CPTs)

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*	.70
t	*f*	.01
f	*t*	.70
f	*f*	.01

2 parents

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

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

1 parent

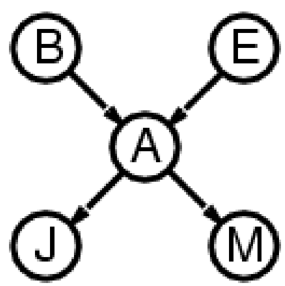
The Joint Probability Distribution

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

- Using chain rule, but only depends on parents:

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

- Example:



Construct following arrows

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

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!

Conditional Independence

- Example: *causal chain*

X: Low pressure

Y: Rain

Z: Traffic

- Is Z independent of X given Y?

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

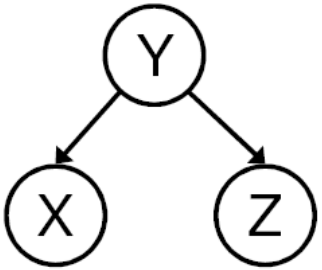
2. Bayes' rule: $= \frac{P(X)\frac{P(X|Y)P(Y)}{P(X)}P(Z|Y)}{P(Y)} = \mathrm{P(X|Y)P(Z|Y)}$

= Definition of conditional independence

X and Z are conditionally independent given Y

Conditional Independence cont.

- *Common cause*



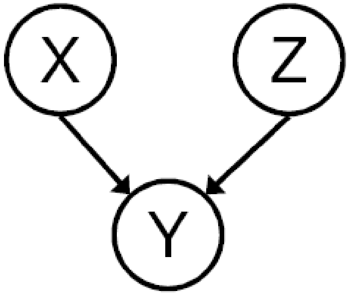
Y: Project due

X: Newsgroup busy

Z: Lab full

- Are X and Z independent?
  - No
- Are they conditionally independent given Y?
  - Yes

- *Common effect*



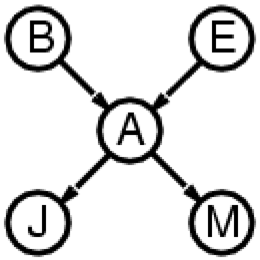
X: Raining

Z: Ballgame

Y: Traffic

- Are X and Z independent?
  - Yes
- Are they conditionally independent given Y?
  - No

Compactness



- For a network with n Boolean variables, the full joint distribution requires $O(2^n)$ probabilities.
  Example: Burglary network
  Complete joint probability specification $2^5 - 1 = 31$

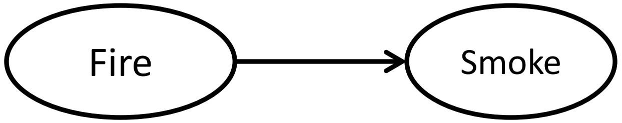
- If each variable X_i has at most k Boolean parents, then each conditional probability table (CPT) has at most 2^k rows.
- The CPTs for all n nodes contain then at most $O(n \times 2^k)$ probabilities.
- This reduces the complexity from exponential to linear in n and makes it vert compact!

- Example: Burglary network
  Using CPTs: $1 + 1 + 4 + 2 + 2 = 10$ probabilities

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 $X_1, \dots, X_{i-1}$ such that
     $P(X_i \mid Parents(X_i)) = P(X_i \mid X_1, \dots X_{i-1})$
     that is, add a connection only from nodes it directly depends on.

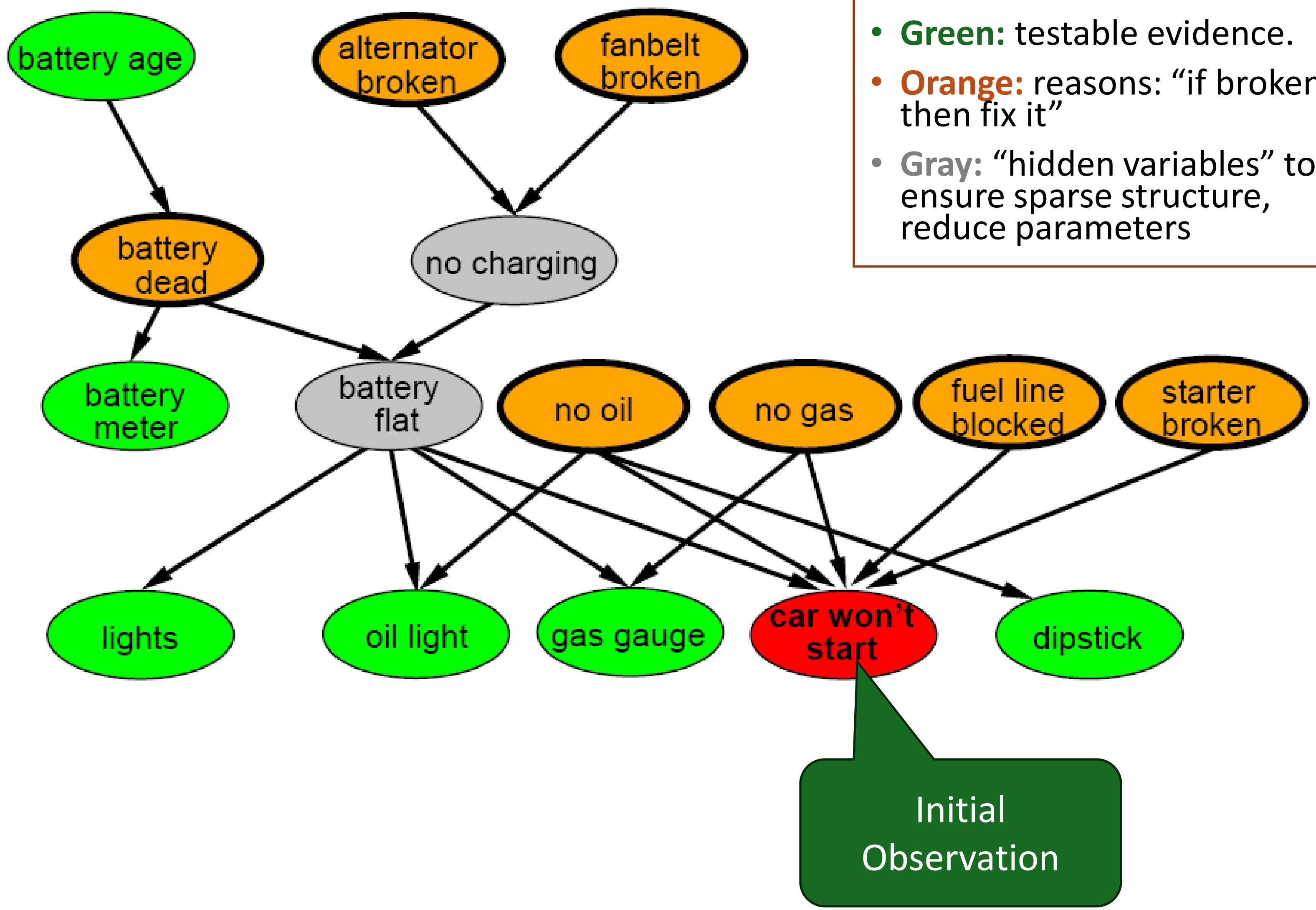
Note: There are many ways to order the variables. Networks are typically constructed by domain experts with causality in mind. E.g., Fire causes Smoke:



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

A more realistic Bayes Network: Car diagnosis

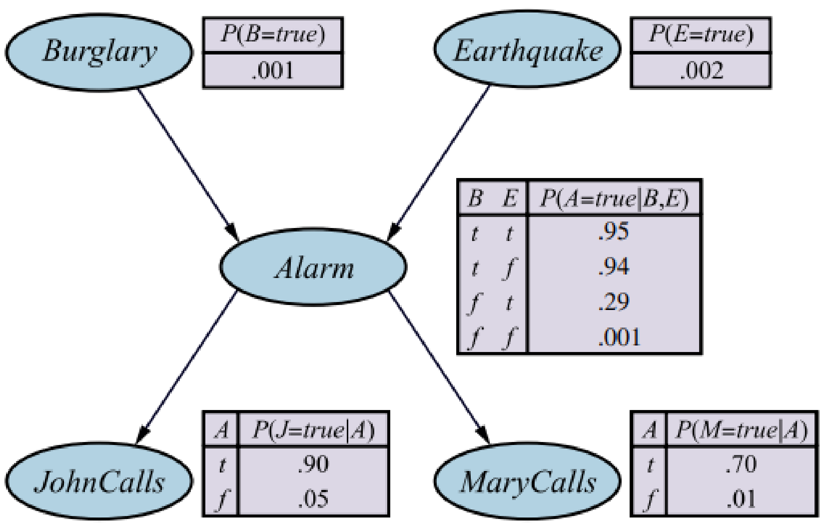
- **Initial observation**: car won't start.
- **Green:** testable evidence.
- **Orange:** reasons: "if broken, then fix it"
- **Gray:** "hidden variables" to ensure sparse structure, reduce parameters

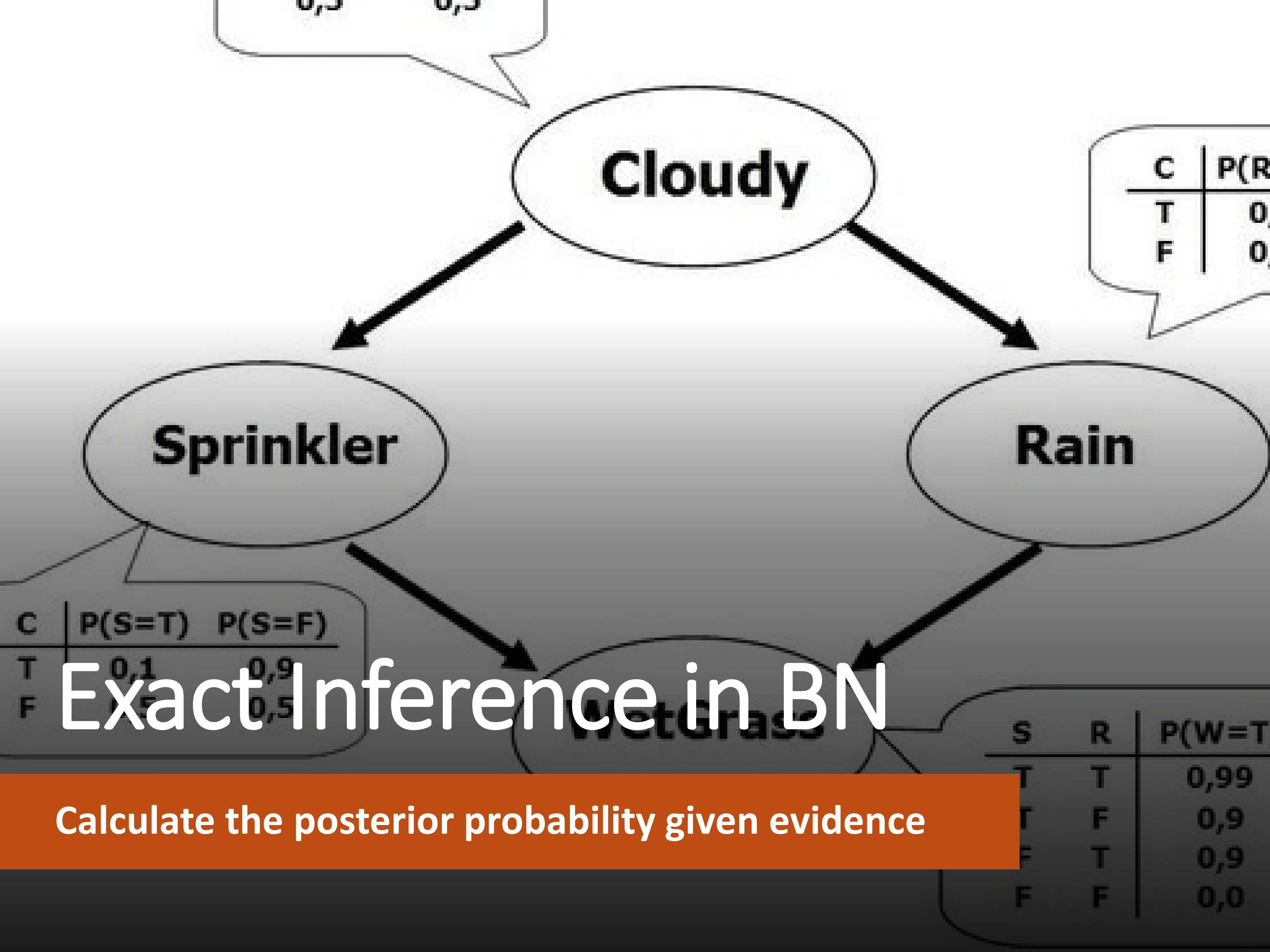


Summary

- Bayesian networks provide a **natural representation for joint probabilities** used to calculated conditional probabilities needed for inference (prediction).
- **Independence** and **conditional independence** (induced by causality) reduces the number of needed parameters and creates a compact network.
- Representation
  - Topology (nodes and edges)
  - Conditional probability tables
  - Typically easy for domain experts to construct

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





Exact Inference in BN

Calculate the posterior probability given evidence

Exact Inference

Goal

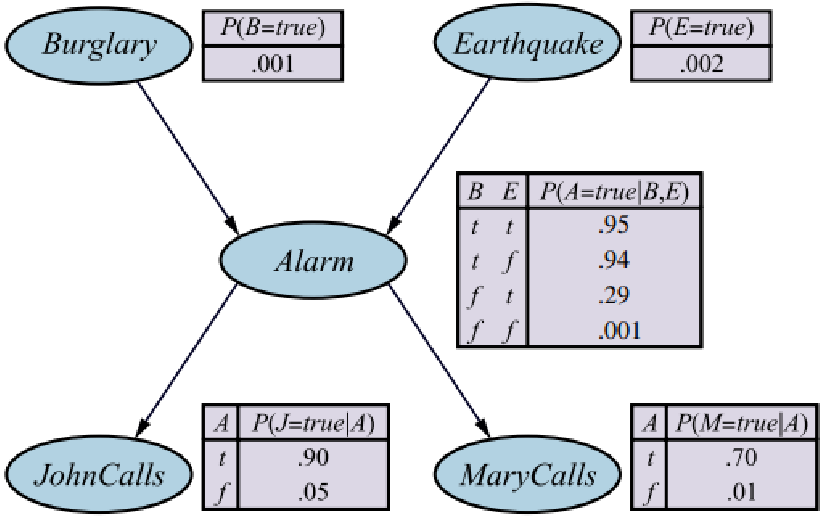
- Query *variables:* X
- *Evidence* (*observed*) variables: $E = e$
- *Set of unobserved* variables: Y
- Calculate the probability of X given e.

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



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

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

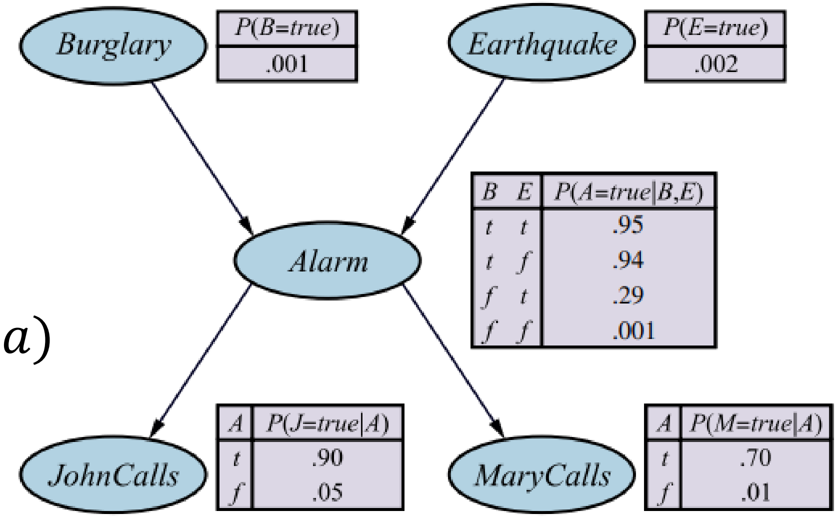
$$= \sum_e \sum_a P(b)P(e)P(a|b,e)P(j|a)P(m|a)$$

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

Full joint probability and marginalize over E and A

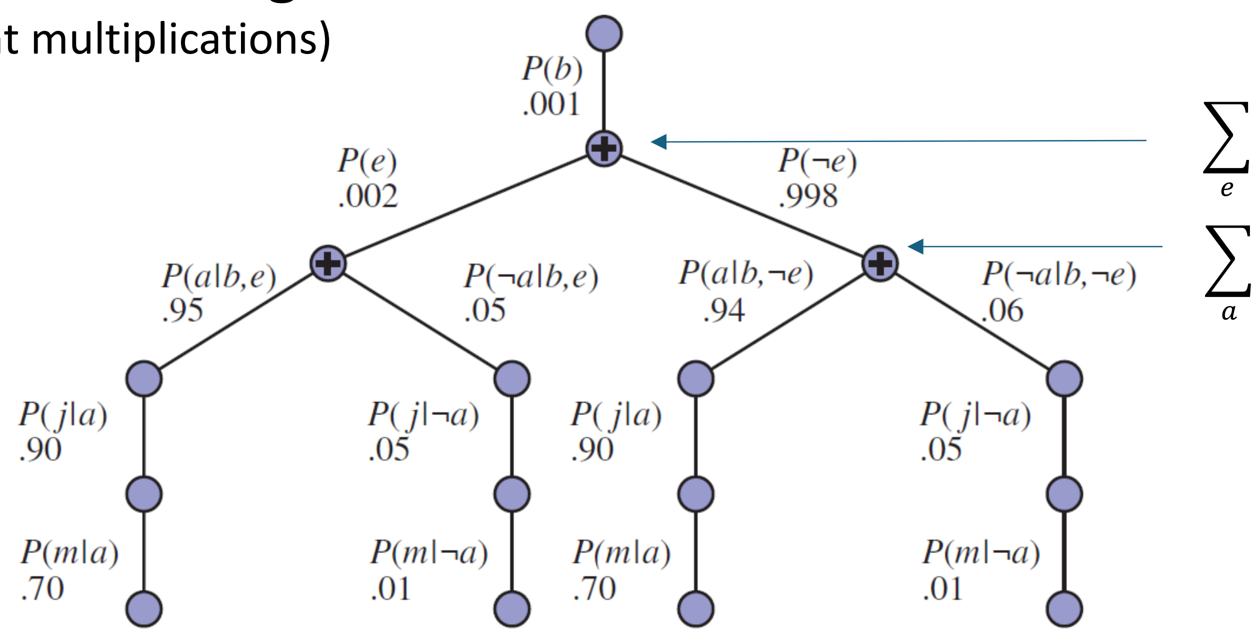
Exact inference: Example – Evaluation Tree

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



Implementation using an evaluation tree and CPTs

(lines represent multiplications)



Issues with Exact Inference in AI

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

Problems

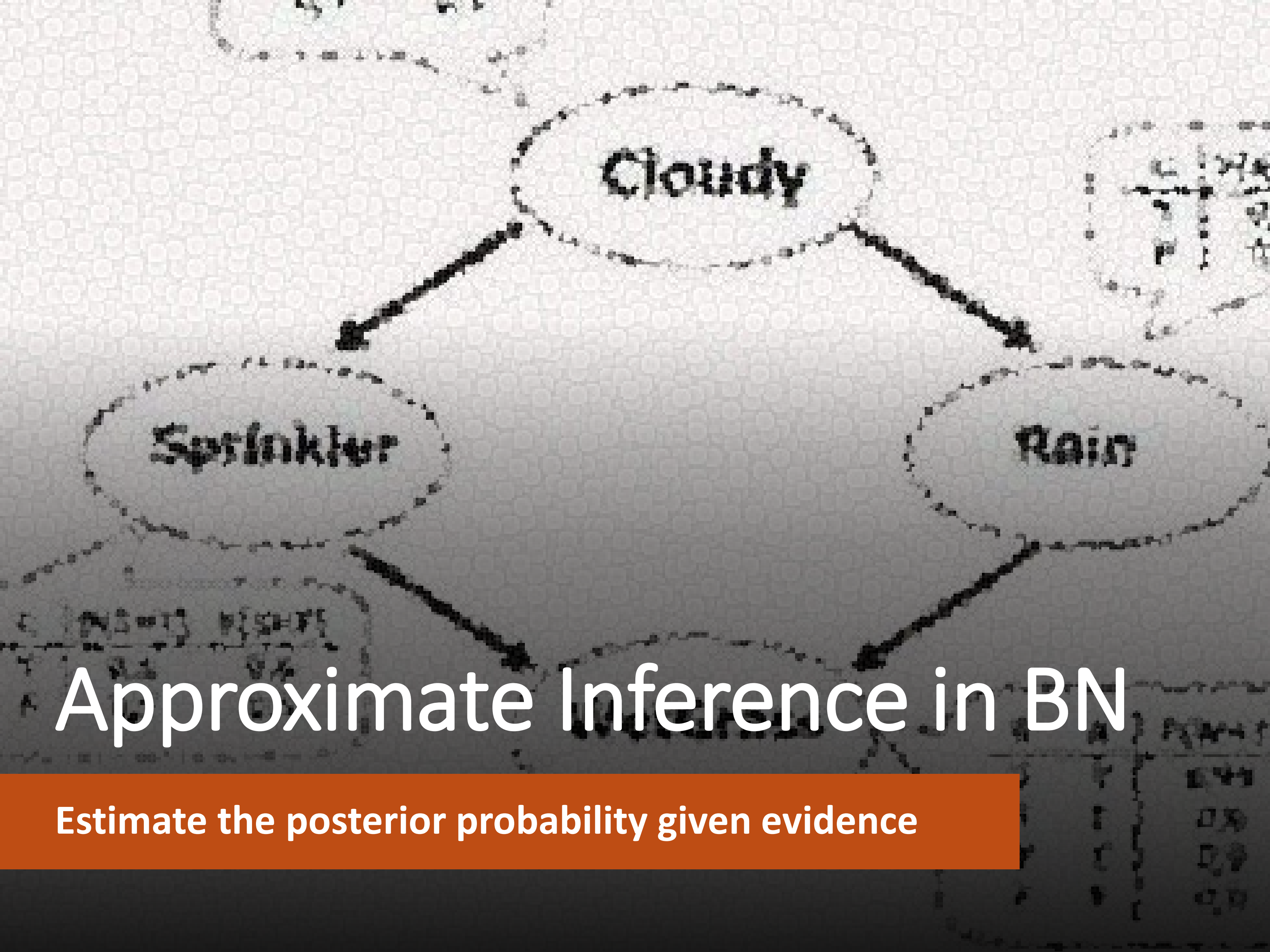
1. **Full joint distribution is too large** to store.

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

2. Marginalizing out many unobservable variables Y may involve **too many summation terms**.

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

In praxis, **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.

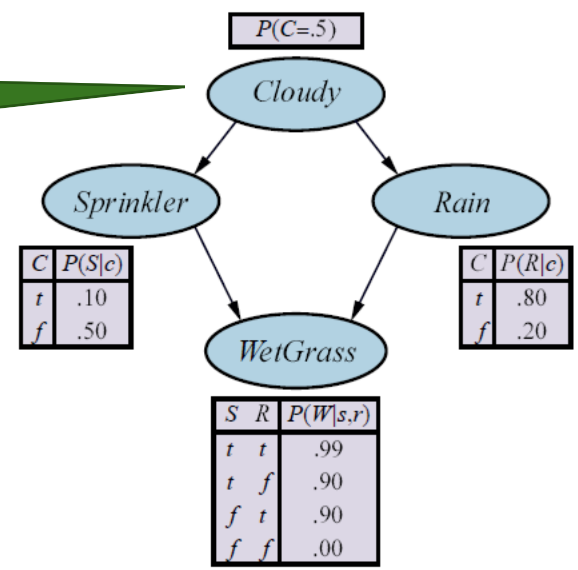
Idea: Generate samples from the network to estimate joint and conditional probability distributions using Monte Carlo methods.

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

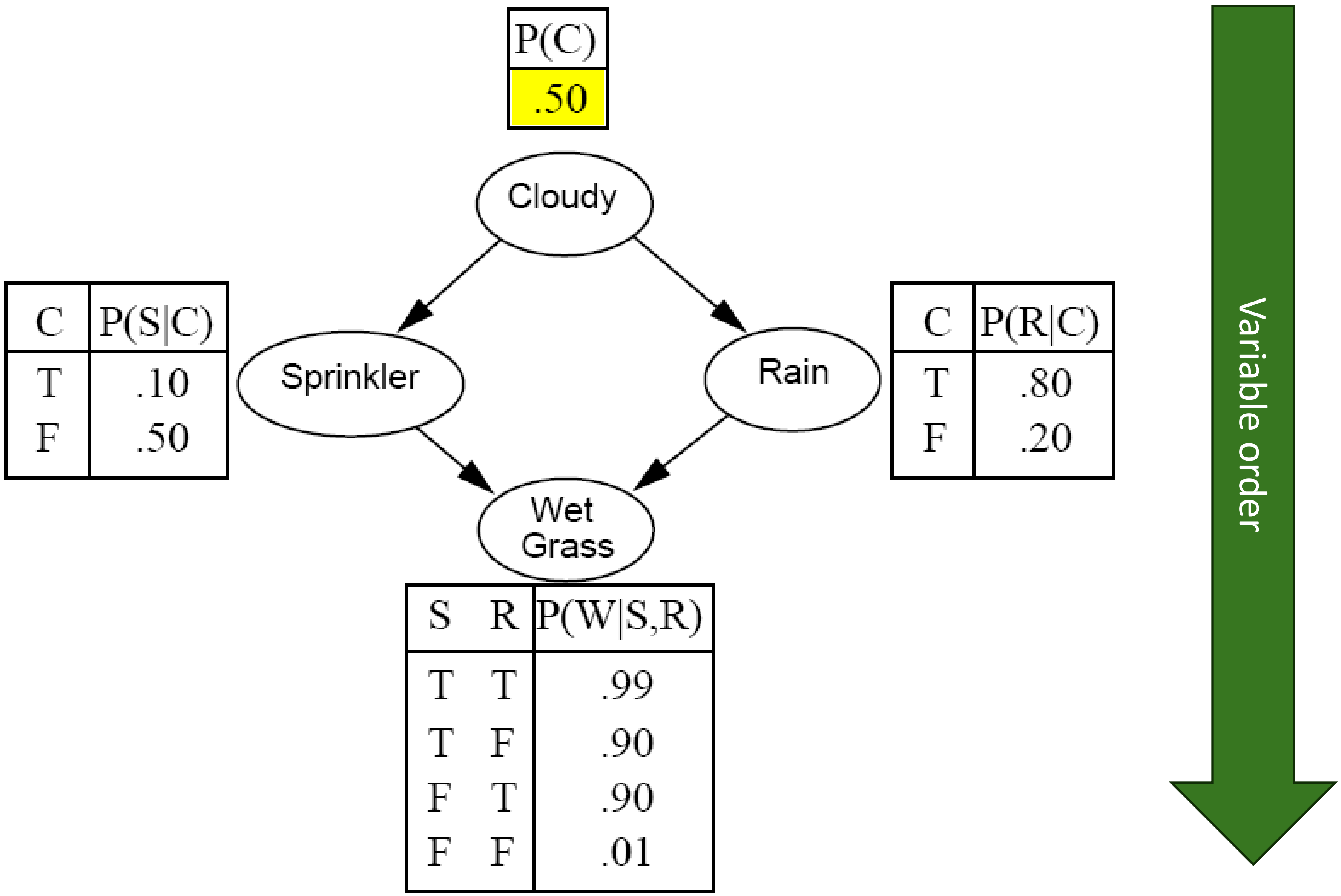
  $\mathbf{x} \leftarrow$ an event with n elements
  for each variable X_i **in** $X_1, \ldots, X_n$ **do**
    $\mathbf{x}[i] \leftarrow$ a random sample from $\mathbf{P}(X_i \mid parents(X_i))$
  return x

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

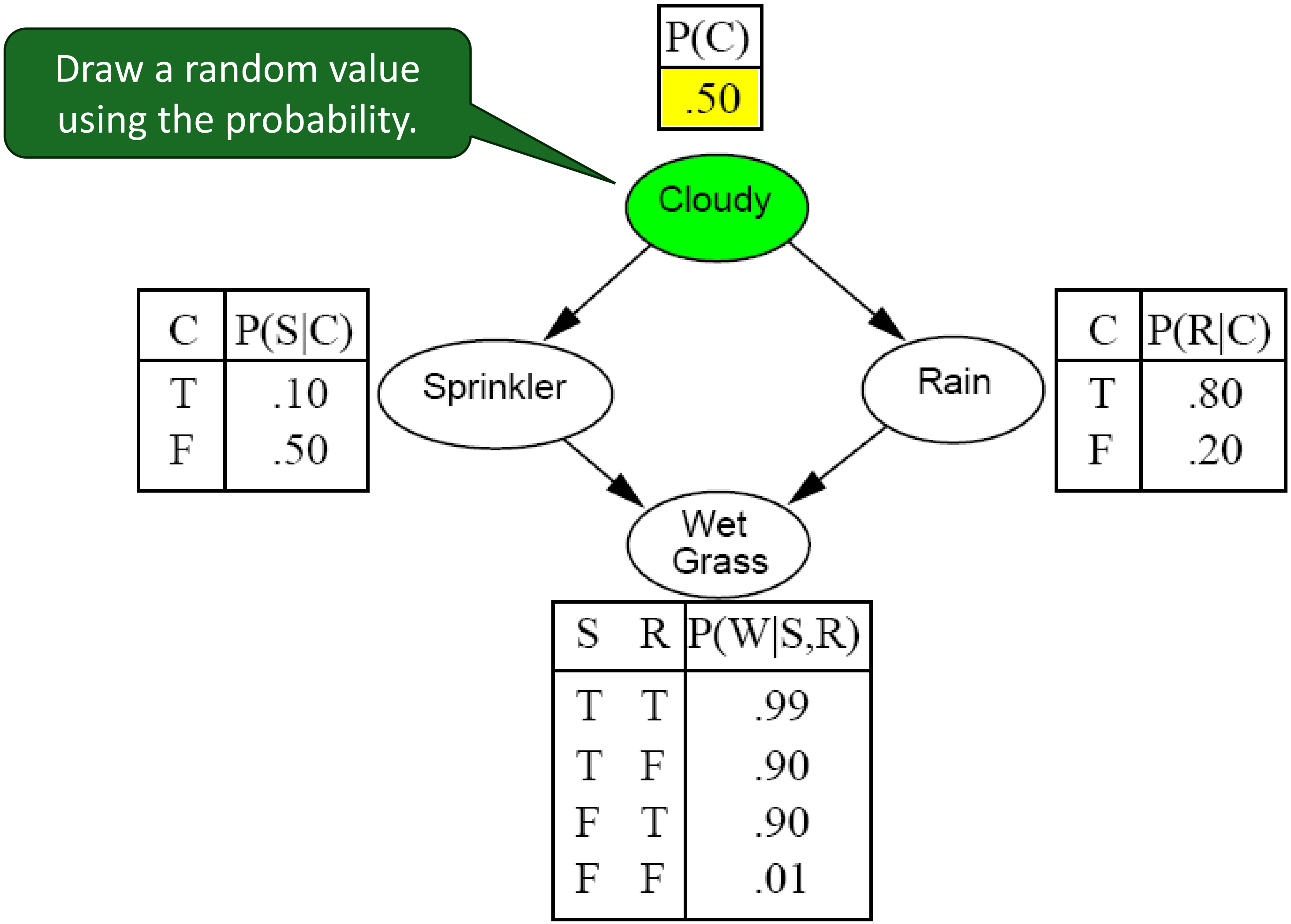


Variable Order

Example: Sampling from a Bayesian Network



Example: Sampling from a Bayesian Network



P(C)
.50

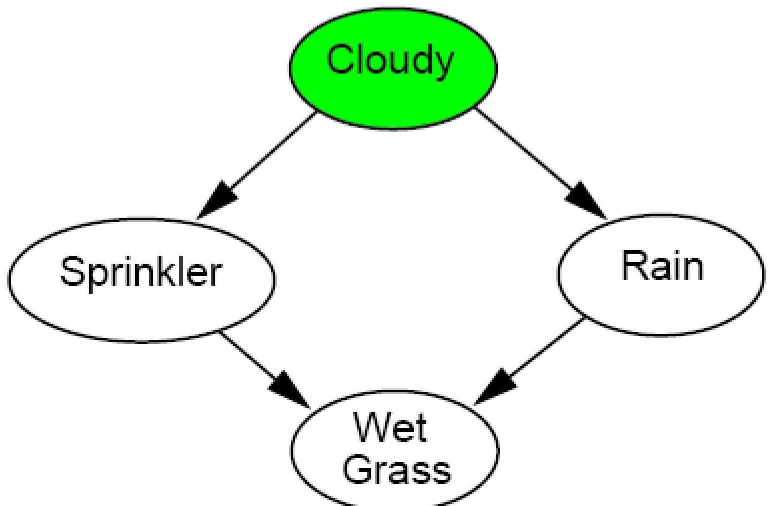
C	P(S|C)
T	.10
F	.50

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

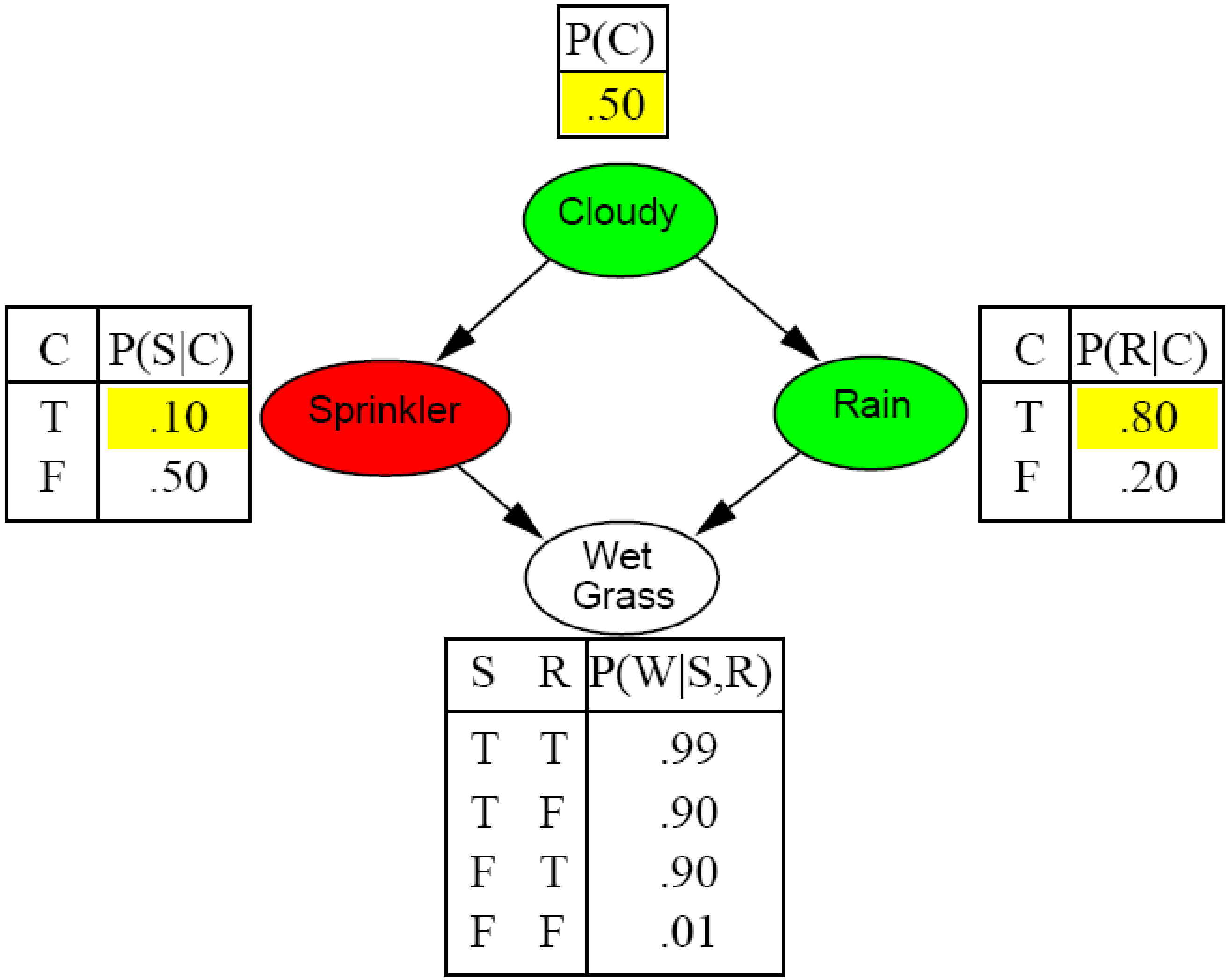


C	P(S|C)
T	.10
F	.50

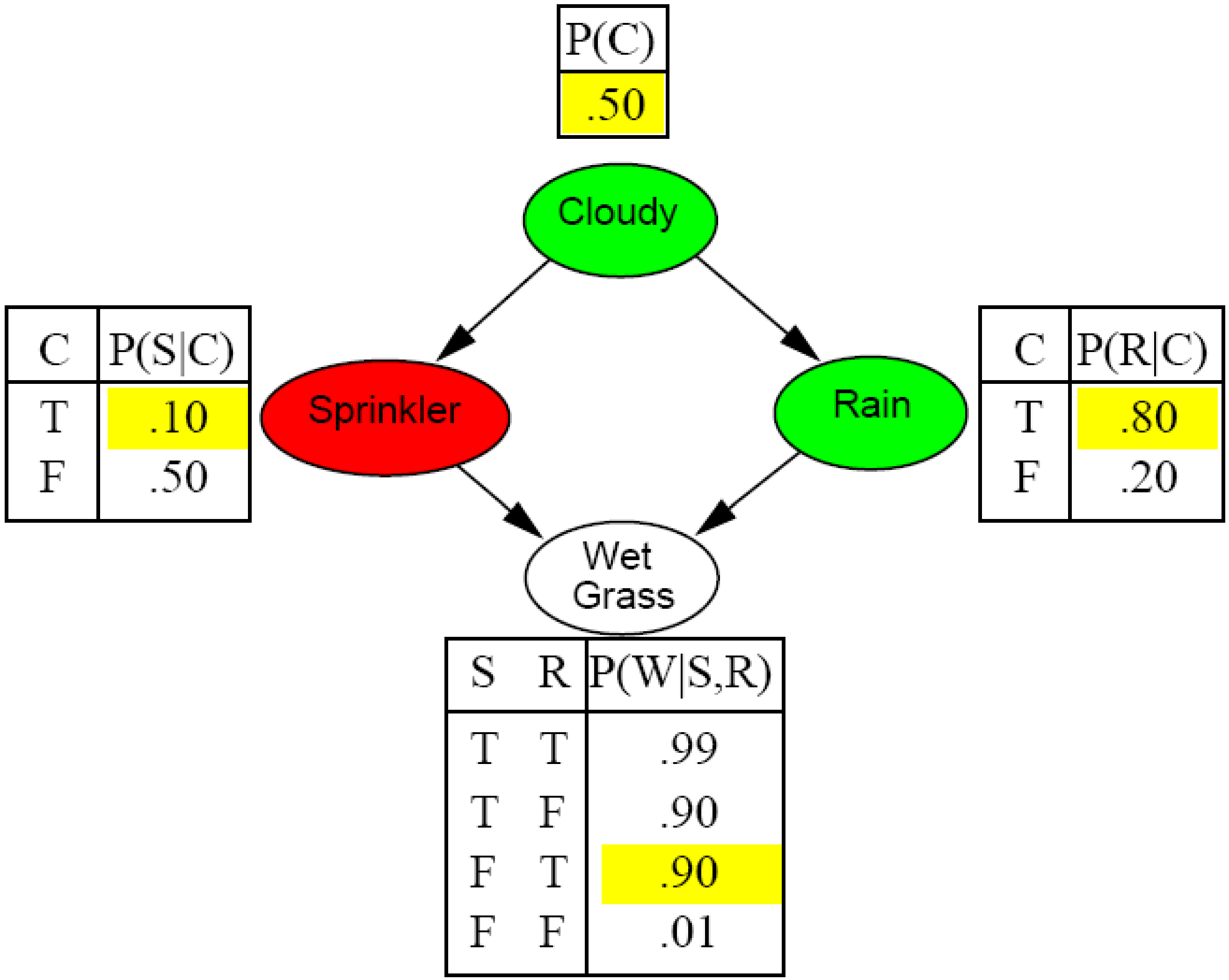
C	P(R|C)
T	.80
F	.20

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

Example: Sampling from a Bayesian Network

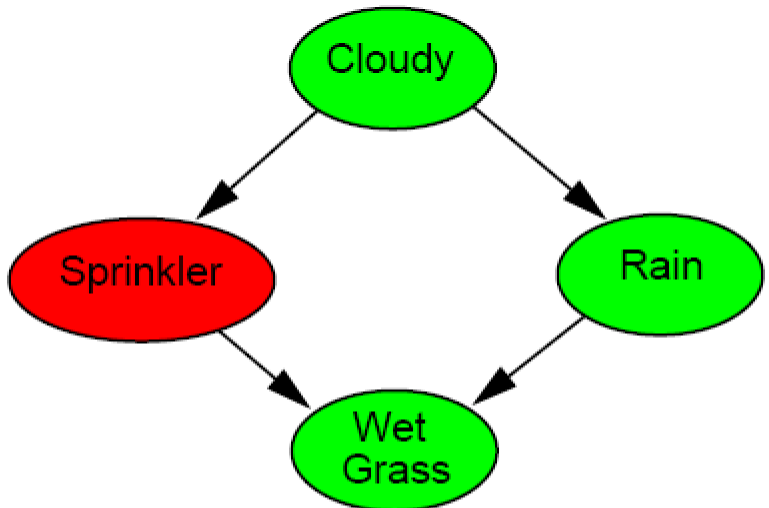


Example: Sampling from a Bayesian Network



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 Probability Distribution from Individual Samples

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

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

The marginal probability of 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 Probabilities: **Rejection Sampling**

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

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

Issue: What if e is a rare event?

- Example: burglary $\wedge$ earthquake
- Rejection sampling ends up throwing away most of the samples. This is very inefficient!

Estimating Conditional Probabilities: **Rejection Sampling**

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

  for j = 1 **to** N **do**
    x ← PRIOR-SAMPLE(bn)
    if x is consistent with **e then**
      C[j] ← **C**[j]+1 where x_j is the value of X in **x**
  return NORMALIZE(**C**)

We throw away many samples if e is rare!

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

Goal: Avoid throw out samples like in rejection sampling.

1. Fix the evidence $E = e$ 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 of the evidence variable on the evidence parents!

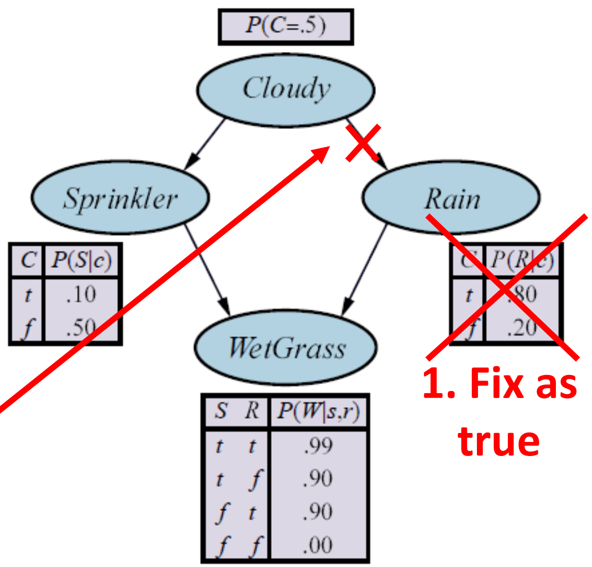
2. Correct the probabilities using weights

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

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

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

Example: Evidence = it rains



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

- **Idea**: Instead of creating each sample individually from scratch, **generates a sequence of samples**.
- Create the next state (= sample) by making random changes to the current state (= modify non-evidence variables). The sequence of states forms a random process called a **Markov Chain** (MC).
- The MC's stationary distribution turns out to be the posterior distribution of the non-evidence variables.
- The stationary distribution of a MC can be estimated using **Monte Carlo** simulation by counting how often each state is reached and normalize to obtain probability estimates.
- Algorithms:
  1. Gibbs sampling works well for BNs since we have CPTs.
  2. Metropolis-Hastings sampling is more general.

Note: Simulated annealing local search belongs to the family of 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

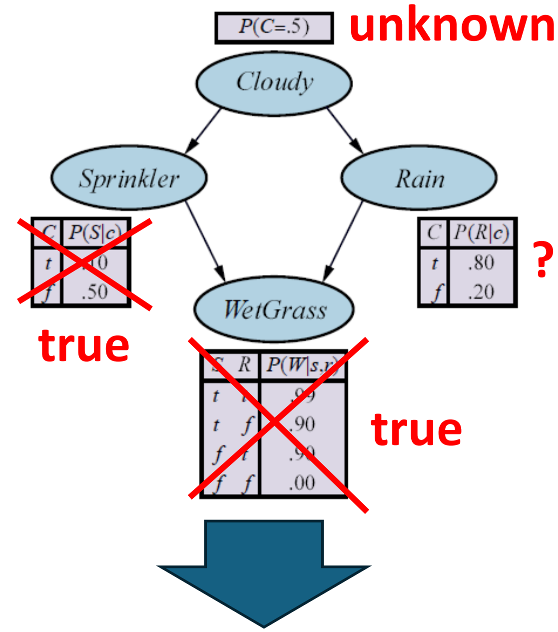
Count

- $mb(Z_i)$ is the Markov blanket of random variable Z_i. It makes sure that the new value is consistent with the other values. The Markov blanket of a variable consists of all variables it can be dependent of (parents, children and parents of children).

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

Gibbs Sampling: Example

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



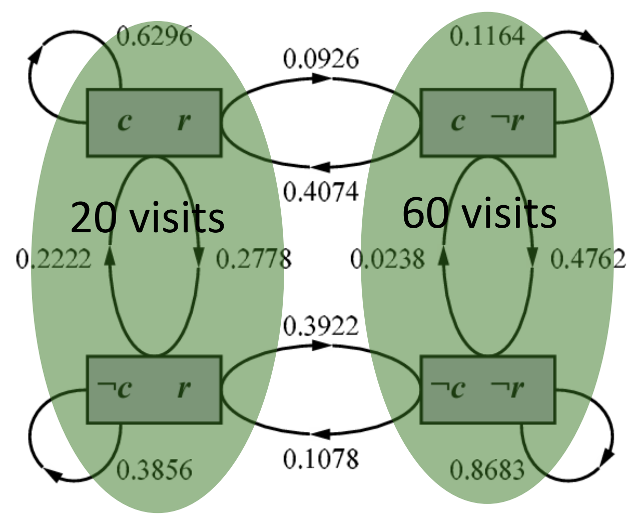
Determine states and calculate transition probabilities of the Markov chain for changing one variable using $P(z_i|mb(Z_i))$.

The algorithm randomly wanders around in this graph using the stated transition probabilities.

Assume that we observe 20 states with $Rain = true$ and 60 with $rain = false$:

$$NORMALIZE(\langle 20{,}60\rangle) = \langle 0.25{,}0.75\rangle$$

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



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

Conclusion

- Bayesian networks provide an efficient way to store a complete probabilistic model 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.
- State of the art is to use approximate inference by sampling from the model.
- Software libraries provide general inference engines.