CS:4420 Artificial Intelligence

Spring 2017

Probabilistic Reasoning

Cesare Tinelli

The University of Iowa

[a] These notes were originally developed by Stuart Russell and are used with permission. They are copyrighted material and may not be used in other course settings outside of the University of Iowa in their current or modified form without the express written consent of the copyright holders.

Readings

- Chap. 14 of [Russell and Norvig, 2012]

Making Probabilistic Reasoning Feasible

Recall:

- A joint probability distribution (JPD) contains all the relevant information to reason about the various kinds of probabilities of a set $\{X_1, \ldots, X_n\}$ of random variables.
- Unfortunately, JPD tables are difficult to create and also very expensive to store.
- One possibility is to work with conditional probabilities and exploit the fact that many random variables are conditionally independent.
- Belief Networks are a successful example of probabilistic systems that exploit conditional independence to reason *effectively* under uncertainty.

Review of Basic Concepts

The JPD is a collection of probabilities:

$$\mathbf{P}(X_1, \ldots, X_n) = \{P(X_1 = x_1 \wedge \cdots \wedge X_n = x_n) \mid x_i \in Domain(X_i)\}$$

Conditional Probability:

$$P(X_1 = x_1 \mid X_2 = x_2) = \frac{P(X_1 = x_1 \wedge X_2 = x_2)}{P(X_2 = x_2)}$$

or

$$P(X_1 = x_1 \wedge X_2 = x_2) = P(X_1 = x_1 \mid X_2 = x_2) P(X_2 = x_2)$$

Review of Basic Concepts (2)

Chain rule:

$$P(X_1 = x_1 \mid X_2 = x_2, ..., X_n = x_n) \quad = \quad \frac{P(X_1{=}x_1 \wedge X_2{=}x_2 \wedge \cdots \wedge X_n{=}x_n)}{P(X_2{=}x_2 \wedge \cdots \wedge X_n{=}x_n)}$$

or

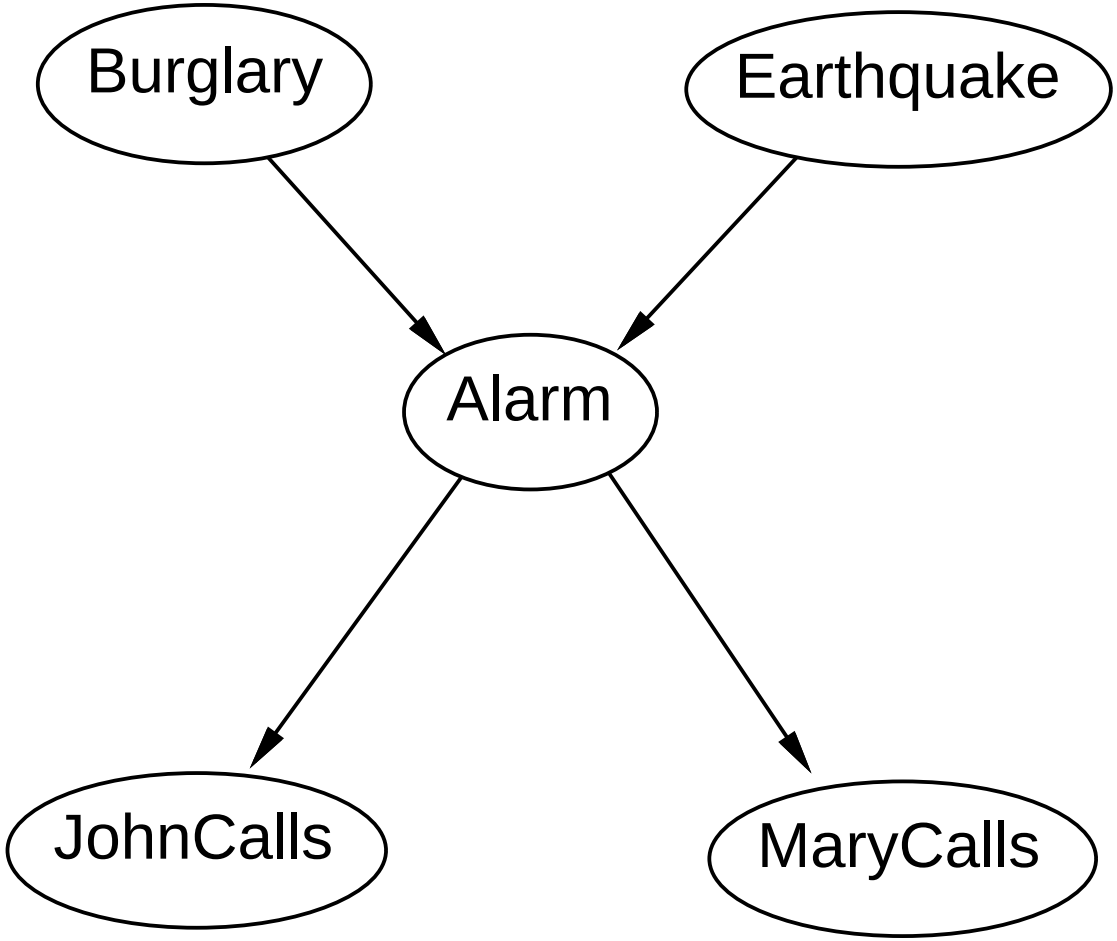
$$P(X_1 = x_1 \wedge X_2 = x_2 \wedge \cdots \wedge X_n = x_n)$$
$$= \; P(X_1 = x_1 \mid X_2 = x_2, ..., X_n = x_n) P(X_2 = x_2 \wedge \cdots \wedge X_n = x_n)$$
$$= \; \prod_{i=1}^{n} P(X_i = x_i \mid X_{i+1} = x_{i+1}, ..., X_n = x_n)$$

Conditional Independence: If

$$P(X_1 = x_1 \mid X_2 = x_2, ..., X_n = x_n) \quad = \quad P(X_1 = x_1 \mid X_2 = x_2, ..., X_{n-1} = x_{n-1})$$

then $X_1 = x_1$ is conditionally independent from $X_n = x_n$ given the evidence $X_2 = x_2, ..., X_{n-1} = x_{n-1}$

A Belief Network



Belief Networks

Let $X_1, \ldots, X_n$ be discrete random variables.

A belief network (or Bayesian network) for $X_1, \ldots, X_n$ is a graph with m nodes such that

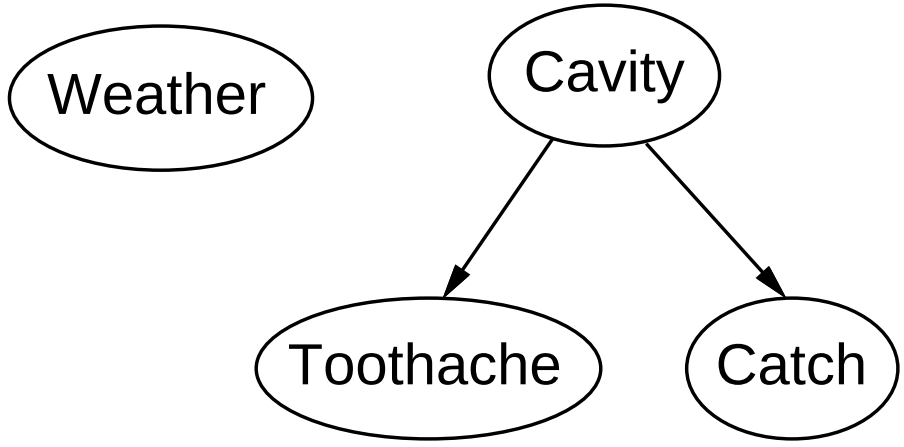
- there is a node for each X_i
- all the edges between two nodes are directed
- there are no cycles
- each node has a conditional probability table (CPT), given in terms of its parents

The intuitive meaning of an edge from a node X_i to a node X_j is that X_i has a direct influence on X_j

Network Semantics

The topology of the network encodes conditional independence assertions

Example:



- $Weather$ is independent from the other variables
- $Toothache$ and $Catch$ are conditionally independent given $Cavity$

Conditional Probability Tables

Each node X_i in a belief network has an associated CPT expressing the probability of X_i, given its parents as evidence

Example:

CPT for $Alarm$:

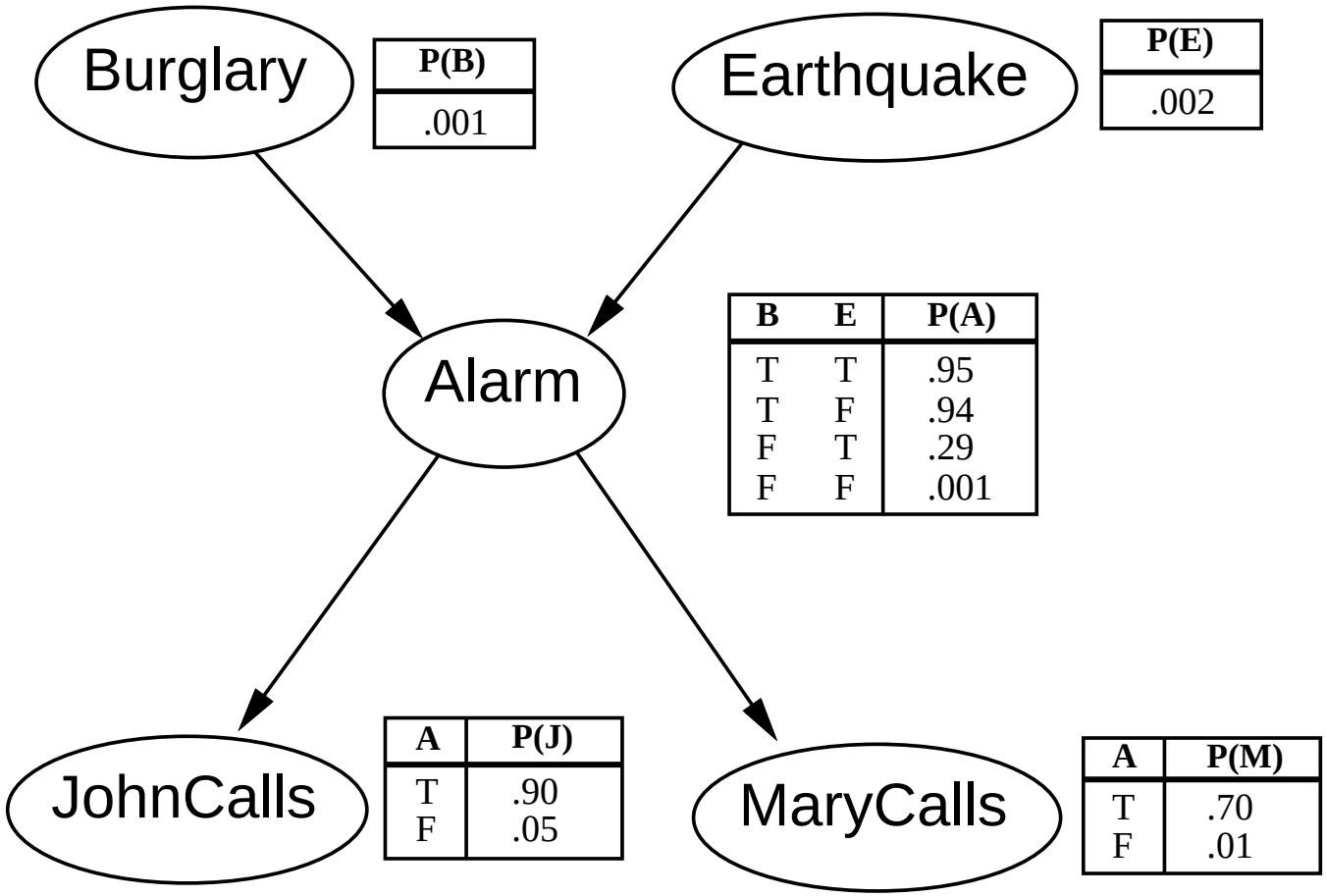
$Burglary$	$Earthquake$	$Alarm$ T	F
T	T	0.950	0.050
T	F	0.940	0.060
F	T	0.290	0.710
F	F	0.001	0.999

$P(alarm \mid burglary \wedge earthquake) = 0.950$

$P(\neg alarm \mid \neg burglary \wedge earthquake) = 0.710$

...

A Belief Network with CPTs



Note: The tables only show $P(X = true)$ here because $P(X = false) = 1 - P(X = true)$

The Semantics of Belief Networks

There are two equivalent ways to interpret a belief network for the variables $X_1, \ldots, X_n$:

1. The network is a representation of the JPD $\mathbf{P}(X_1, \ldots, X_n)$
2. The network is a collection of conditional independence statements about $X_1, \ldots, X_n$

Interpretation 1 is helpful when constructing belief networks

Interpretation 2 is helpful in designing inference procedures based on them

Belief Network as JPDs

The whole JPD $\mathbf{P}(X_1, \ldots, X_n)$ can be computed from a belief network for $X_1, \ldots, X_n$ and its CPTs

For each tuple $\langle x_1, \ldots, x_n \rangle$ of possible values for $\langle X_1, \ldots, X_n \rangle$,

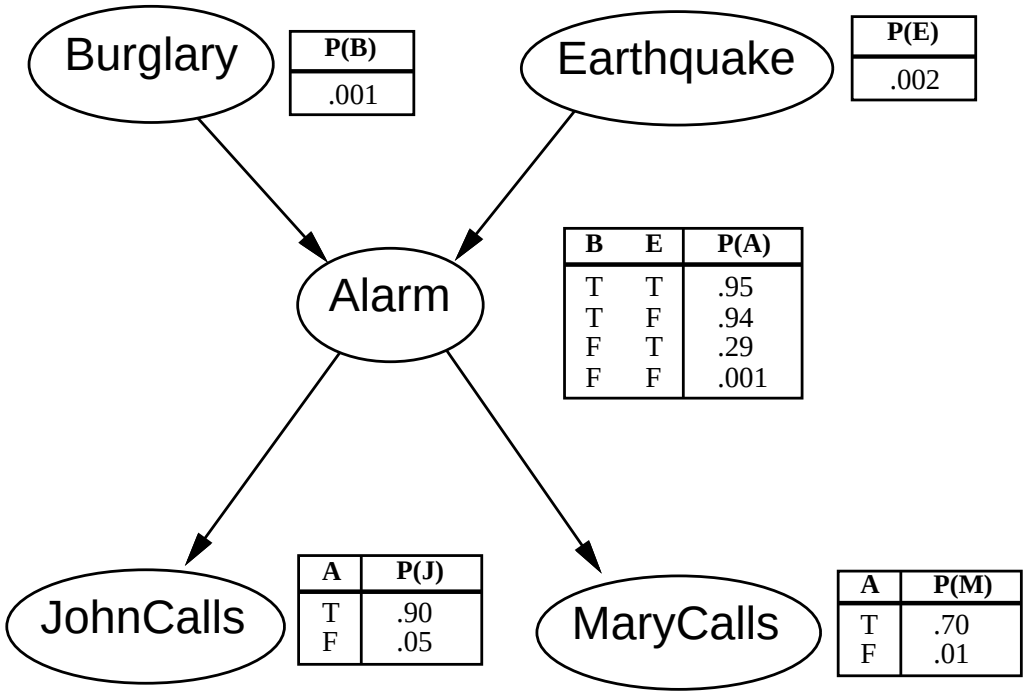
$$P(X_1 = x_1 \wedge \cdots \wedge X_n = x_n) = \prod_{i=1}^{n} P(X_i = x_i \mid Parents(X_i))$$

where

$$Parents(X_i) = \{X_j = x_j \mid 1 \leq j \leq n \text{ and } X_j \text{ is a parent of } X_i\}$$

Belief Network as JPDs

$$P(X_1 = x_1 \wedge \cdots \wedge X_n = x_n) \;=\; \prod_{i=1}^{n} P(X_i = x_i \mid Parents(X_i))$$



$$P(j \wedge m \wedge a \wedge \neg b \wedge \neg e)$$
$$= \; P(j \mid a)\, P(m \mid a)\, P(a \mid \neg b \wedge \neg e)\, P(\neg b)\, P(\neg e)$$
$$= \; 0.9 \times 0.7 \times 0.001 \times 0.999 \times 0.998 = 0.00062$$

Belief Networks and Cond. Independence

Let $\{X_1, X_2, \ldots, X_n\}$ be any set of nodes in the network such that

- all the parents of X_1 are in $\{X_2, \ldots, X_n\}$
- no node in $\{X_2, \ldots, X_n\}$ is a descendant of X_1

Let $\langle x_1, \ldots, x_n \rangle$ be a value assignment for $\langle X_1, \ldots, X_n \rangle$

From the equation

$$P(X_1 = x_1 \wedge \cdots \wedge X_n = x_n) = \prod_{i=1}^{n} P(X_i = x_i \mid Parents(X_i))$$

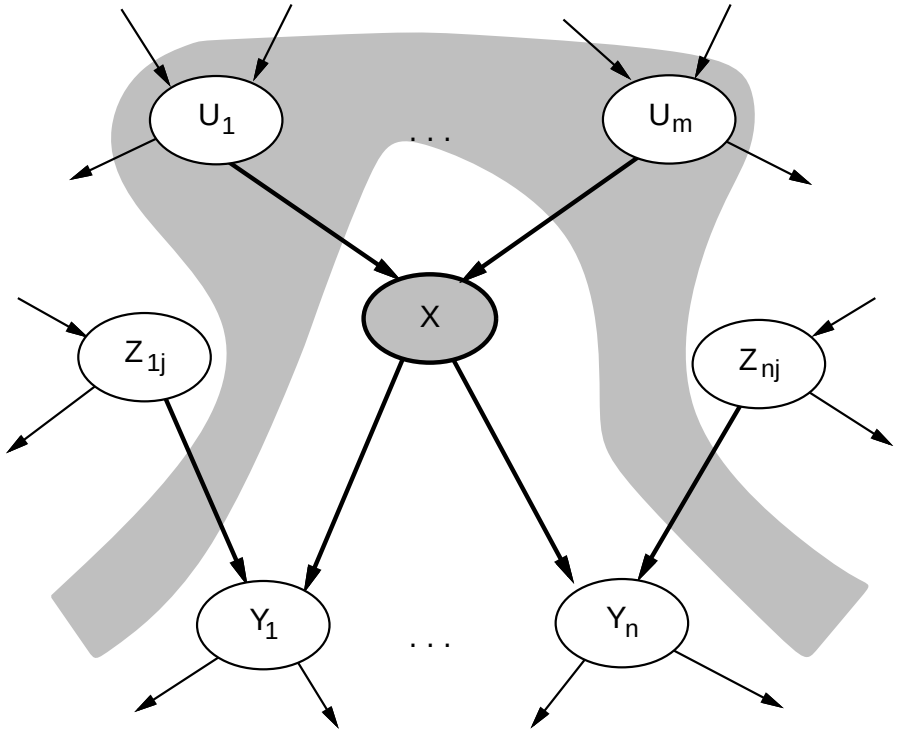
we can show that

$$P(X_1 = x_1 \mid X_2 = x_2 \wedge \cdots \wedge X_n = x_n) = P(X_1 = x_1 \mid Parents(X_1))$$

Belief Networks and Cond. Independence

A consequence of the last equation is:

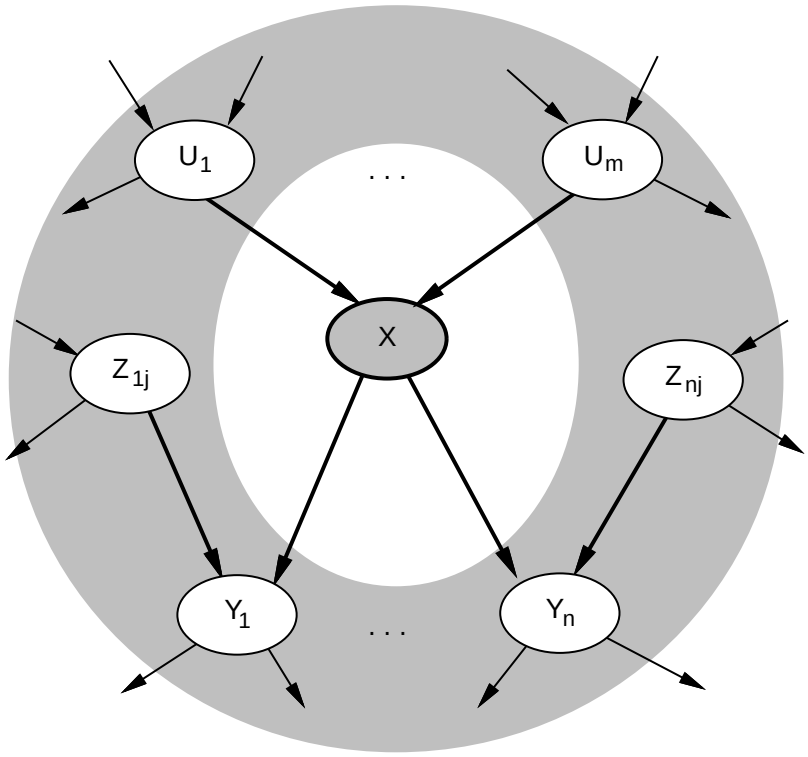
> *Given all its parents as evidence, each node in the network is conditionally independent from its non-descendants*



Belief Networks and Cond. Independence

Another consequence is:

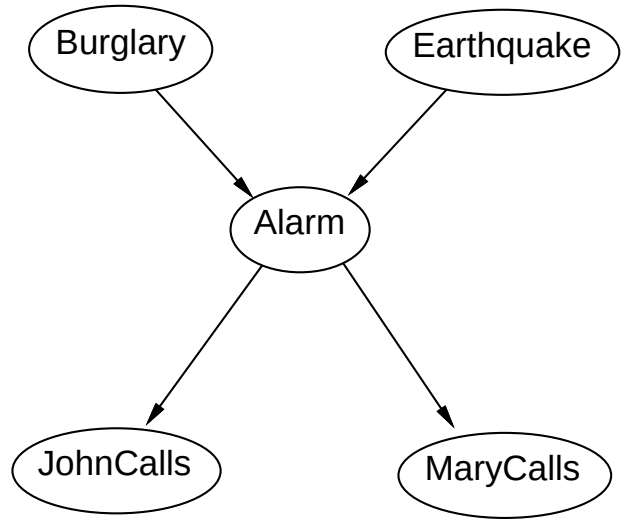
> *Given all its parents, children, and children's parents as evidence, each node in the network is conditionally independent from all the other nodes*



Belief Networks and Cond. Independence

$$P(X_1 = x_1 \mid X_2 = x_2 \wedge \cdots \wedge X_n = x_n) \;=\; P(X_1 = x_1 \mid Parents(X_1))$$

Examples:



$$P(b \mid e) = P(b)$$
$$P(j \mid m \wedge a) = P(j \mid a)$$
$$P(j \mid a \wedge e) = P(j \mid a)$$
$$P(j \mid a \wedge b \wedge e) = P(j \mid a)$$
$$P(j \mid m \wedge a \wedge b \wedge e) = P(j \mid a)$$

Exercise: Find all the conditional independences holding in this network

Constructing Belief Networks

General Procedure

1. Identify a set of random variables $\{X_i\}_i$ that describe the domain
2. Choose an ordering $X_1, \ldots, X_n$ of the variables
3. Start with an empty network
4. For $i = 1 \ldots n$:
   (a) add X_i to the network
   (b) select as parents of X_i nodes from $X_1, \ldots, X_{i-1}$ such that $\mathbf{P}(X_i \mid Parents(X_i)) = \mathbf{P}(X_i \mid X_1, \ldots, X_{i-1})$
   (c) fill in the CPT for X_i

Constructing Belief Networks

General Procedure

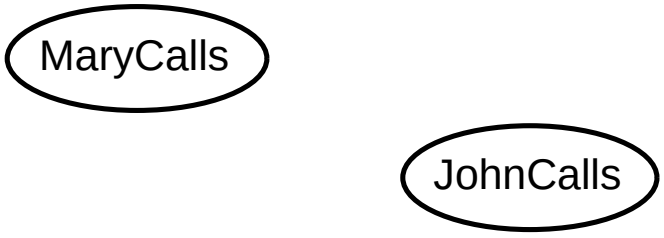
1. Identify a set of random variables $\{X_i\}_i$ that describe the domain
2. Choose an ordering $X_1, \ldots, X_n$ of the variables
3. Start with an empty network
4. For $i = 1 \ldots n$:
   (a) add X_i to the network
   (b) select as parents of X_i nodes from $X_1, \ldots, X_{i-1}$ such that $\mathbf{P}(X_i \mid Parents(X_i)) = \mathbf{P}(X_i \mid X_1, \ldots, X_{i-1})$
   (c) fill in the CPT for X_i

This choice of parents guarantees the network semantics:

$$\begin{aligned} \mathbf{P}(X_1, \ldots, X_n) &= \prod_{i=1}^{n} \mathbf{P}(X_i \mid X_1, \ldots, X_{i-1}) && \text{(chain rule)} \\ &= \prod_{i=1}^{n} \mathbf{P}(X_i \mid Parents(X_i)) && \text{(by construction)} \end{aligned}$$

Example

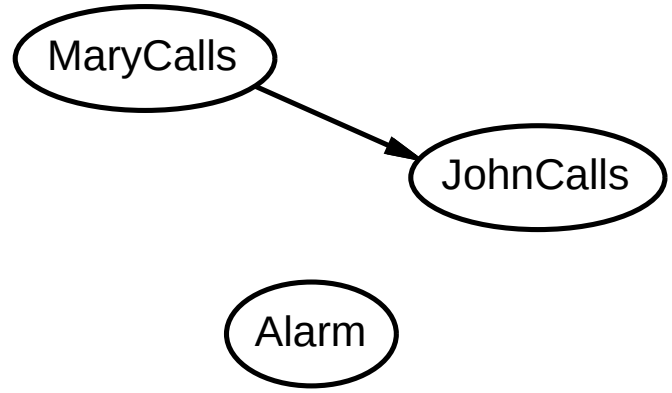
Suppose we choose the ordering $MaryCalls$, $JohnCalls$, $Alarm$, $Burglary$, $Earthquake$



$P(j \mid m) = P(j)$?

Example

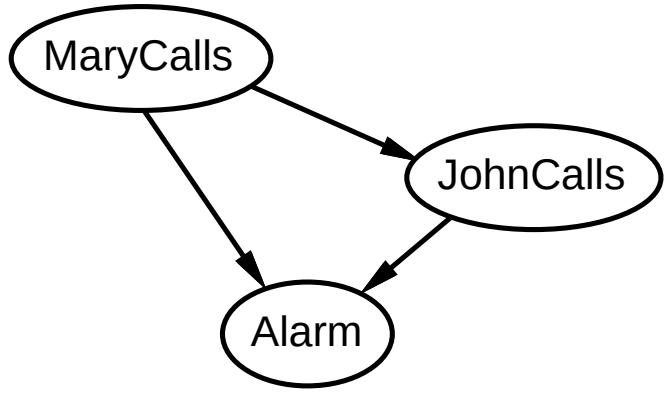
Suppose we choose the ordering $MaryCalls$, $JohnCalls$, $Alarm$, $Burglary$, $Earthquake$



$P(j \mid m) = P(j)$? No
$P(a \mid j, m) = P(a \mid j)$?
$P(a \mid j, m) = P(a)$?

Example

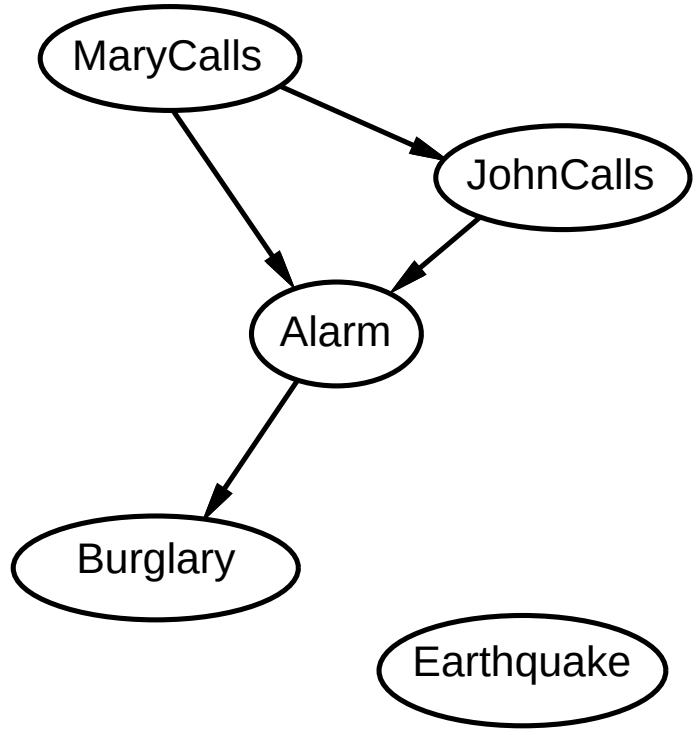
Suppose we choose the ordering $MaryCalls$, $JohnCalls$, $Alarm$, $Burglary$, $Earthquake$



$P(j \mid m) = P(j)$? No
$P(a \mid j, m) = P(a \mid j)$? No
$P(a \mid j, m) = P(a)$? No
$P(b \mid a, j, m) = P(b \mid a)$?
$P(b \mid a, j, m) = P(b)$?

Example

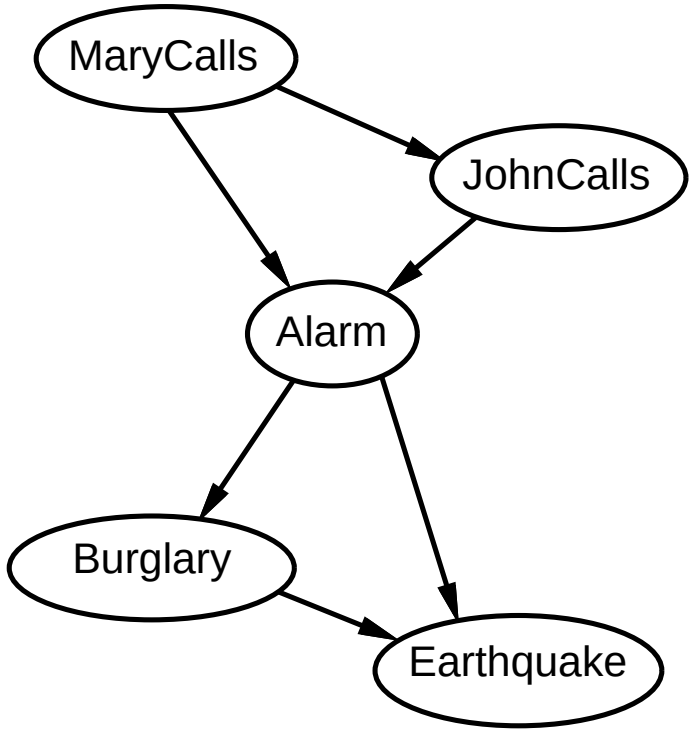
Suppose we choose the ordering $MaryCalls$, $JohnCalls$, $Alarm$, $Burglary$, $Earthquake$



$P(j \mid m) = P(j)$? No
$P(a \mid j, m) = P(a \mid j)$? No
$P(a \mid j, m) = P(a)$? No
$P(b \mid a, j, m) = P(b \mid a)$? Yes
$P(b \mid a, j, m) = P(b)$? No
$P(e \mid b, a, j, m) = P(e \mid a)$?
$P(e \mid b, a, j, m) = P(e \mid a, b)$?

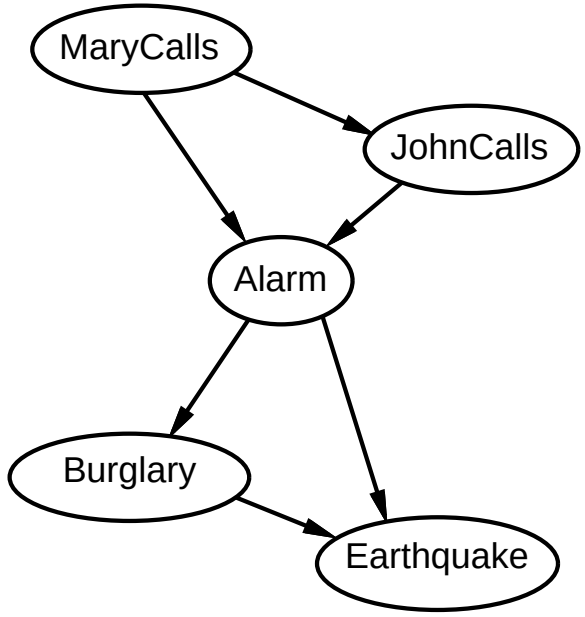
Example

Suppose we choose the ordering $MaryCalls$, $JohnCalls$, $Alarm$, $Burglary$, $Earthquake$



$P(j \mid m) = P(j)$? No
$P(a \mid j, m) = P(a \mid j)$? No
$P(a \mid j, m) = P(a)$? No
$P(b \mid a, j, m) = P(b \mid a)$? Yes
$P(b \mid a, j, m) = P(b)$? No
$P(e \mid b, a, j, m) = P(e \mid a)$? No
$P(e \mid b, a, j, m) = P(e \mid a, b)$? Yes

Example contd.



Deciding conditional independence is hard in non-causal directions

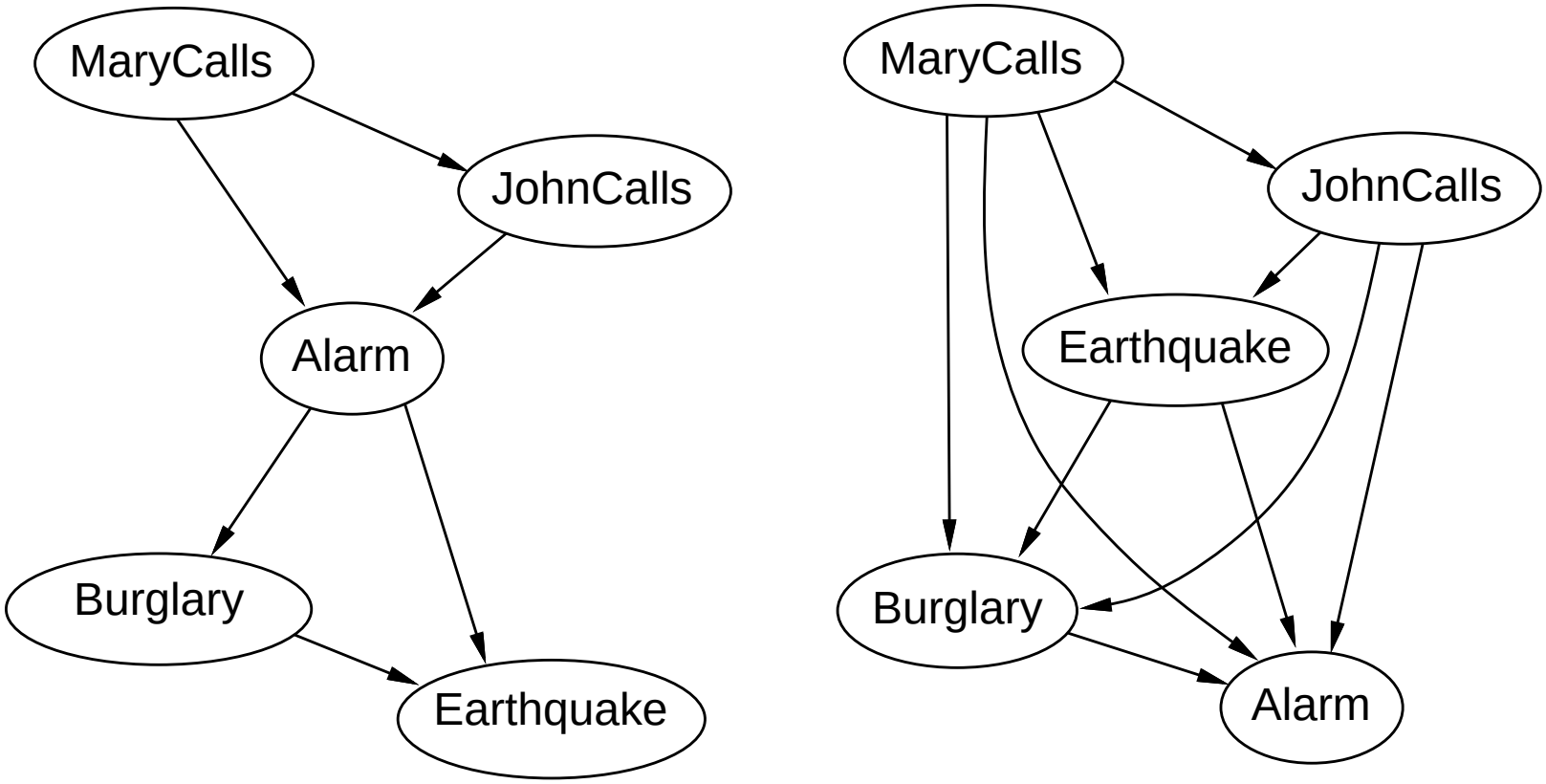
Causal models and conditional independence seem hardwired for humans!

Assessing conditional probabilities is hard in non-causal directions

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

Ordering the Variables

The order in which add the variables to the network is important.

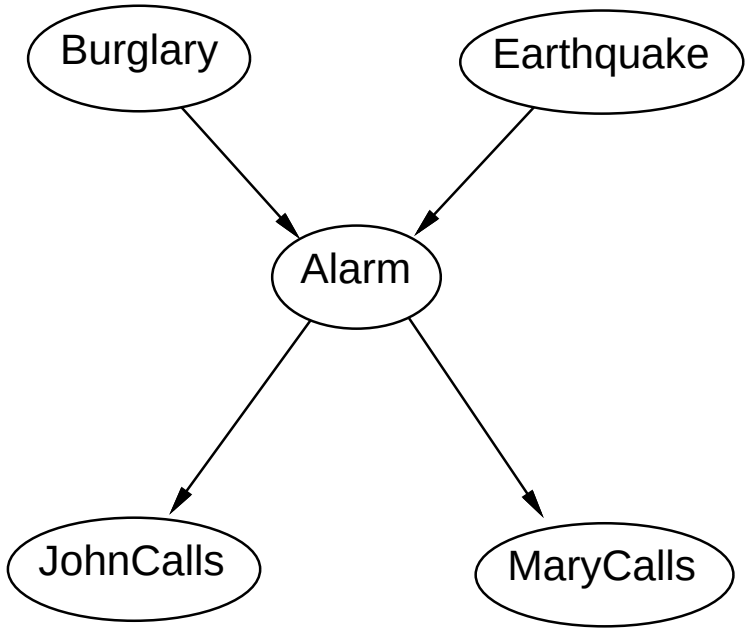


"Wrong" orderings produces more complex networks.

Ordering the Variables Right

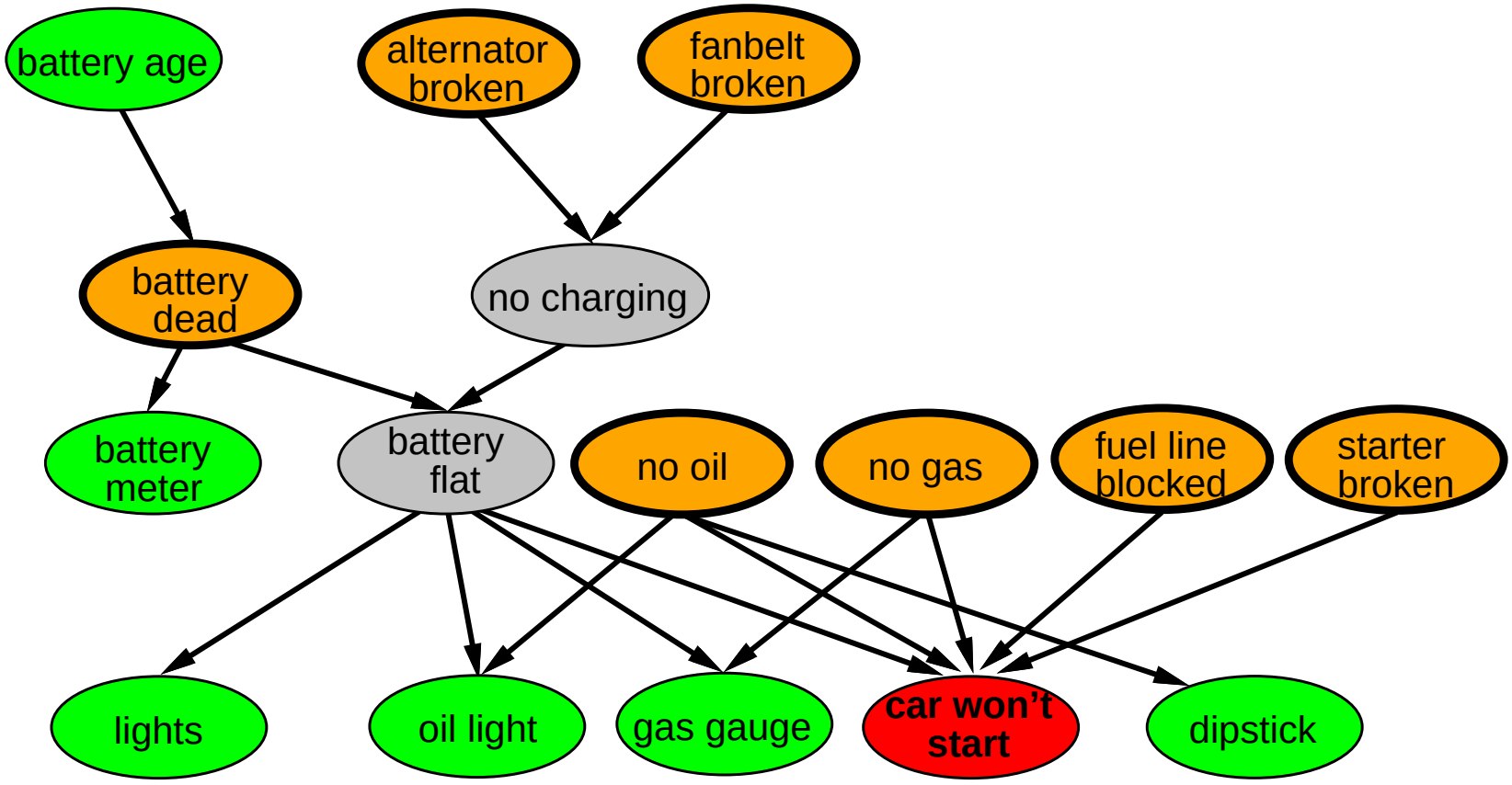
A general, effective heuristic for constructing simpler belief networks is to exploit causal links between random variables whenever possible

This is done by adding variables to the network so that causes get added before effects



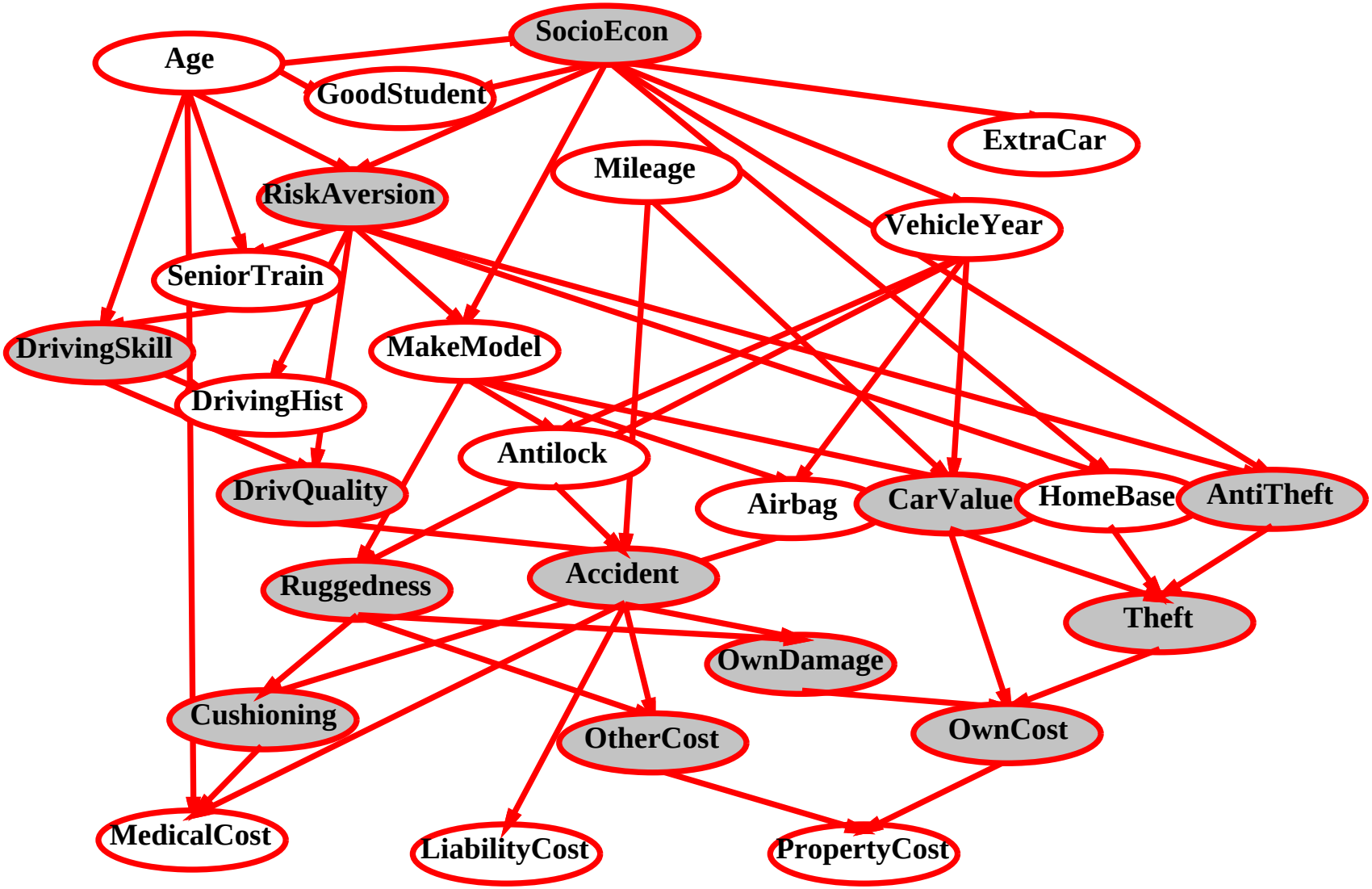
Example: Car diagnosis

Initial evidence: car won't start



Testable variables (green), actionable variables (orange)
Hidden variables (gray) ensure sparse structure, reduce parameters

Example: Car insurance



Compactness of Belief Networks

- A belief network is a complete and non-redundant representation of a full joint probability distribution.
- In addition, its is often more compact than a full joint.
- The reason is that probabilistic domains are often representable as a locally structured system.
- In a locally structured system, each subcomponent interacts with only a bounded number of other components, regardless of the size of the system.
- The complexity of local structures generally grows linearly, instead of exponentially.

Locally Structured Systems

- In many real-world domains, each random variable is influenced by at most k others, for some fixed constant k.
- Assuming that we have n variables and, for simplicity, that they are all Boolean, a joint table will have 2^n entries.
- In a well constructed belief network, each node will have at most k parents.
- This means that each node will have a CPT with at most 2^k entries, for a total of $n2^k$ entries.
- Example: $n = 20$, $k = 5$

$$\begin{array}{lrcr} \text{entries in network CPTs} & \leq & 20 \times 2^5 & = & 640 \\ \text{entries in joint table} & = & 2^{20} & = & 1,048,576 \end{array}$$

Representing CPTs

Even with a fairly small number of parents per node, constructing the CPTs for a belief network may require a lot a work

If the network is built with the right topology, however, the relationship between parent and children nodes will typically fall into a category with some canonical distribution

Examples:

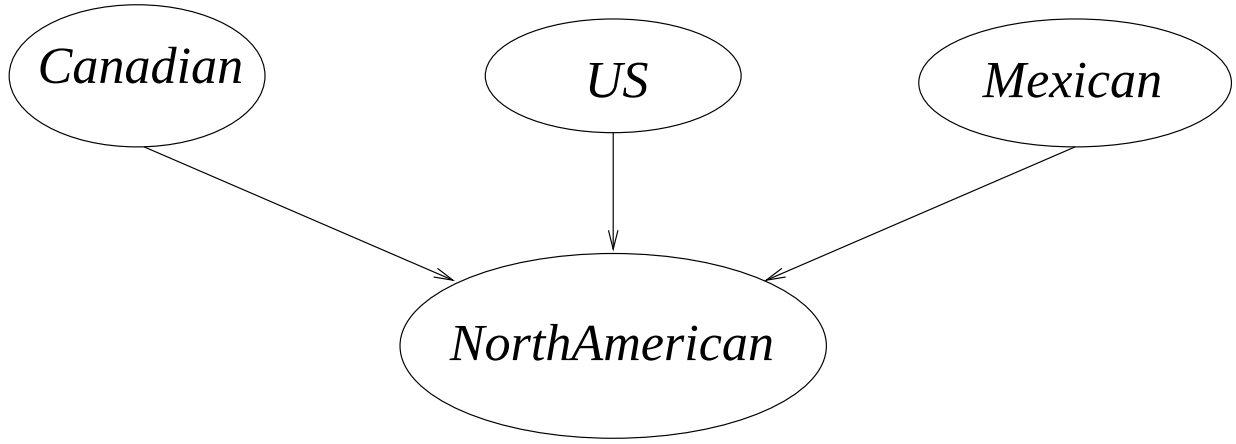
- Deterministic nodes
- Noisy-OR relationships

Deterministic Nodes

A node is deterministic if its value is a function of the values of its parents, with no uncertainty.

- Logical implications or equivalences:

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



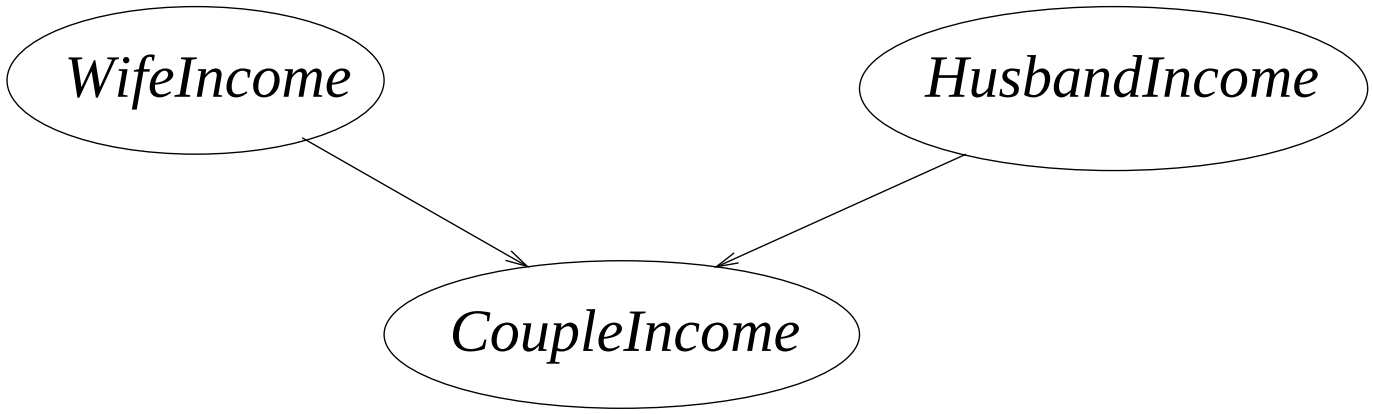
$$P(n \mid \neg c \wedge \neg u \wedge \neg m) = 0$$

$$P(n \mid u) = P(n \mid c \wedge u) = \ldots = P(n \mid u \wedge \neg m)$$
$$= P(n \mid c \wedge \neg u \wedge \neg m) = \ldots = 1$$

Deterministic Nodes (cont.)

- Functional relationships:

$$CoupleIncome = WifeIncome + HusbandIncome$$



$$P(C = 80,000 \mid H = 50,000 \wedge W = 30,000) = 1$$
$$P(C = 95,000 \mid H = 50,000 \wedge W = 30,000) = 0$$

Noisy-OR

A generalization of logical OR. Adds uncertainty to statements like

$$Fever \Leftrightarrow Cold \vee Flu \vee Malaria$$

Three assumptions are made:

1. Each cause has an independent chance of producing the effect
2. All possible causes are listed
3. The reason for a cause not to produce the effect is independent from the reason for another cause not to produce the effect:

$$P(\neg Effect \mid Cause_i \wedge OtherCauses)$$
$$= P(\neg Effect \mid Cause_i)\, P(\neg Effect \mid OtherCauses)$$

The possibility that a cause does not produce an effect is given by a noise-parameter

CPTs for Noisy-ORs

Knowing the noise parameters (in boldface below) is enough for computing the whole CTP

CTP of $Fever$:

$Cold$	Flu	$Malaria$	$P(Fever)$	$P(\neg Fever)$
F	F	F	0.00	1.00
F	F	T	0.90	**0.10**
F	T	F	0.80	**0.20**
F	T	T	0.98	$0.02 = 0.2 \times 0.1$
T	F	F	0.40	**0.60**
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	$0.012 = 0.6 \times 0.2 \times 0.1$

A noisy-OR with k causes can be specified with k values, the noise parameters, instead of the 2^k values of a full CPT

Inference in Belief Networks

Main task of a belief network: Compute the conditional probability of a set of query variables, given exact values for some evidence variables

$$P(Query \mid Evidence)$$

Belief networks are flexible enough so that any node can serve as either a query or an evidence variable

In general, to decide what actions to take, an agent

1. first gets values for some variables from its percepts, or from its own reasoning
2. then asks the network about the possible values of the other variables

Probabilistic Inference with BNs

Belief networks are a very flexible tool for probabilistic inference because they allow several kinds of inference:

Diagnostic inference (from effects to causes)

E.g. $P(Burglary \mid JohnCalls)$

Causal inference (from causes to effects)

E.g. $P(JohnCalls \mid Burglary)$

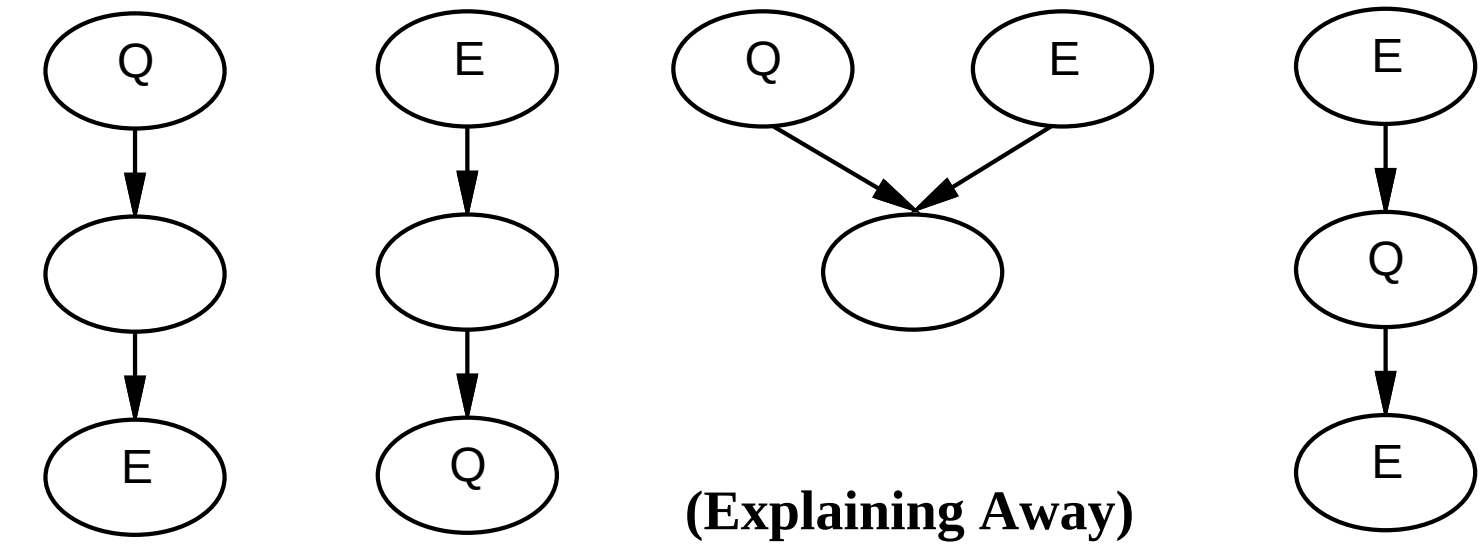
Intercausal inference (between causes of a common effect)

$P(Burglary \mid Alarm \wedge Earthquake)$

Mixed inference (combination of the above)

$P(Alarm \mid JohnCalls \wedge \neg Earthquake)$

Types of Inference in Belief Networks



Q = query variable

E = evidence variable

Inference tasks

Simple queries: compute posterior marginal $\mathbf{P}(X_i \mid \mathbf{E}=\mathbf{e})$

E.g., $P(NoGas \mid Gauge=empty, Lights=on, Starts=false)$

Conjunctive queries:

$$\mathbf{P}(X_i, X_j \mid \mathbf{E}=\mathbf{e}) = \mathbf{P}(X_i \mid \mathbf{E}=\mathbf{e})\mathbf{P}(X_j \mid X_i, \mathbf{E}=\mathbf{e})$$

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

Value of information: which evidence to seek next?

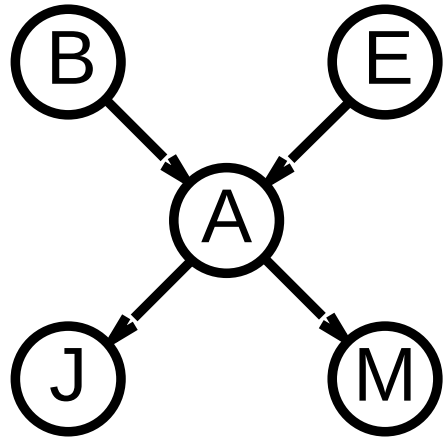
Sensitivity analysis: which probability values are most critical?

Explanation: why do I need a new starter motor?

Inference by enumeration

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

Simple query on the burglary network:



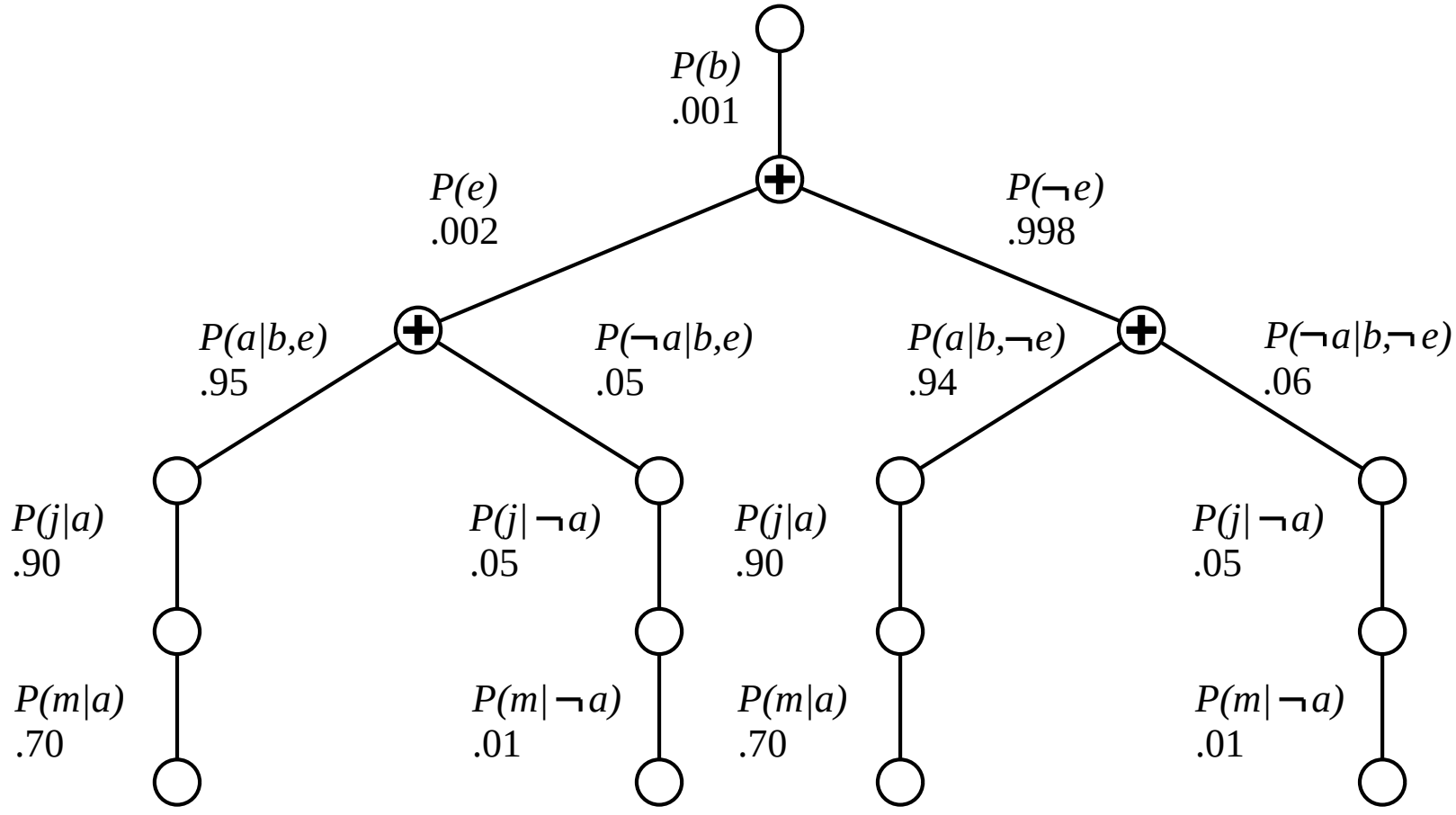
$$\begin{aligned}
&\mathbf{P}(B \mid j, m) \\
&\quad = \mathbf{P}(B, j, m)/P(j, m) \\
&\quad = \alpha \mathbf{P}(B, j, m) \\
&\quad = \alpha \sum_e \sum_a \mathbf{P}(B, e, a, j, m)
\end{aligned}$$

Rewrite full joint entries using product of CPT entries:

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

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

Evaluation tree



Enumeration is inefficient: repeated computation

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

Inference by variable elimination

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

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

Variable elimination: Basic operations

Summing out a variable from a product of factors:

1. move any constant factors outside the summation
2. add up submatrices in pointwise product of remaining factors

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

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

Pointwise product of factors f_1 and f_2:

$$f_1(x_1, \ldots, x_j, y_1, \ldots, y_k) \times f_2(y_1, \ldots, y_k, z_1, \ldots, z_l) = f(x_1, \ldots, x_j, y_1, \ldots, y_k, z_1, \ldots, z_l)$$

E.g., $f_1(A, B) \times f_2(B, C) = f(A, B, C)$

Pointwise Multiplication

A	B	$\mathbf{f}_1(A,B)$	B	C	$\mathbf{f}_2(B,C)$	A	B	C	$\mathbf{f}_3(A,B,C)$
T	T	0.3	T	T	0.2	T	T	T	$0.3 \times 0.2 = 0.06$
T	F	0.7	T	F	0.8	T	T	F	$0.3 \times 0.8 = 0.24$
F	T	0.9	F	T	0.6	T	F	T	$0.7 \times 0.6 = 0.42$
F	F	0.1	F	F	0.4	T	F	F	$0.7 \times 0.4 = 0.28$
						F	T	T	$0.9 \times 0.2 = 0.18$
						F	T	F	$0.9 \times 0.8 = 0.72$
						F	F	T	$0.1 \times 0.6 = 0.06$
						F	F	F	$0.1 \times 0.4 = 0.04$

Variable elimination algorithm

function ELIMINATION-ASK(X, **e**, bn) **returns** a distribution over X

inputs: X, the query variable

e, evidence specified as an event

bn, a belief network specifying joint distribution $\mathbf{P}(X_1, \ldots, X_n)$

$factors \leftarrow [\,]$; $vars \leftarrow$ REVERSE(VARS[bn])

for each var **in** $vars$ **do**

$factors \leftarrow [$MAKE-FACTOR(var, **e**)$|factors]$

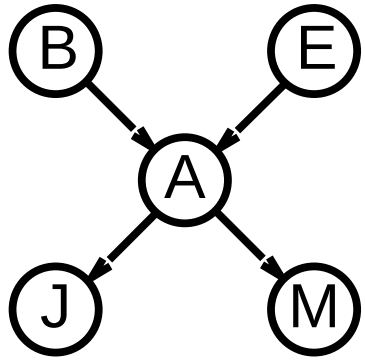
if var is a hidden variable **then** $factors \leftarrow$ SUM-OUT($var, factors$)

return NORMALIZE(POINTWISE-PRODUCT($factors$))

Any ordering of the variables will do for correctness

Following topological order over BN is usually most efficient (although finding optimal ordering is NP-hard)

Irrelevant variables



Consider the query $P(JohnCalls \mid Burglary = true)$

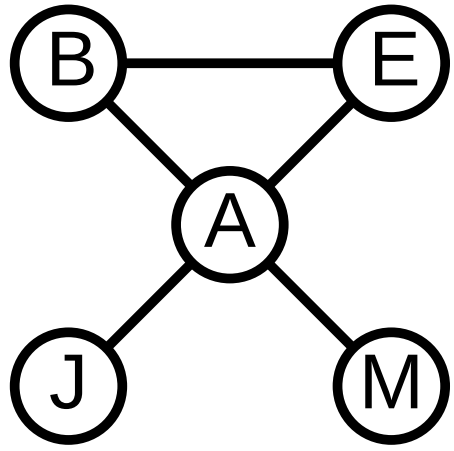
$$P(J \mid b) = \alpha P(b) \sum_e P(e) \sum_a P(a \mid b, e) P(J \mid a) \sum_m P(m \mid a)$$

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

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

Here, $X = J$, $\mathbf{E} = \{B\}$, and $Ancestors(\{X\} \cup \mathbf{E}) = \{A, B, E\}$
so M is irrelevant

Irrelevant variables contd.



Defn: moral graph of belief network: marry all parents and drop arrows

Defn: $\mathbf{A}$ is m-separated from $\mathbf{B}$ by $\mathbf{C}$ iff separated by $\mathbf{C}$ in the moral graph

Thm 2: Y is irrelevant if m-separated from X by $\mathbf{E}$

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

Complexity of exact inference

Singly connected networks (or polytrees):

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

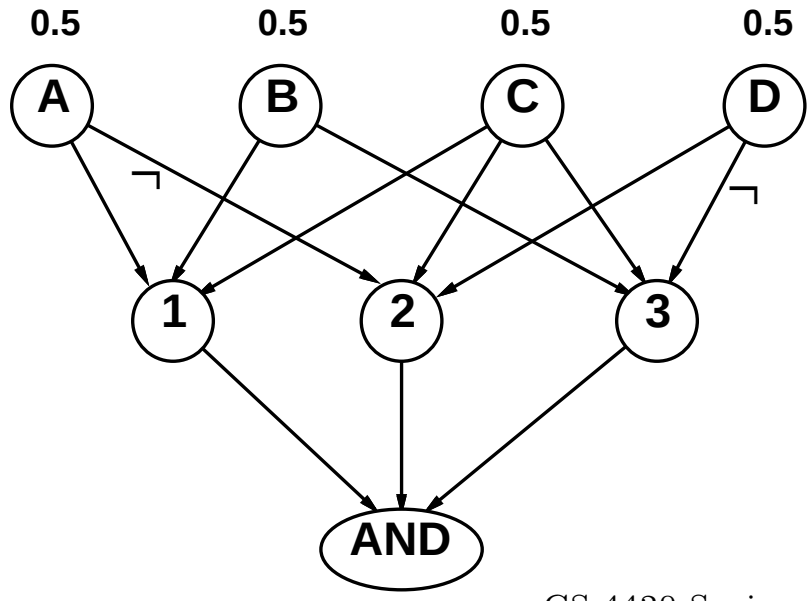
Multiply connected networks:

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

1. A v B v C

2. C v D v ¬A

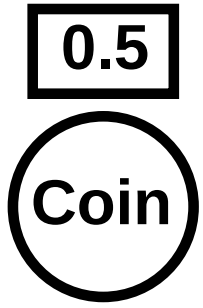
3. B v C v ¬D



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

Sampling from an empty network

function PRIOR-SAMPLE(bn) **returns** an event sampled from bn

inputs: bn, a belief network specifying jpd $\mathbf{P}(X_1, \ldots, X_n)$

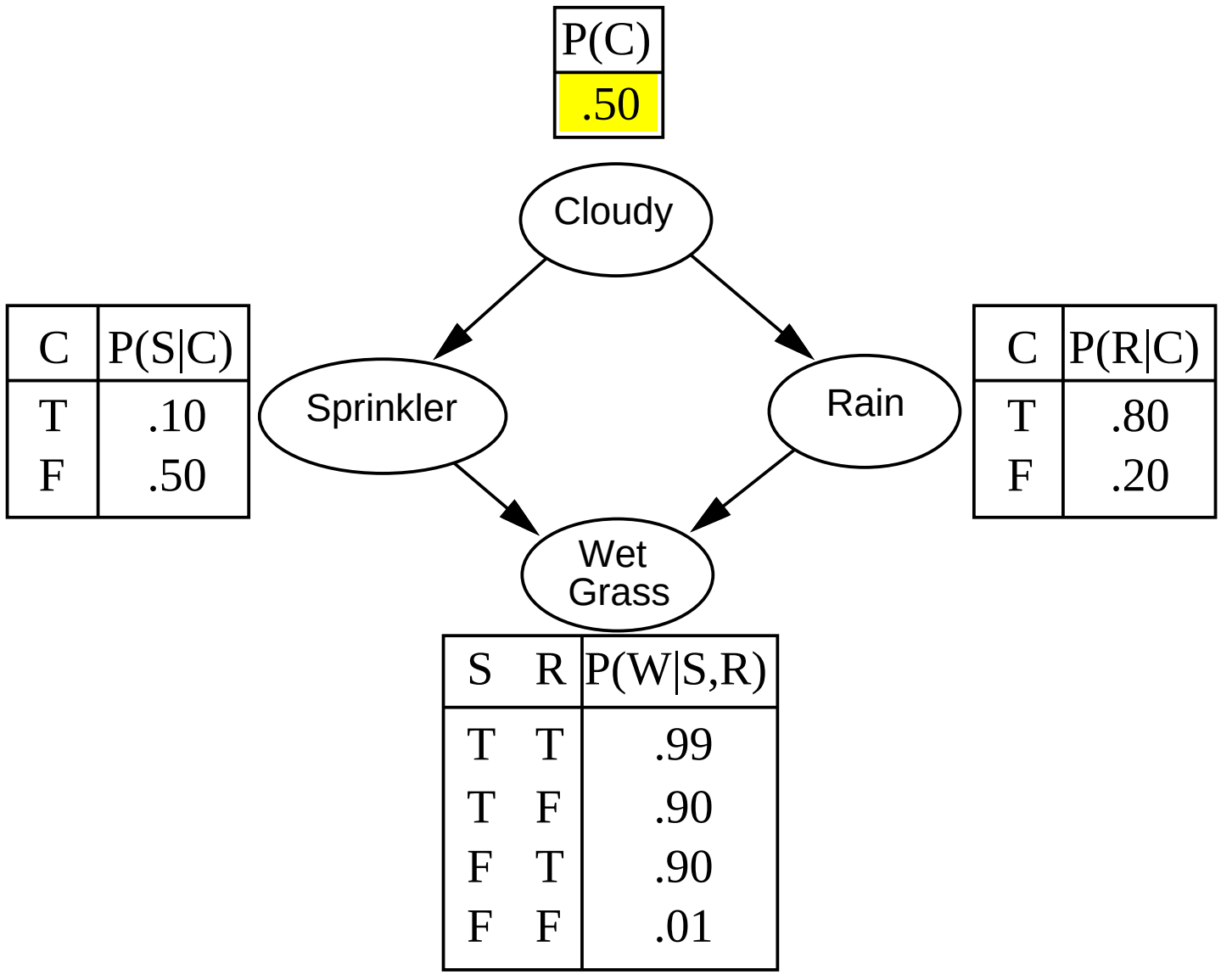
$\mathbf{x} \leftarrow$ an event with n elements

for $i = 1$ **to** n **do**

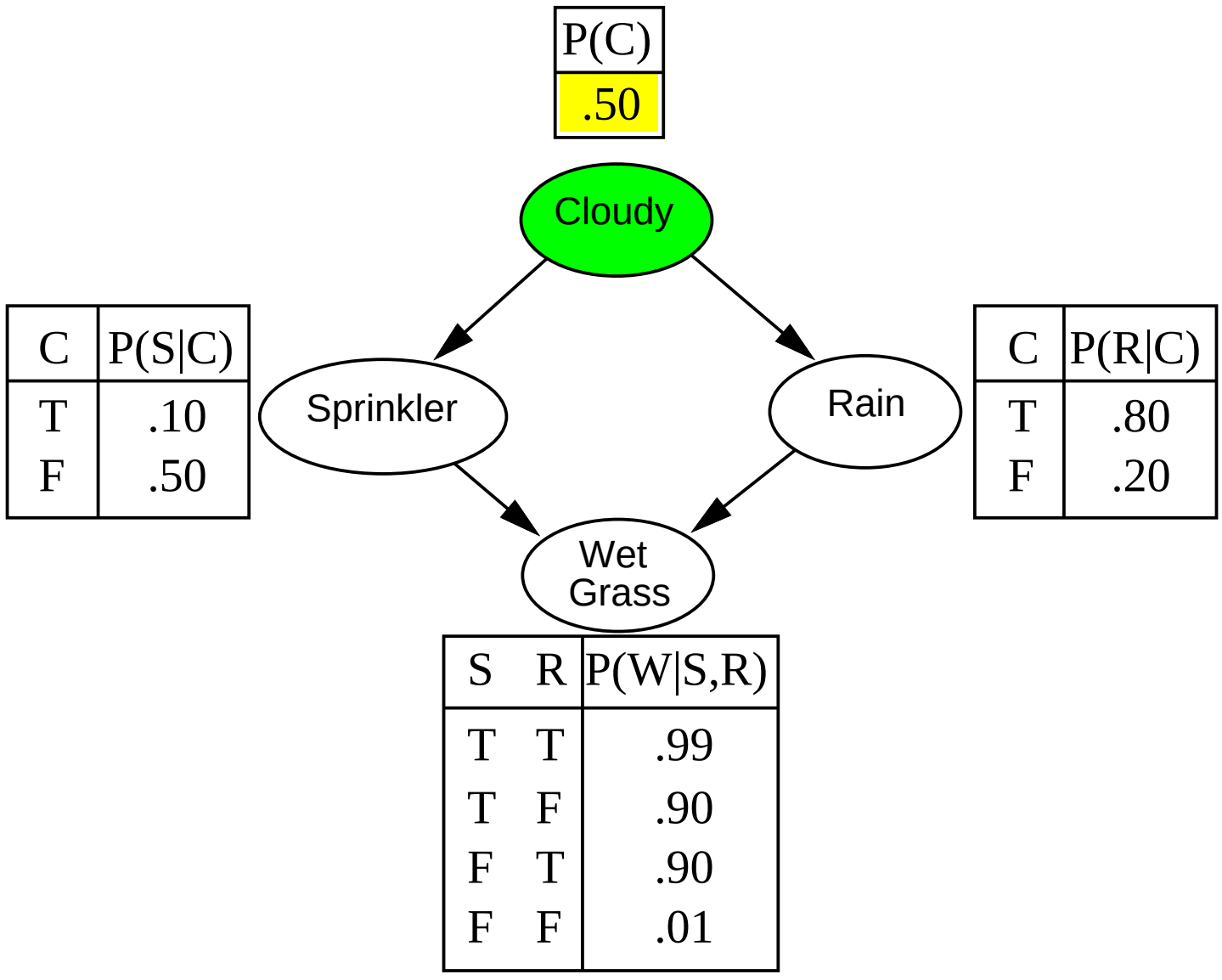
$x_i \leftarrow$ a random sample from $\mathbf{P}(X_i \mid Parents(X_i))$

return x

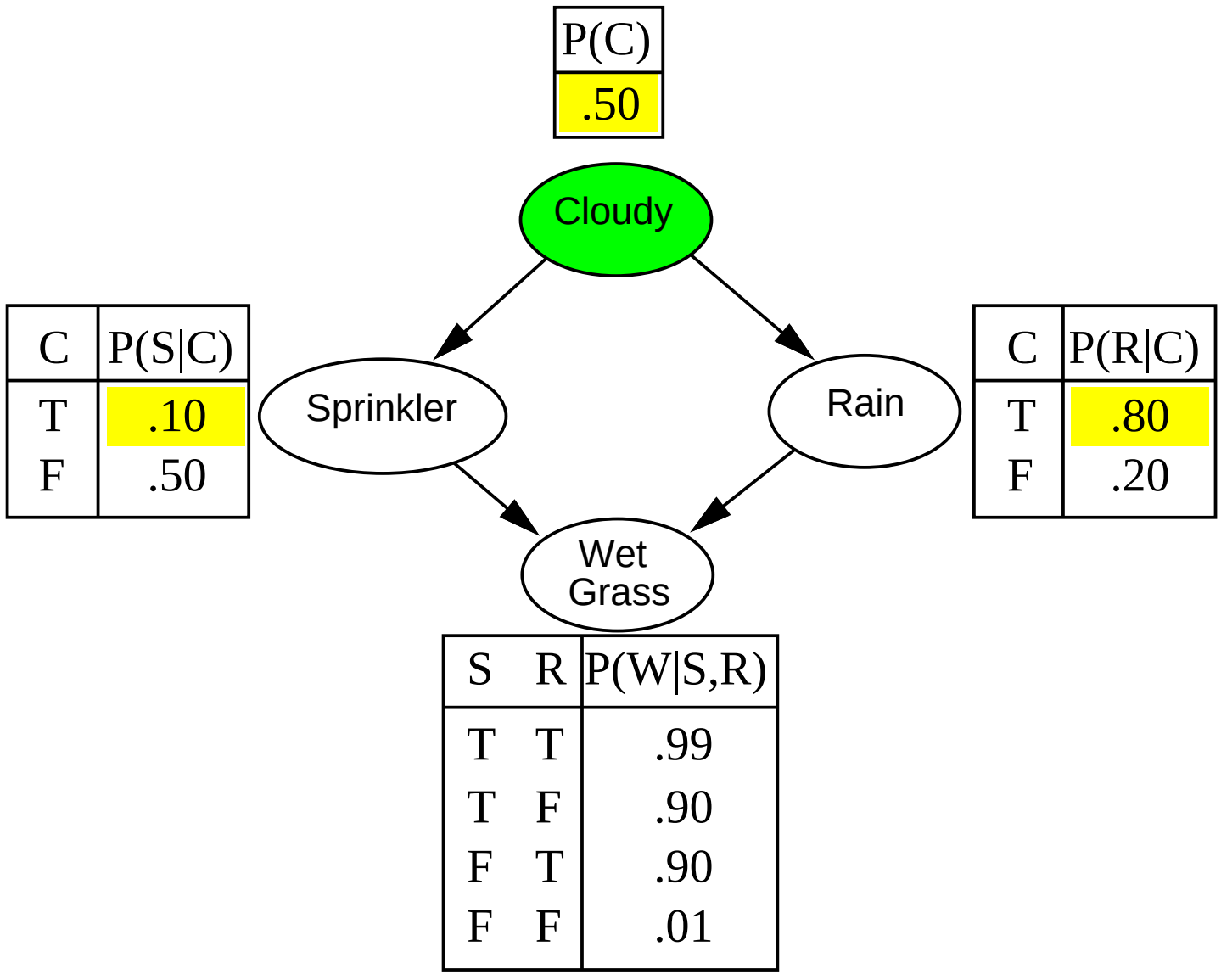
Example


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

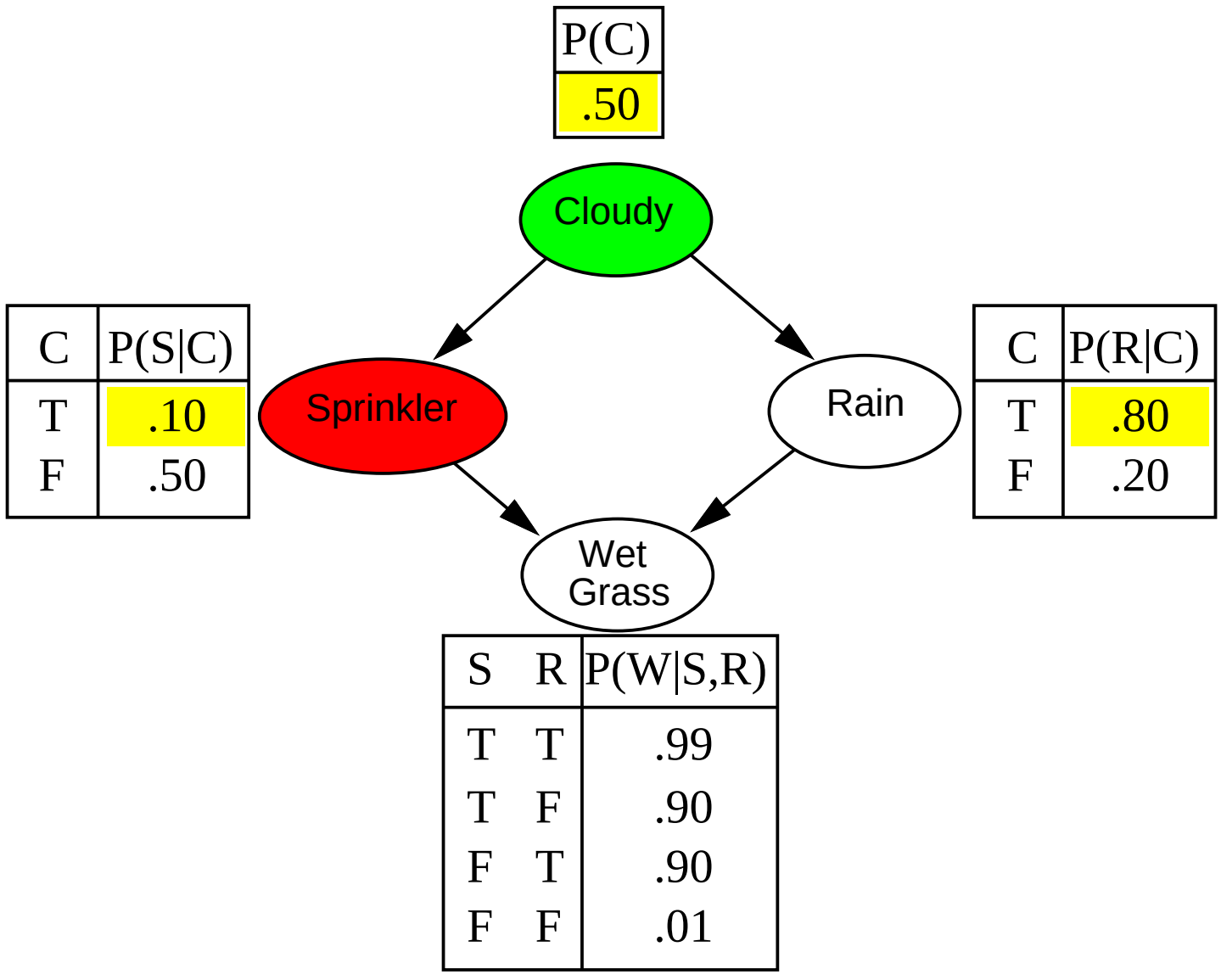
Example


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

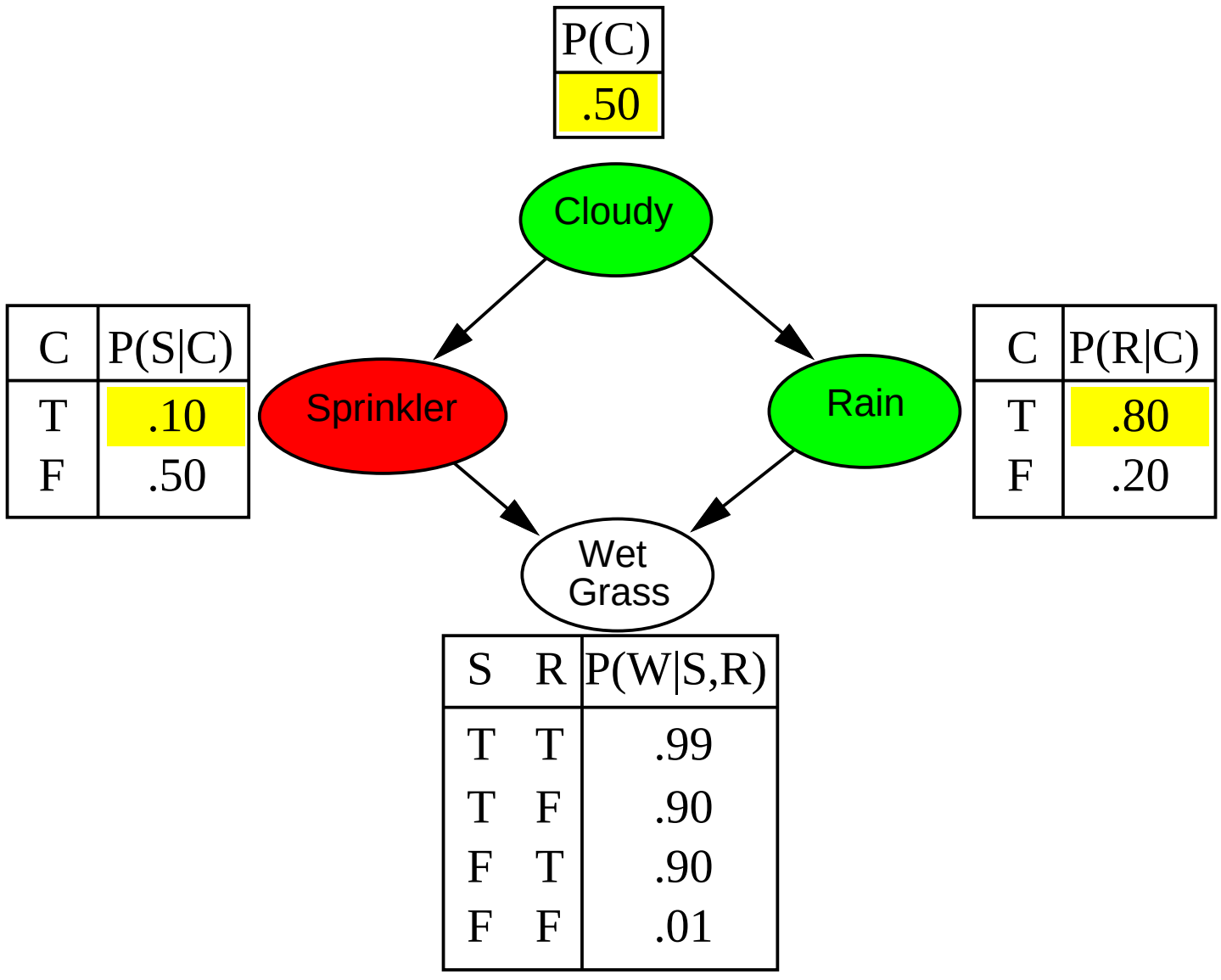
Example


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

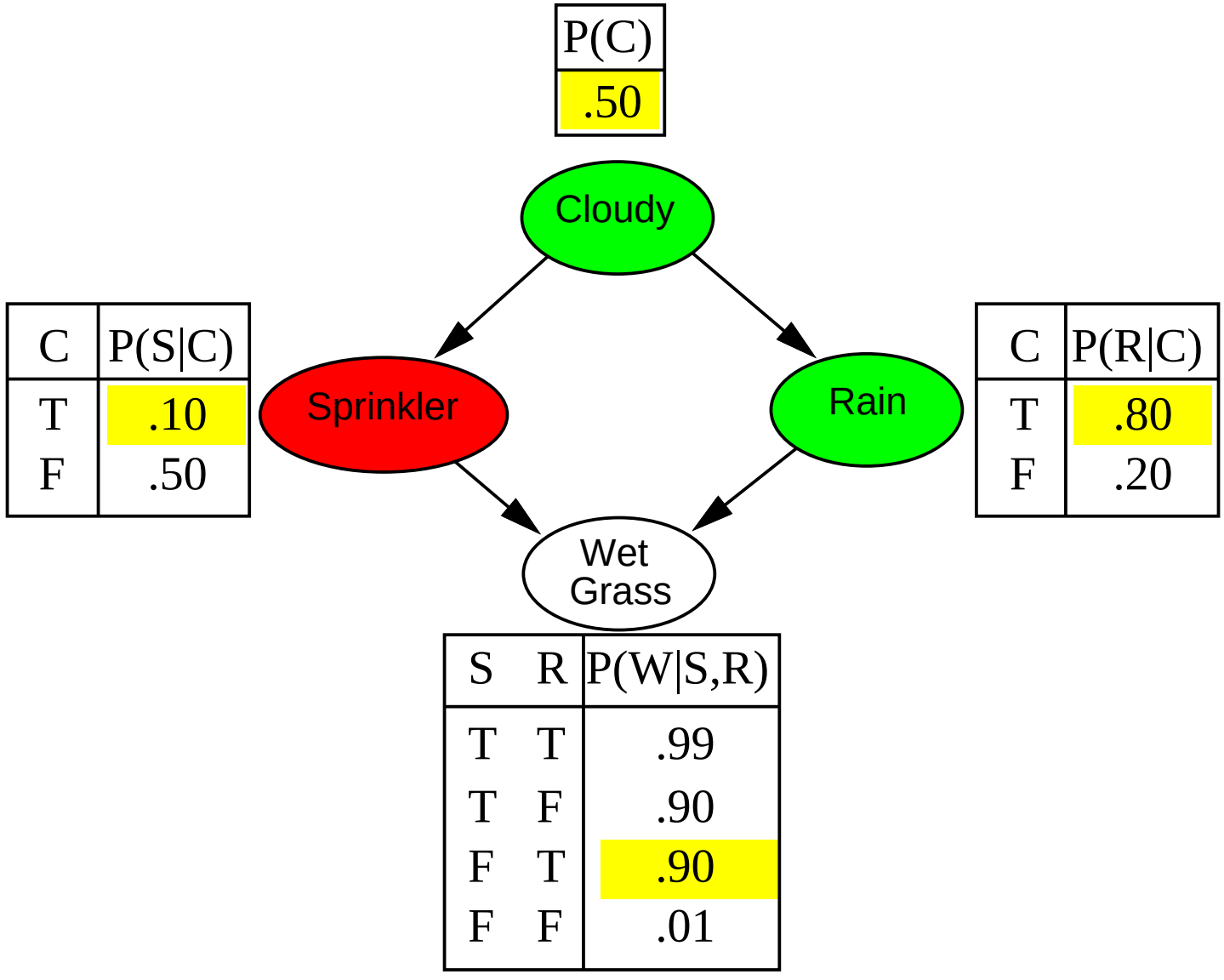
Example


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

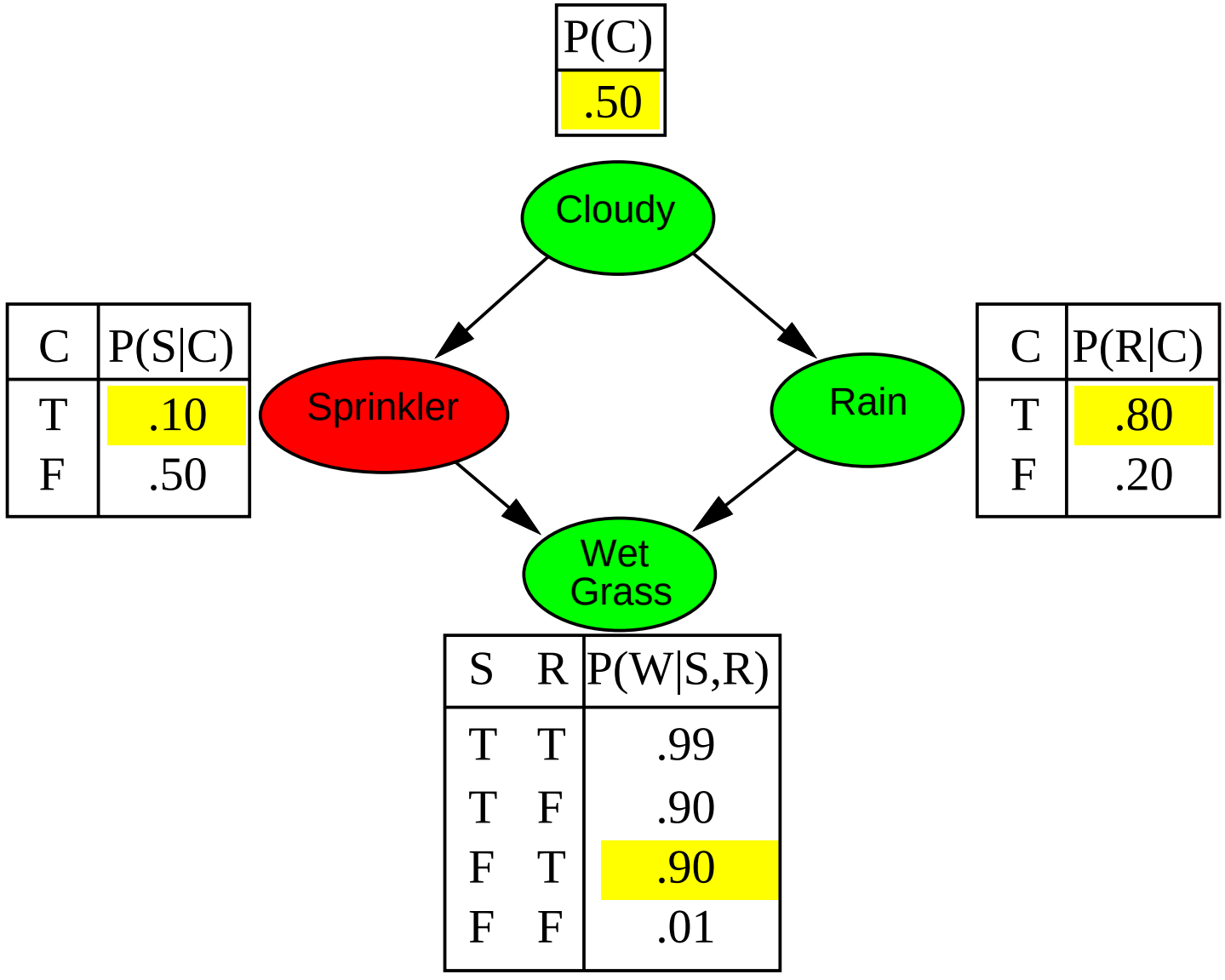
Example


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

Sampling from an empty network contd.

Probability that PRIORSAMPLE generates a particular event:

$$S_{PS}(x_1 \ldots x_n) = \prod_{i=1}^{n} P(x_i \mid Parents(X_i)) = P(x_1 \ldots x_n)$$

i.e., the true prior probability

E.g., $S_{PS}(t, f, t, t) = 0.5 \times 0.9 \times 0.8 \times 0.9 = 0.324 = P(t, f, t, t)$

Let $N_{PS}(x_1 \ldots x_n)$ be the number of samples generated for event $x_1, \ldots, x_n$. Then,

$$\begin{aligned} \lim_{N\to\infty} \hat{P}(x_1, \ldots, x_n) &= \lim_{N\to\infty} N_{PS}(x_1, \ldots, x_n)/N \\ &= S_{PS}(x_1, \ldots, x_n) \\ &= P(x_1 \ldots x_n) \end{aligned}$$

That is, estimates derived from PRIORSAMPLE are consistent

Shorthand: $\hat{P}(x_1, \ldots, x_n) \approx P(x_1 \ldots x_n)$

Rejection sampling

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

function REJ-SAMPLING(X, $\mathbf{e}$, bn, N) **returns** an estimate of $P(X|\mathbf{e})$
  local vars: $\mathbf{N}$, a vector of counts over X, initially zero
  for $j = 1$ to N **do**
    $\mathbf{x} \leftarrow$ PRIOR-SAMPLE(bn)
    if $\mathbf{x}$ is consistent with $\mathbf{e}$ **then**
      $\mathbf{N}[x] \leftarrow \mathbf{N}[x]+1$ where x is the value of X in $\mathbf{x}$
  return NORMALIZE($\mathbf{N}[X]$)

E.g., estimate $\mathbf{P}(Rain \mid Sprinkler = true)$ using 100 samples

27 samples have $Sprinkler = true$. Of these, 8 have $Rain = true$

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

Similar to a basic real-world empirical estimation procedure

Analysis of rejection sampling

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

Hence rejection sampling returns consistent posterior estimates

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

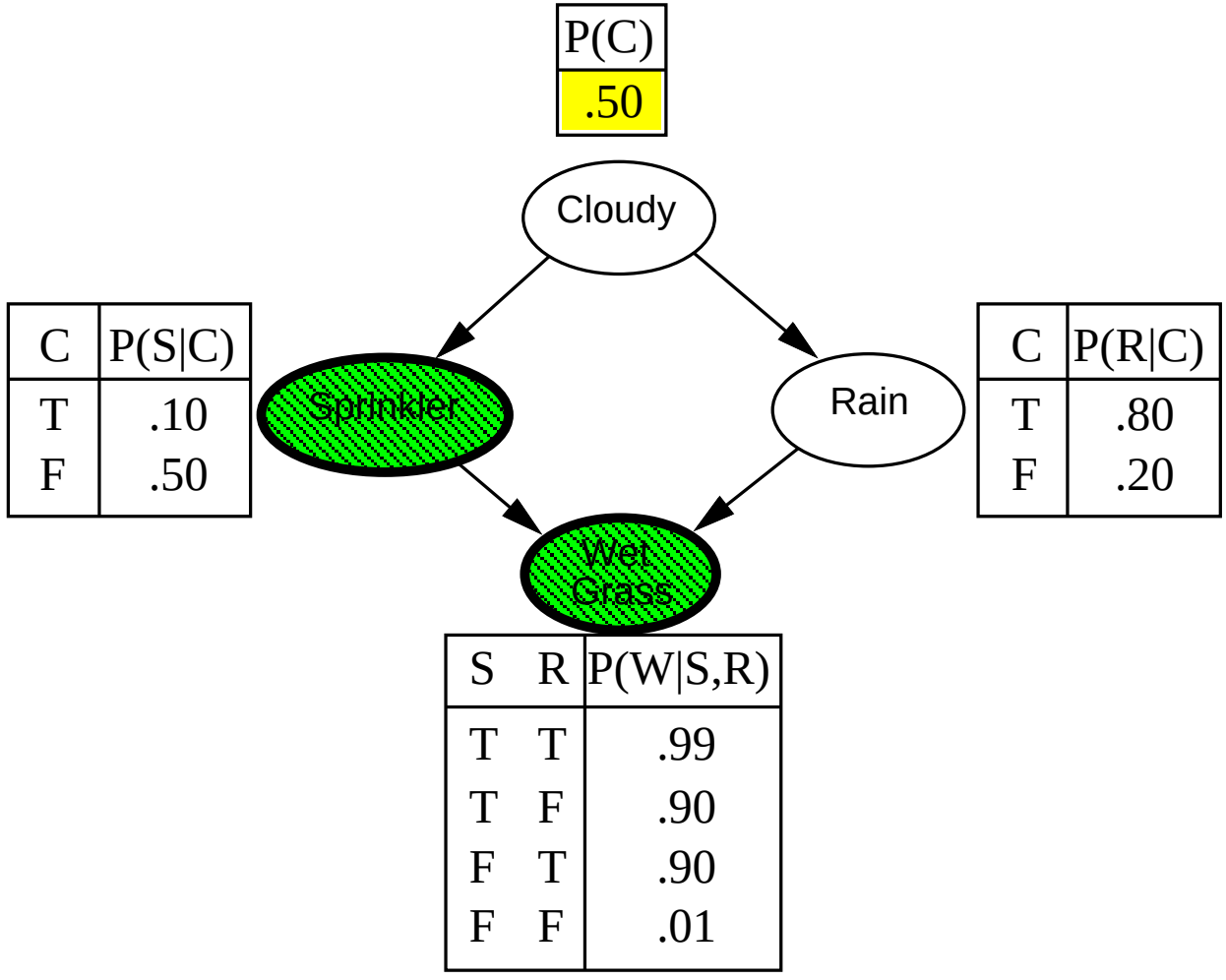
$P(\mathbf{e})$ drops off exponentially with number of evidence variables!

Likelihood weighting

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

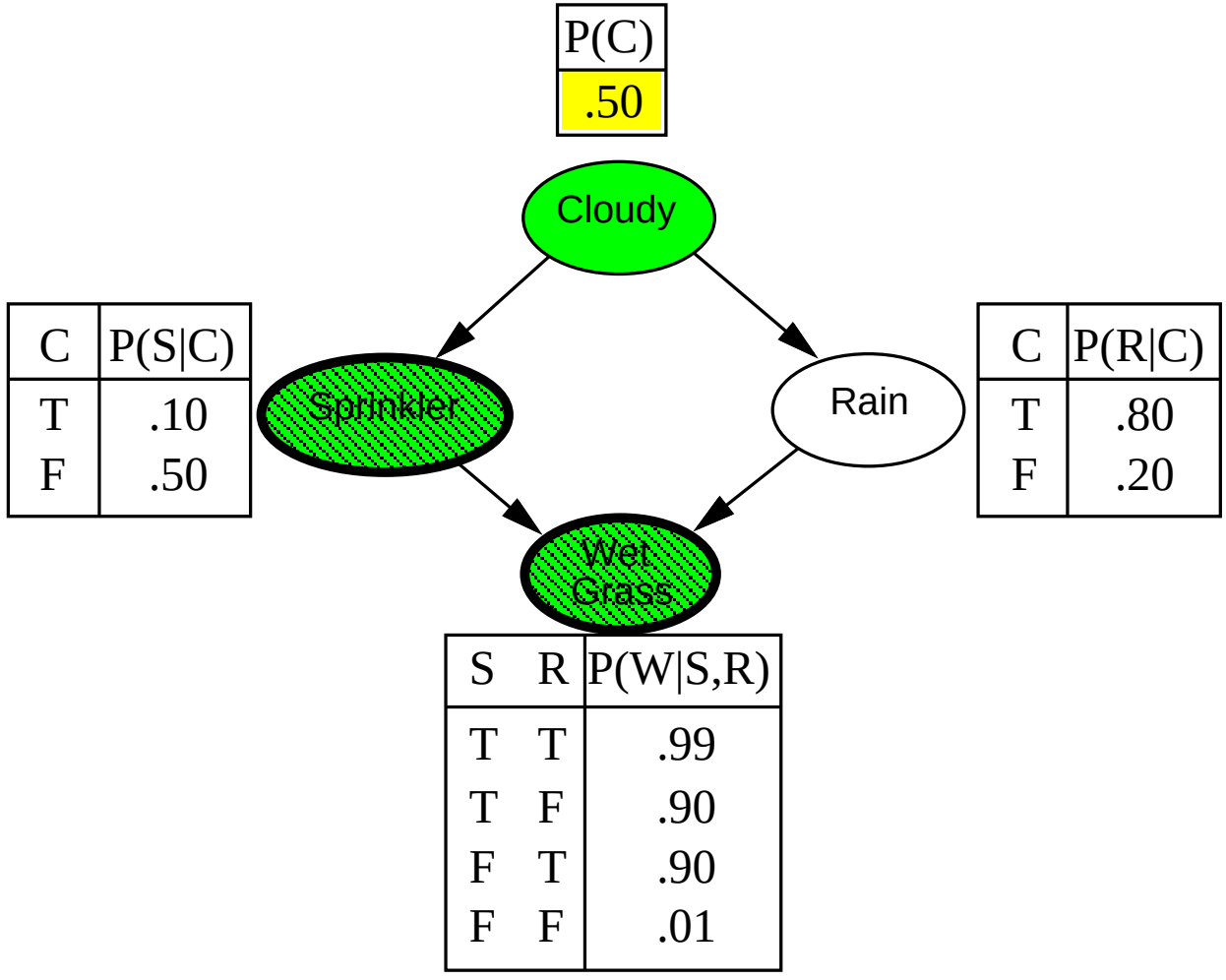
function LIKELYHOOD-W(X, **e**, bn, N) **returns** an estimate of $P(X|\mathbf{e})$
  local vars: **W**, a vector of weighted counts over X, initially zero
  for $j = 1$ to N **do**
    $\mathbf{x}, w \leftarrow$ WEIGHTED-SAMPLE(bn)
    $\mathbf{W}[x] \leftarrow \mathbf{W}[x] + w$ where x is the value of X in $\mathbf{x}$
  return NORMALIZE($\mathbf{W}[X]$)
function WEIGHTED-SAMPLE(bn, **e**) **returns** an event and a weight
  $\mathbf{x} \leftarrow$ 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 $\mathbf{x}$, w

Likelihood weighting example



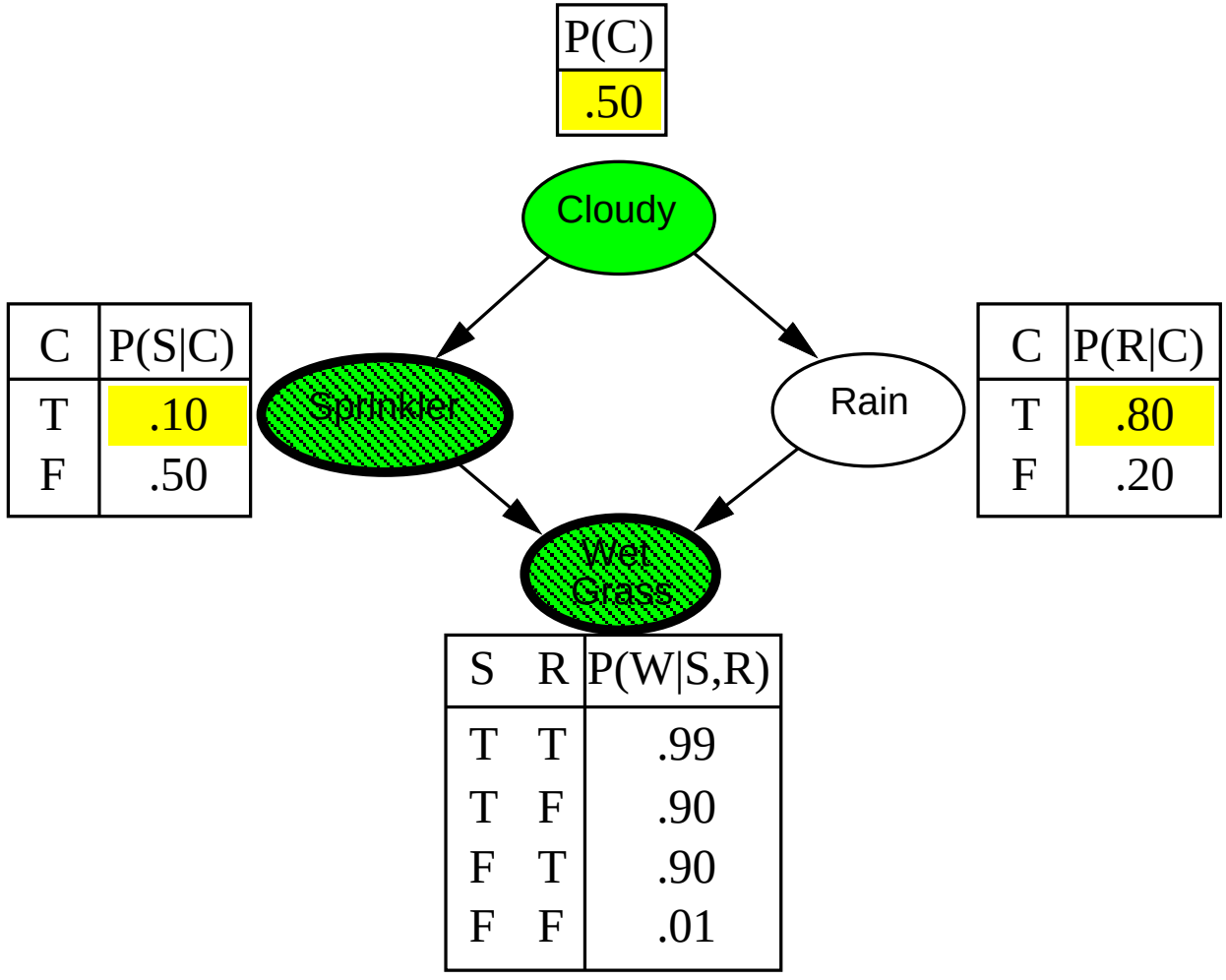
$w = 1.0$

Likelihood weighting example



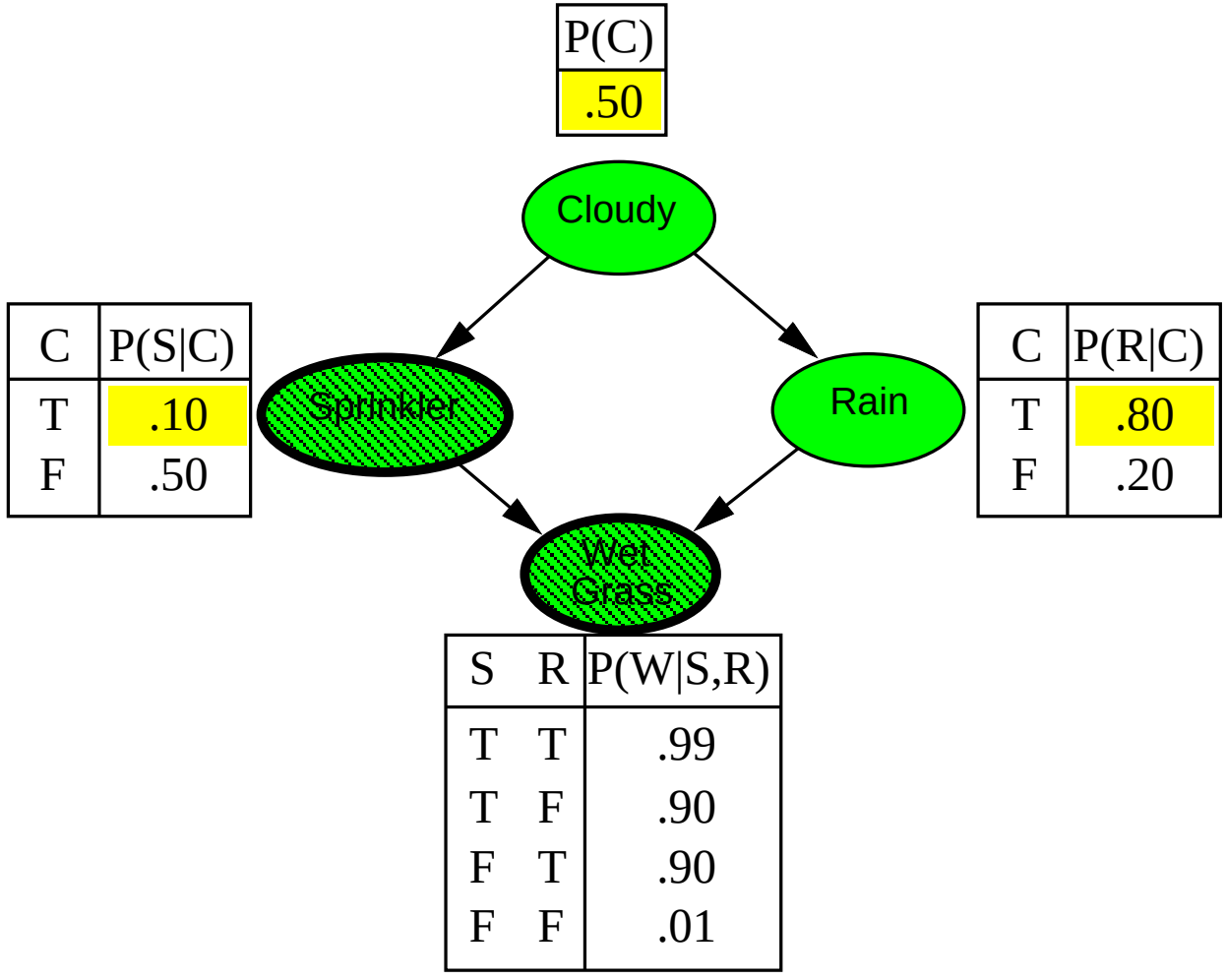
$w = 1.0$

Likelihood weighting example



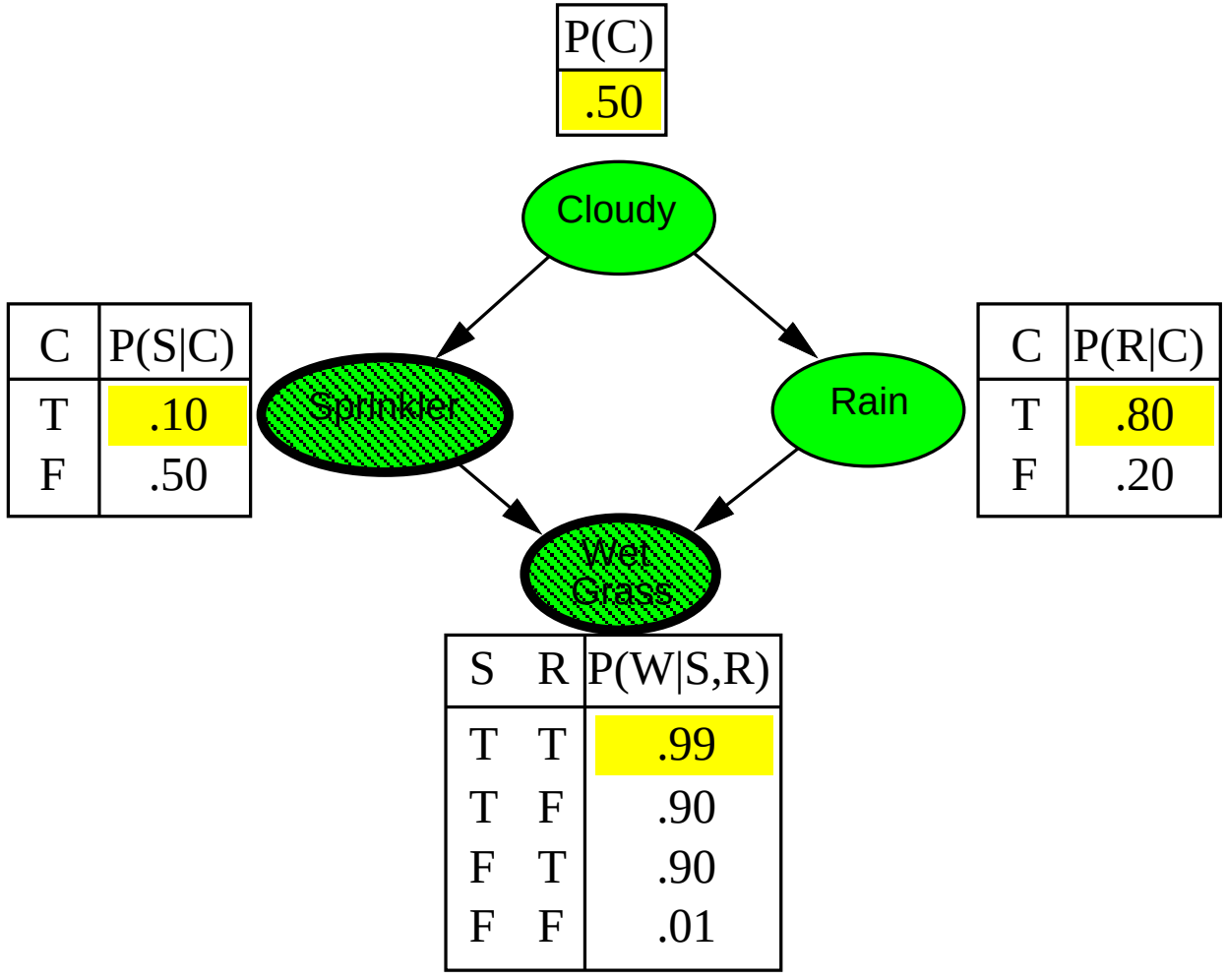
$w = 1.0 \times 0.1$

Likelihood weighting example



$w = 1.0 \times 0.1$

Likelihood weighting example



$w = 1.0 \times 0.1$

Likelihood weighting analysis

Sampling probability for WEIGHTED-xSAMPLE is

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

Note: pays attention to evidence in ancestors only $\Longrightarrow$ somewhere "in between" prior and posterior distribution

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

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

Weighted sampling probability is

$$\begin{aligned} &S_{WS}(\mathbf{z}, \mathbf{e}) w(\mathbf{z}, \mathbf{e}) \\ &\quad = \prod_{i=1}^{l} P(z_i \mid Parents(Z_i)) \prod_{i=1}^{m} P(e_i \mid Parents(E_i)) \\ &\quad = P(\mathbf{z}, \mathbf{e}) \text{ (by standard global semantics of network)} \end{aligned}$$

Hence likelihood weighting returns consistent estimates
but performance still degrades with many evidence variables
because a few samples have nearly all the total weight

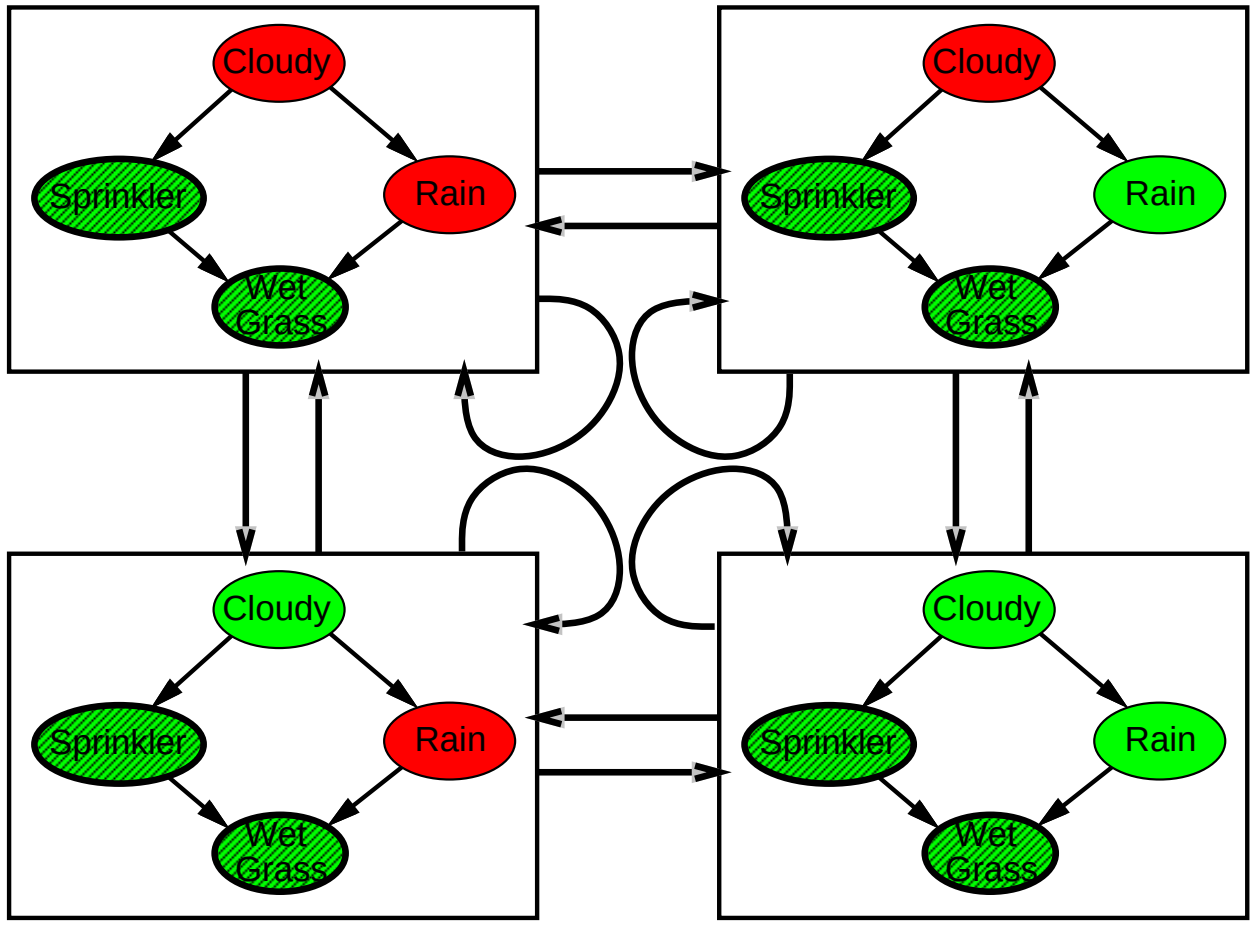
Approximate inference using MCMC

"State" of network = current assignment to all variables.
Generate next state by sampling one variable given Markov blanket
Sample each variable in turn, keeping evidence fixed

function MCMC-ASK(X, **e**, bn, N) **returns** an estimate of $P(X|\mathbf{e})$
  local vars: $\mathbf{N}[X]$, a vector of counts over X, initially zero
    $\mathbf{Z}$, nonevidence variables in bn
    $\mathbf{x}$, current state of the network, initially copied from $\mathbf{e}$
  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 Z_i in $\mathbf{Z}$ **do**
      sample the value of Z_i in $\mathbf{x}$ from $\mathbf{P}(Z_i|MB(Z_i))$
      given the values of $MB(Z_i)$ in $\mathbf{x}$
  return NORMALIZE($\mathbf{N}[X]$)

The Markov chain

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



Wander about for a while, average what you see

MCMC example contd.

Estimate $\mathbf{P}(Rain \mid sprinkler, wetGrass)$

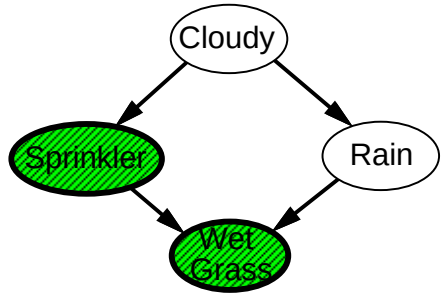
Sample $Cloudy$ or $Rain$ given its Markov blanket, repeat
Count number of times $Rain$ is true and false in the samples

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

$\hat{\mathbf{P}}(Rain \mid sprinkler, wetGrass) = \textsc{Normalize}(\langle 31, 69 \rangle)$
$= \langle 0.31, 0.69 \rangle$

Theorem: Chain approaches stationary distribution, i.e., long-run fraction of time spent in each state is exactly proportional to its posterior probability

Markov blanket sampling



Markov blanket of $Cloudy$ is $\{Sprinkler, Rain\}$
Markov blanket of $Rain$ is $\{Cloudy, Sprinkler, WetGrass\}$

Probability given the Markov blanket is calculated as
$P(x'_i \mid MarkovBlanket(X_i))$
$= P(x'_i \mid Parents(X_i)) \prod_{Z_j \in Children(X_i)} P(z_j \mid Parents(Z_j))$

Main computational problems:

1. Difficult to tell if convergence has been achieved
2. Can be wasteful if Markov blanket is large:
   $P(X_i \mid MarkovBlanket(X_i))$ will not change much
   (law of large numbers)