Overview

I Artificial Intelligence

II Problem Solving

III Knowledge, Reasoning, Planning

IV Uncertain Knowledge and Reasoning

- 12. Quantifying Uncertainty
- **13. Probabilistic Reasoning**
- 14. Probabilistic Reasoning over Time
- 15. Probabilistic Programming
- 16. Making Simple Decisions
- 17. Making Complex Decisions
- 18. Multiagent Decision Making

V Machine Learning

VI Communicating, Perceiving, and Acting

VII Conclusions

Probabilistic Reasoning

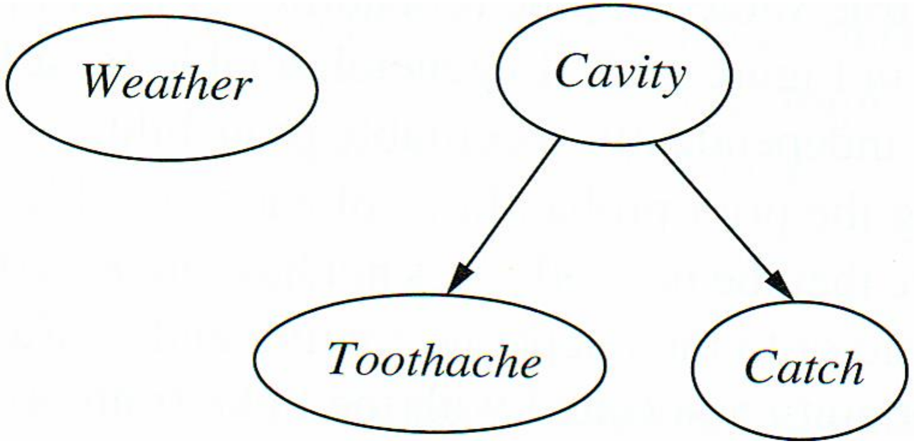
- Representing knowledge in an uncertain domain
- Semantics of Bayesian networks
- Exact inference in Bayesian networks
- Approximate inference for Bayesian networks
- Causal networks

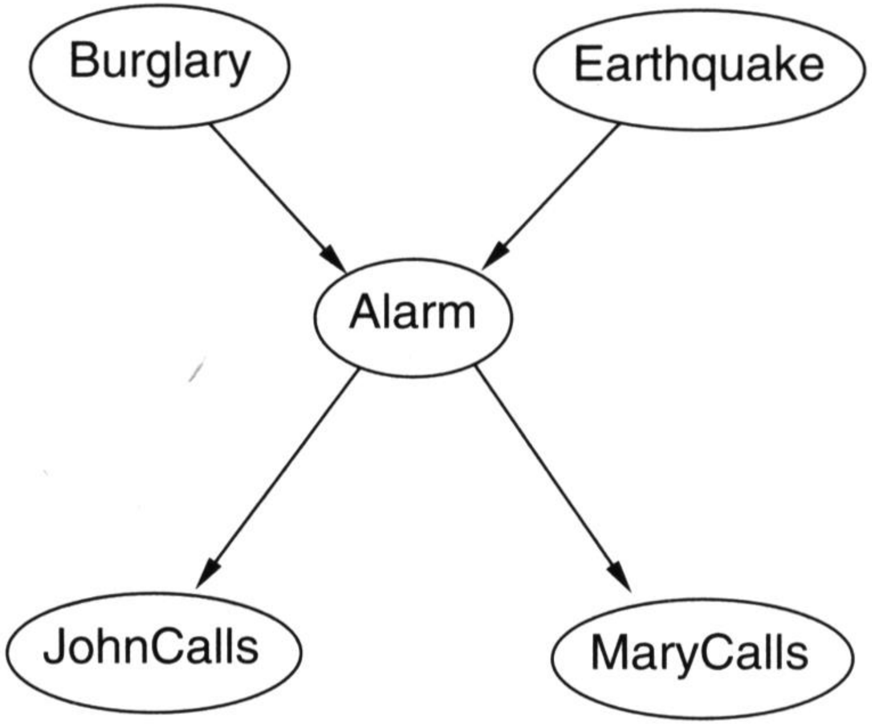
Representing knowledge in an Uncertain Domain

- A full joint probability distribution is sufficient to answer any questions about a domain
  - But for many variables too much probabilities necessary
- Strong independance assumption in Bayes rule greatly simplifies the task
  - But often oversimplification
- Good compromise: **Bayesisan networks** (Bayes net, belief network)
  - Can represent all dependencies among variables in a concise form

Bayesian Network

- Bayesian Network is a directed acyclic graph:
  - Each node represents a random variable (discrete or continuous)
  - Directed links (arrows) connecting nodes represent probabilistic relations between nodes
  - Each node X_i contains a conditional probability table $(X_i | Parents(X_i))$ quantifying the effect of its parents.
- Meaning of an arrow ($X \rightarrow Y$): X directly influences Y
  - Typically equivalent to cause-effect relations
  - However, cause-effect links are not enforced when constructing the net
- Example networks (without probability tables):
  - Toothache
  - Burglar alarm network in California

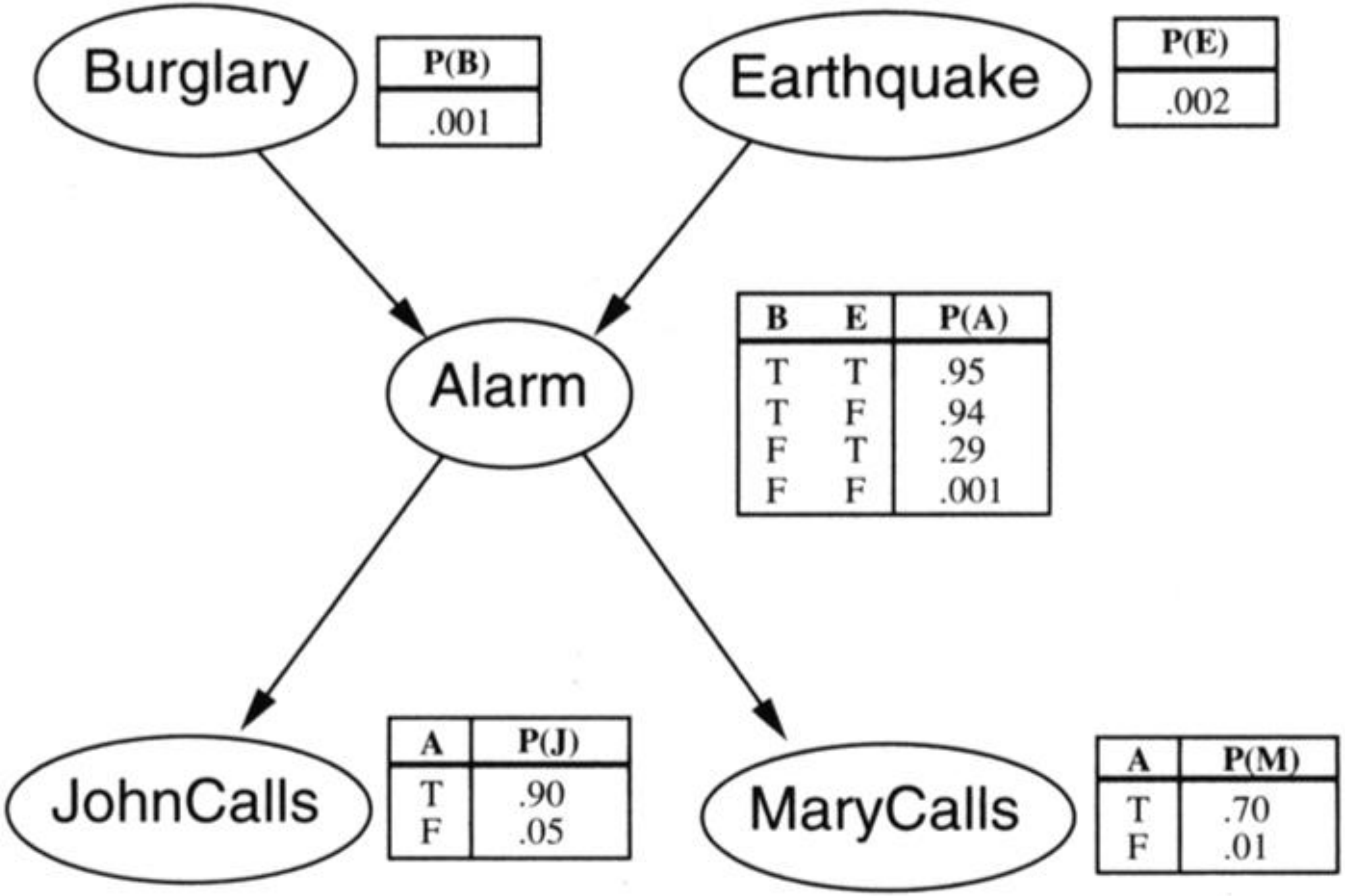




Full Example for Bayesian Network

An alarm (A) can be caused either by a burglary (B) or by an earthquake (E) and may be noticed by the neighbors John (J) or Mary (M) sending a message or making a phone call. The probability table for each variable either represents the apriori probability (B,E) or the conditional probability, given its parents (A,J,M).

In total, 10 parameters in truth tables necessary



P(B)
.001

P(E)
.002

B	E	P(A)
T	T	.95
T	F	.94
F	T	.29
F	F	.001

A	P(J)
T	.90
F	.05

A	P(M)
T	.70
F	.01

Semantics of Bayesian Networks

- From a Bayesian net each entry of the joint probability distribution can be computed:

$$P(x_1, ..., x_n) = P(X_1 = x_1 \wedge ... \wedge X_n = x_n) = \Pi^{n}_{i=1} \mathbf{P(\, x_i \mid Parents\,(X_i))}$$

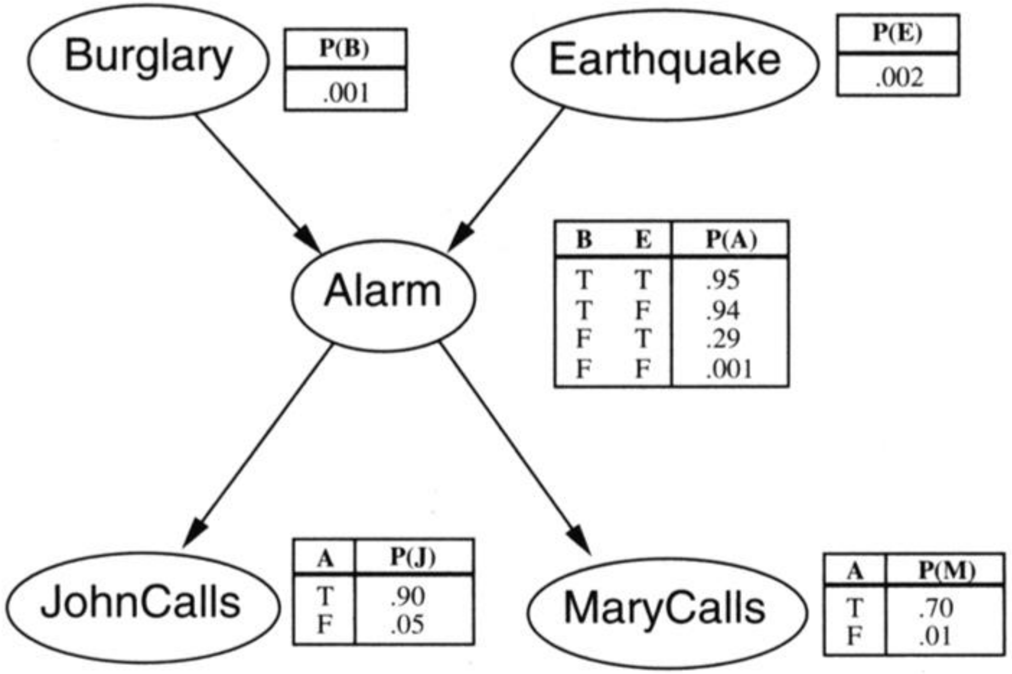
- Example: Probability of alarm without a burglary or earthquake, if both John and Mary call:

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

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

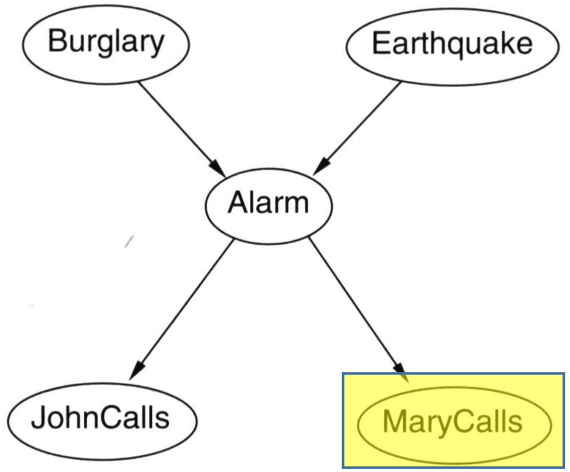
$0.9 * 0.7 * 0.001 * 0.999 * 0.998 =$

0.000628



Construction of Bayesian Nets (1)

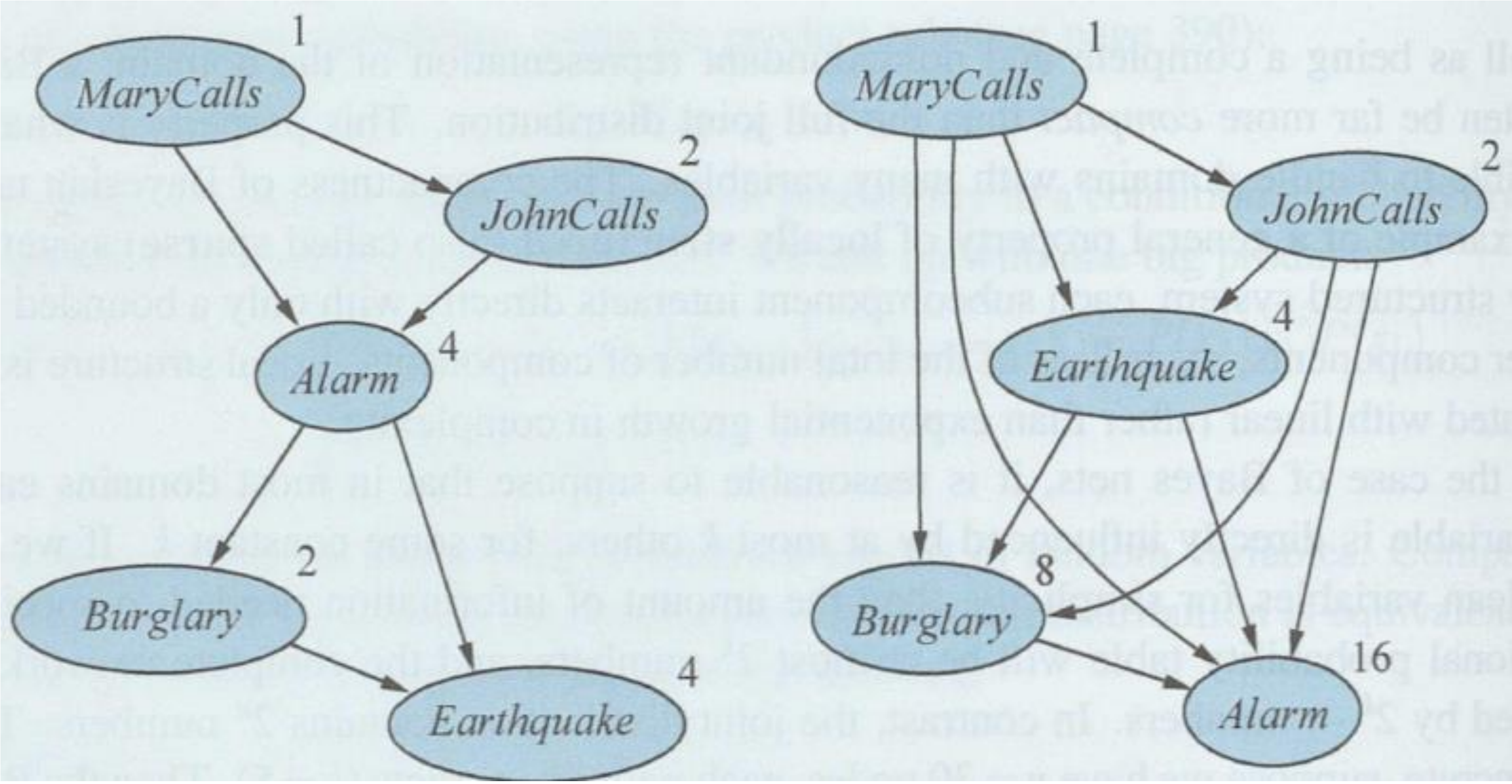
1. Determine the set of variables
2. Order them: Any order works, but if causes precedes effects, the net will be more compact
3. For each variable X_i in given order do:
   a) Choose a minimal set of parents for X_i from X_1 .. X_{i-1} that *directly* influence X_i
   b) Insert all links from each parent to X_i
   c) Define the conditional probability table $\mathbf{P}(X_i|\text{Parents}(X_i))$



- Example for step 3a:
  - Choice of parents for MaryCalls, when Burglary, Earthquake, Alarm and JohnCalls are already choosen
  - Mary cannot see the burglar, feel the earthquake or notice JohnCalls, but only senses the alarm, therefore: $P(M \mid J, A, B, E) = P(M \mid A)$.

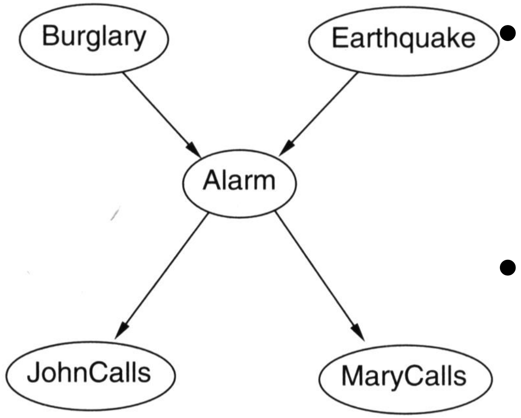
Compactness and Node Ordering

- Left: The network with node orderung M, J, A, B, E has 6 links and 13 parameters in the probability tables
- Right: The network with node orderung M, J, E, B, A has 10 links and 31 parameters
- For comparison: The „causal“ network with node ordering B, E, A, J, M has 4 links and 10 parameters

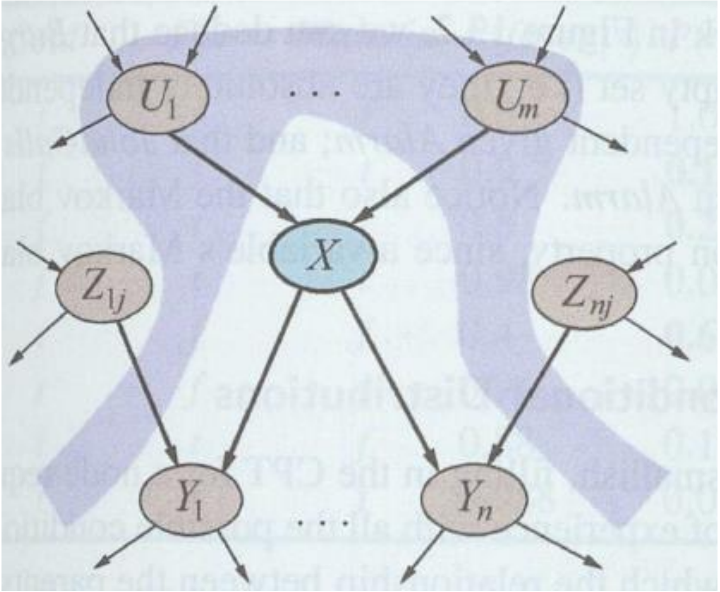


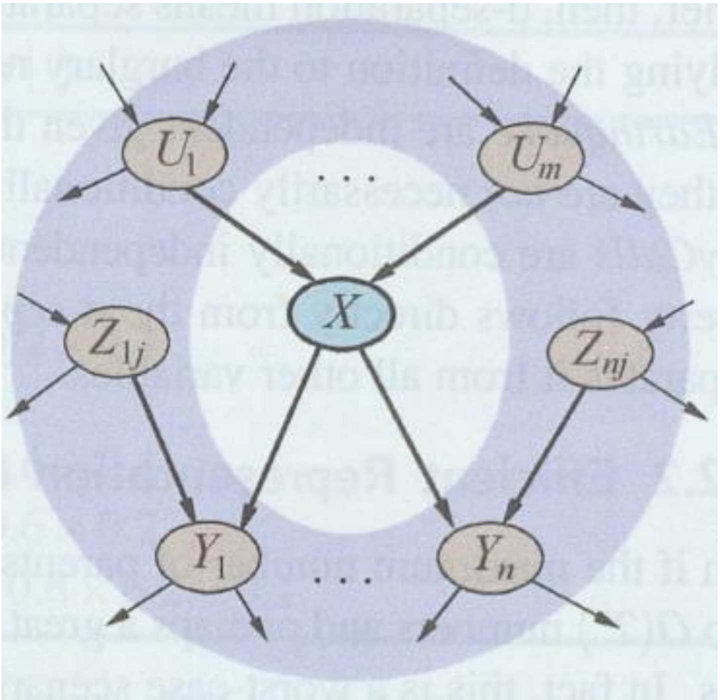
Steps in left network: 1) MaryCalls: No parents. 2) JohnCalls: More probable, if Mary calls therefore one parent. 3) Alarm: More probable, if either John or Mary calls, therefore two parents. 4) Burglary: More probable, if Alarm; no direct dependance on MaryCalls or JohnCalls; therefore only one parent. 5) Earthquake: Depends directly on Burglary and Alarm, therefore two parents

Conditional Independance Relations



- A variable is conditional independant of its predecessors, given its parents, e.g. J is conditional independant from B given A (left).
- A variable is conditional independant of its non-descendants, given its parents (see right above)
- A variable is conditional independant of all other nodes in the network, given its parents, children and children‘s parents (see right below, „Markov Blanket“)
- A set of nodes X is conditional independant to a set of nodes Y, given a third set Z, if Z d-separates X and Y (see next slide)

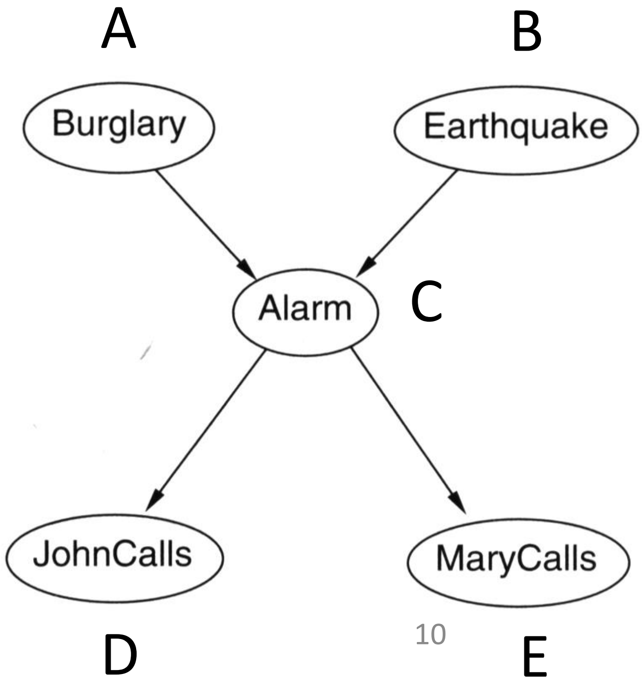




D-Separation

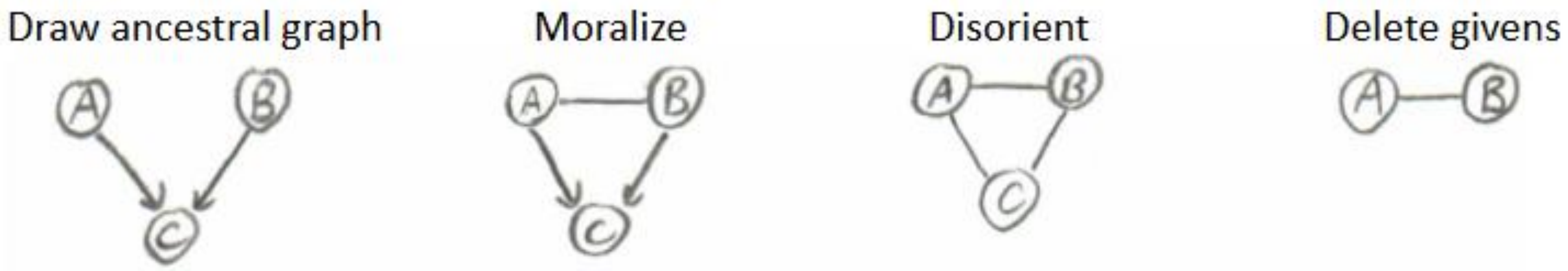
A set of nodes X is conditional independant to a set of nodes Y, given a third set Z, if Z d-seperates X and Y.

1. Consider just the ancestral subgraph consisting of X, Y, Z and their ancestors.
2. Add links between an unlinked pair of nodes that share a common child („marry them"), now we have the so-called moral graph („Moralize")
3. Replace all directed links by undirected links („Disorient")
4. If Z blocks all paths between X and Y, then Z d-separates X and Y („Delete givens")



Example from http://web.mit.edu/jmn/www/6.034/d-separation.pdf

- Are A and B conditionally independant, given C? – NO

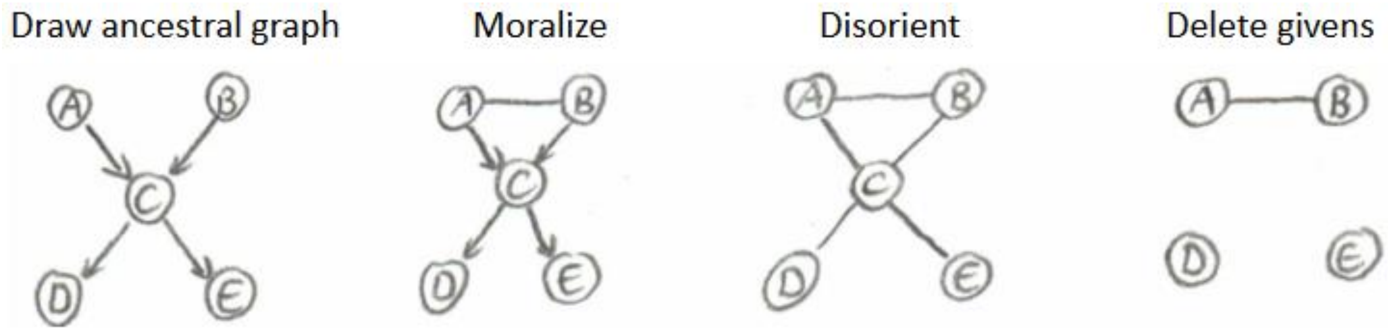


More Examples for D-Separation

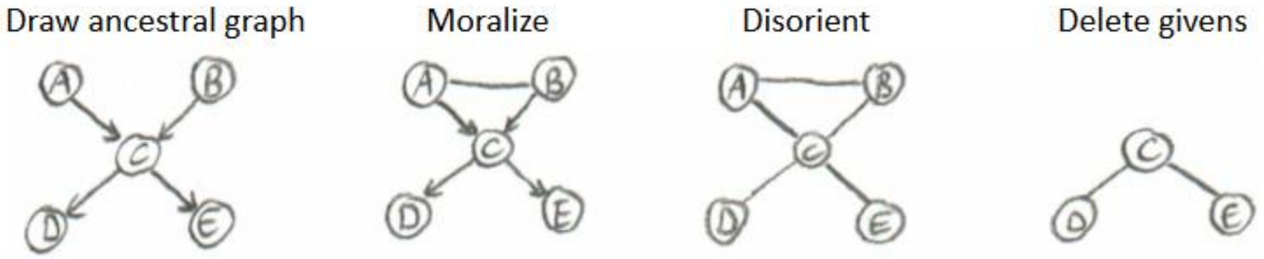
Further examples from http://web.mit.edu/jmn/www/6.034/d-separation.pdf

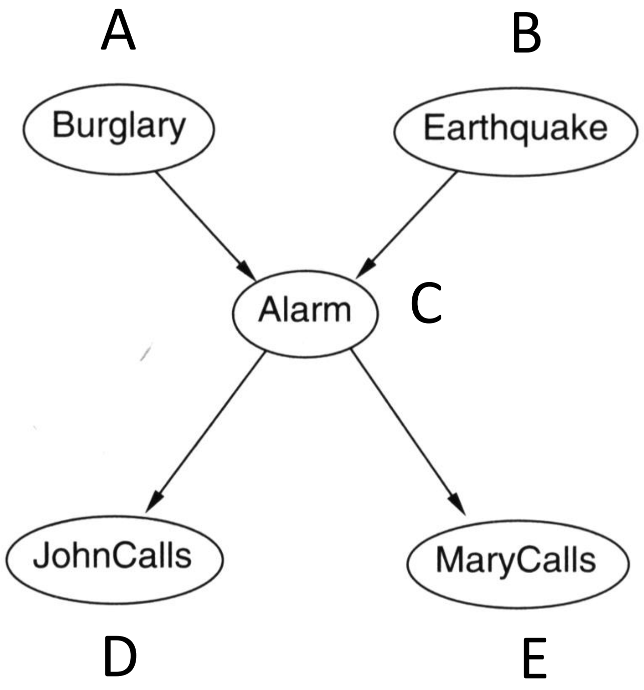
- Are A and B conditionally independant, given D? – NO
- Are A and B marginally independant (without knowing anything)? YES

- Are D and E conditionally independant, given C? – YES



- Are D and E marginally independant? – NO
- Are D and E conditionally independant, given A and B? – NO





Efficient Representation of Conditional Distributions

- Conditional probability tables (CPT) need 2^n entries for n parents
- However, often canonical distributions exist, requiring less entries:
  - **Deterministic distributions:** Value of node ist specified exactly by value of parents (e.g. best price for an item or membership of a person of a continent depending on membership of a country)
  - **Context-specific independance:** A variable is conditionally independant of some of its parents given certain values of other parents, e.g. damage of a car in a time period depends on ruggedness, accident and vandalism in that time period: If accident is false, ruggedness has no effect on damage.
  - Noisy logical relations like **noisy-OR relation**: If the parents of a node are conditional independant and complete, then only the individual parent-node probabilities are necessary and all combinations in the probability table can be inferred (see next slide).
    - Number of necessary independant parameters in probability table decreases from $O(2^n)$ to $O(n)$ with n parents

Example for Noisy-Or Relation

- Assumptions:
  - All causes of fever are known (if necessary a leak-node „Other“ can be added)
  - The three causes of fever (Cold, Flu, Malaria) are independant from each other
- Compute the combined negative probabilities from the negated basic probabilities (in bold):

Cold	*Flu*	*Malaria*	$P(fever \mid \cdot)$	$P(\neg fever \mid \cdot)$
f	*f*	*f*	0.0	1.0
f	*f*	*t*	0.9	**0.1**
f	*t*	*f*	0.8	**0.2**
f	*t*	*t*	0.98	$0.02 = 0.2 \times 0.1$
t	*f*	*f*	0.4	**0.6**
t	*f*	*t*	0.94	$0.06 = 0.6 \times 0.1$
t	*t*	*f*	0.88	$0.12 = 0.6 \times 0.2$
t	*t*	*t*	0.988	$0.012 = 0.6 \times 0.2 \times 0.1$

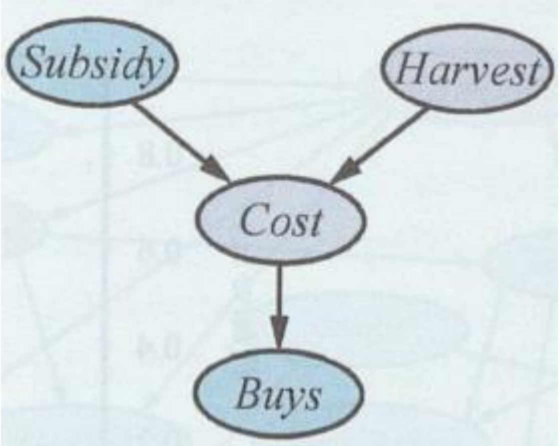
Bayesian Nets with Continuous Variables

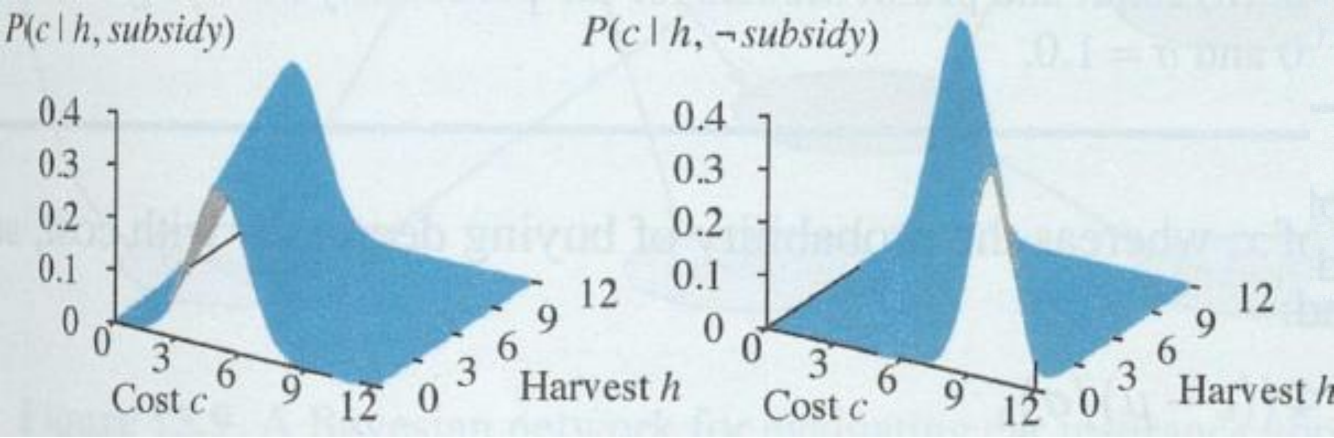
- Problem:
  - Conditional probability tables require discrete value
  - But: Many applications have continuous values
- Possible Solutions (compare last chapter for naive Bayes)
  - Discretization of continuous variables
  - Definition of probability density functions, specified by parameters, e.g. for linear Gaussian distribution mean μ and standard deviation σ

Bayesian Nets with Continuous Variables

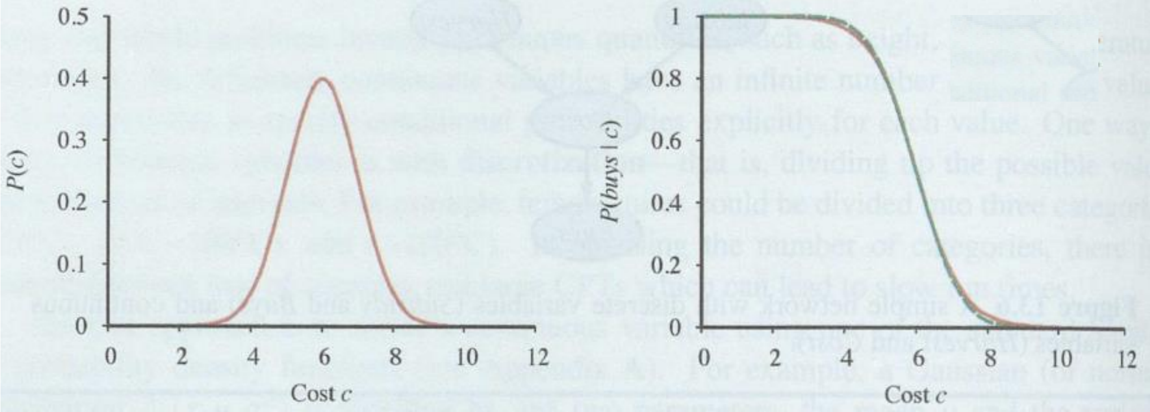
- With combinations of discrete and continuous variables, for each combination of discrete variables a different (linear) Gaussian distribution is necessary:
  - Example:

Simple network with discrete variables (Subsidy, Buys) and continuous variables (Harvest, Cost) and the resulting distributions





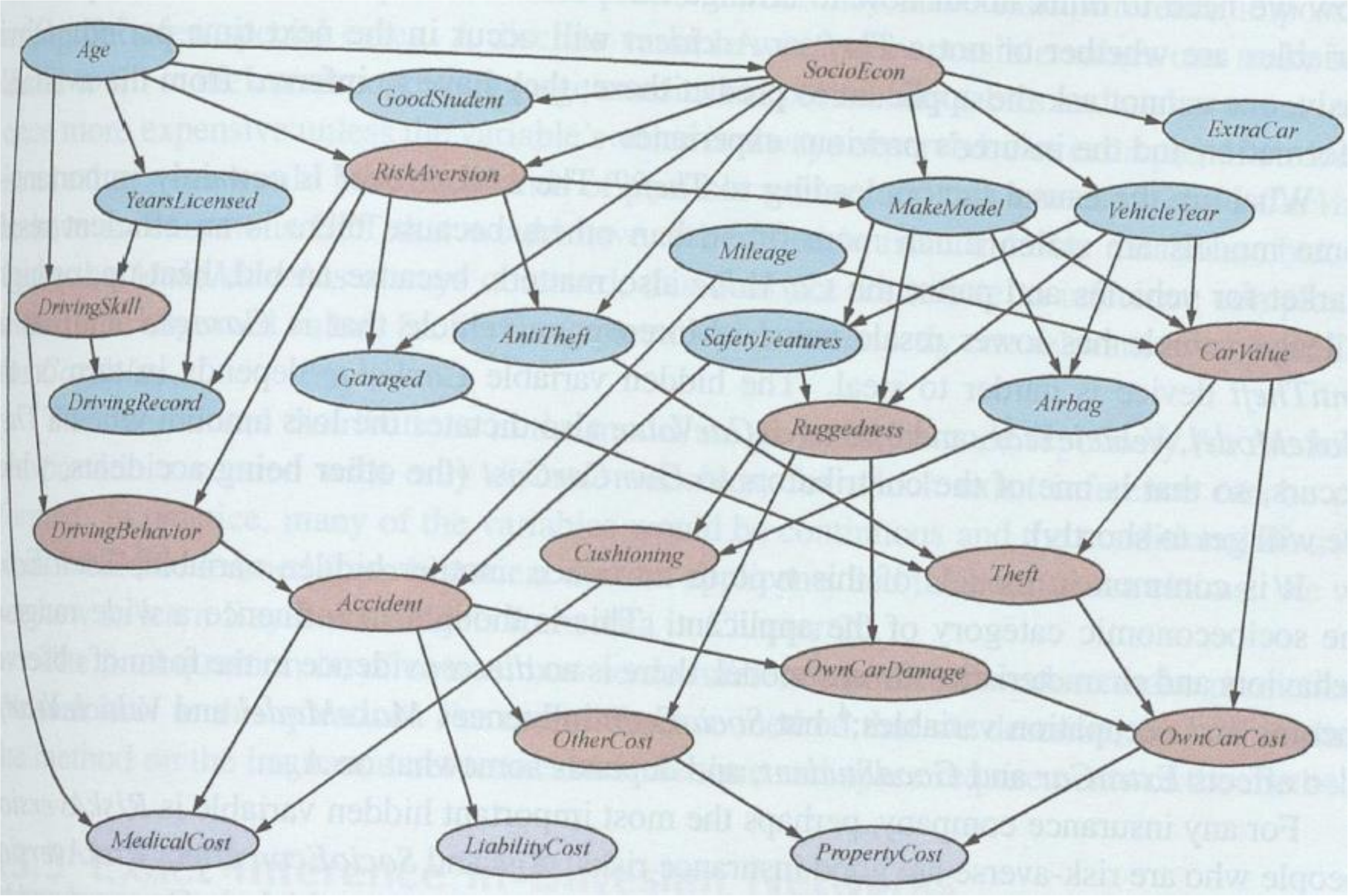
- To infer a discrete decision from continuous parents, a threshold function is necessary
  - Example: A customer will buy with higher probability, if the costs are lower.
  - A soft threshold can be computed using the integral of the standard normal distribution



Case Study: Car insurance

Car insurance network:

- Blue nodes: Input information about the applicant (age, YearsLicensed, i.e duration of having a driver licence, etc.), the vehicle (e.g. MakeMode, VehicleYear, etc.) and the driving situation (e.g. Mileage, Garaged, etc.)
- Brown nodes: Hidden Variables
- Lavender nodes (below): Output variables (MedicalCost, LiabilityCost, PropertyCost)



Exact Inference in Bayesian Networks

- Rep.: Inference procedure for posterior probability of a query variable X, given some observed evidence variables E with value e, while other variables Y are unobserved:

$$\mathbf{P}(X|\mathbf{e}) = \alpha\, \mathbf{P}(X,\mathbf{e}) = \alpha \sum_{\mathbf{y}} \mathbf{P}(X,\mathbf{e},\mathbf{y})$$

- Rep.: In Bayesian nets the complete probability distribution P(X,e,y) can be written as:

$$P(x_1, \ldots, x_n) = \prod^{n}_{i=1} P(x_i \mid Parents\ (X_i))$$

➢ Therefore, a query can be answered using a Bayes net by computing sums of products of conditional probabilities from the network

  ➢ „**Enumerate-Joint-Ask**“ algorithm

Example for Enumerate-Joint-Ask

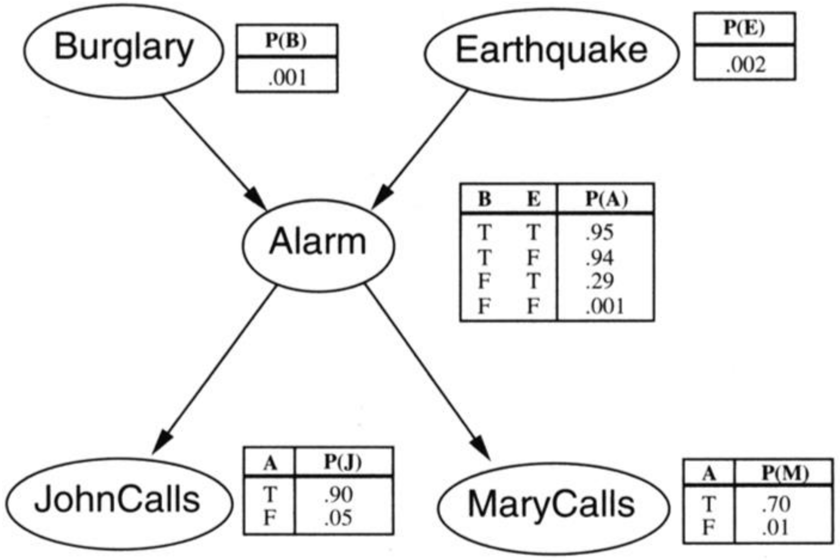
- P(Burglary|JohnCalls=true, MaryCalls=true)
- $P(B|j,m) = \alpha\, P(B,j,m) = \alpha \sum_e \sum_a P(B,e,a,j,m)$
  - For Burglary = true and false:

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

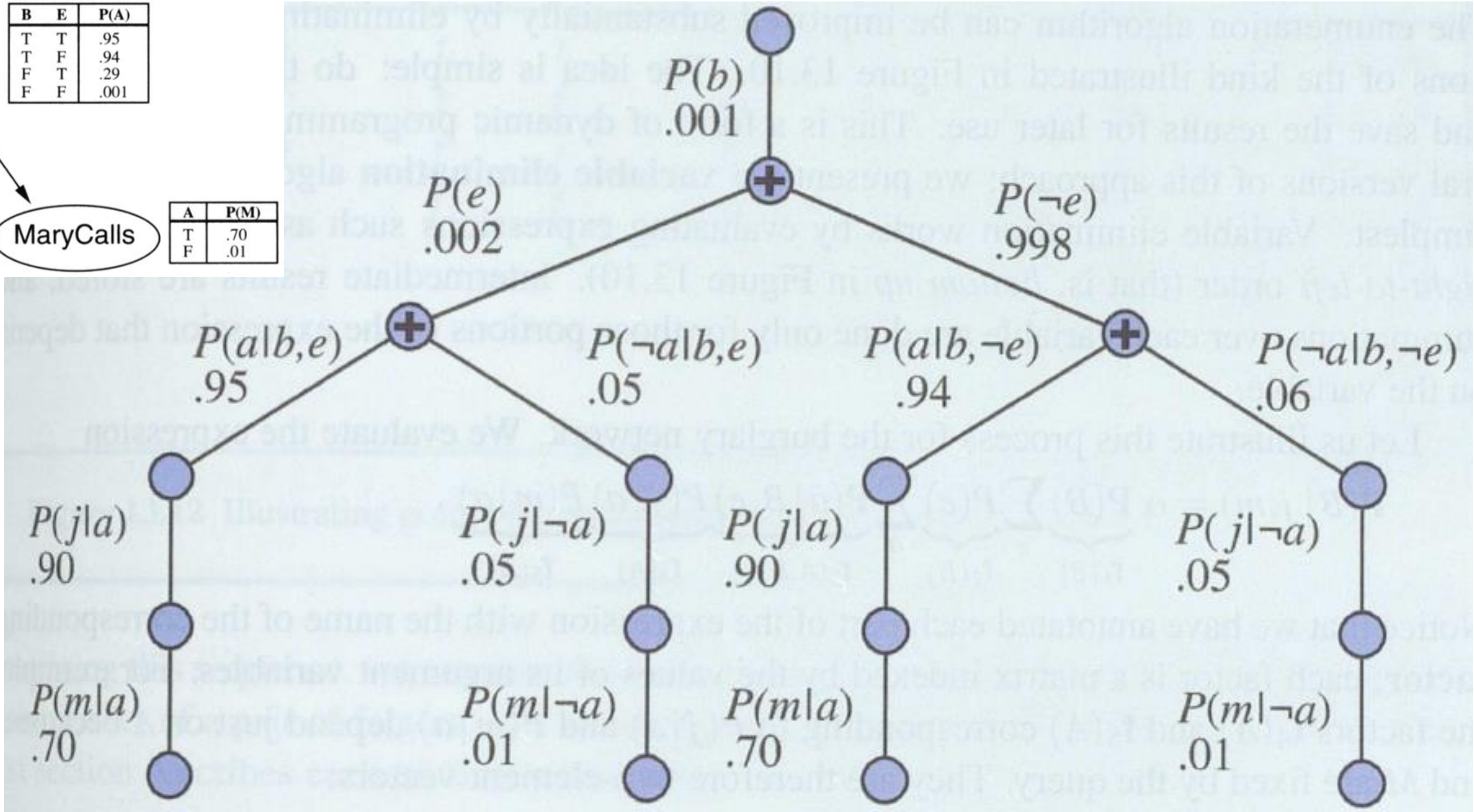
    $\alpha \sum_e \sum_a P(\neg b)\, P(e)\, P(a|\neg b,e)\, P(j|a)\, P(m|a)$
  - To compute these terms, we have to add four terms (because of $\sum_e \sum_a$), each computed by multiplying five numbers.
- Complexity with n boolean variables: $\mathbf{O(n2^n)}$
- Improvement: Extraction of constants and moving summations inwards as far as possible:

  $\alpha\, P(b) \sum_e P(e) \sum_a P(a|b,e)\, P(j|a)\, P(m|a)$ // for Burglary = true
- Result: $P(b|j,m) = \alpha * 0{,}00059$; $P(\neg b|j,m) = \alpha * 0{,}0015$,
  - $P(B|j,m) = \alpha \langle 0{,}00059, 0{,}0015\rangle \approx \langle 0{,}28, 0{,}72\rangle$
  - i.e. P (Burglary, if John and Mary call) = 28%

Structure of the Expression Tree from Example



Some branches (at j and m) are still computed two times.



Enumerate-Joint-Ask Algorithm

function ENUMERATION-ASK(X, **e**, bn) **returns** a distribution over X
  inputs: X, the query variable
    e, observed values for variables **E**
    bn, a Bayes net with variables $vars$

  $\mathbf{Q}(X) \leftarrow$ a distribution over X, initially empty
  for each value x_i of X **do**
    $\mathbf{Q}(x_i) \leftarrow$ ENUMERATE-ALL($vars$, $\mathbf{e}_{x_i}$)
      where $\mathbf{e}_{x_i}$ is **e** extended with $X = x_i$
  return NORMALIZE($\mathbf{Q}(X)$)

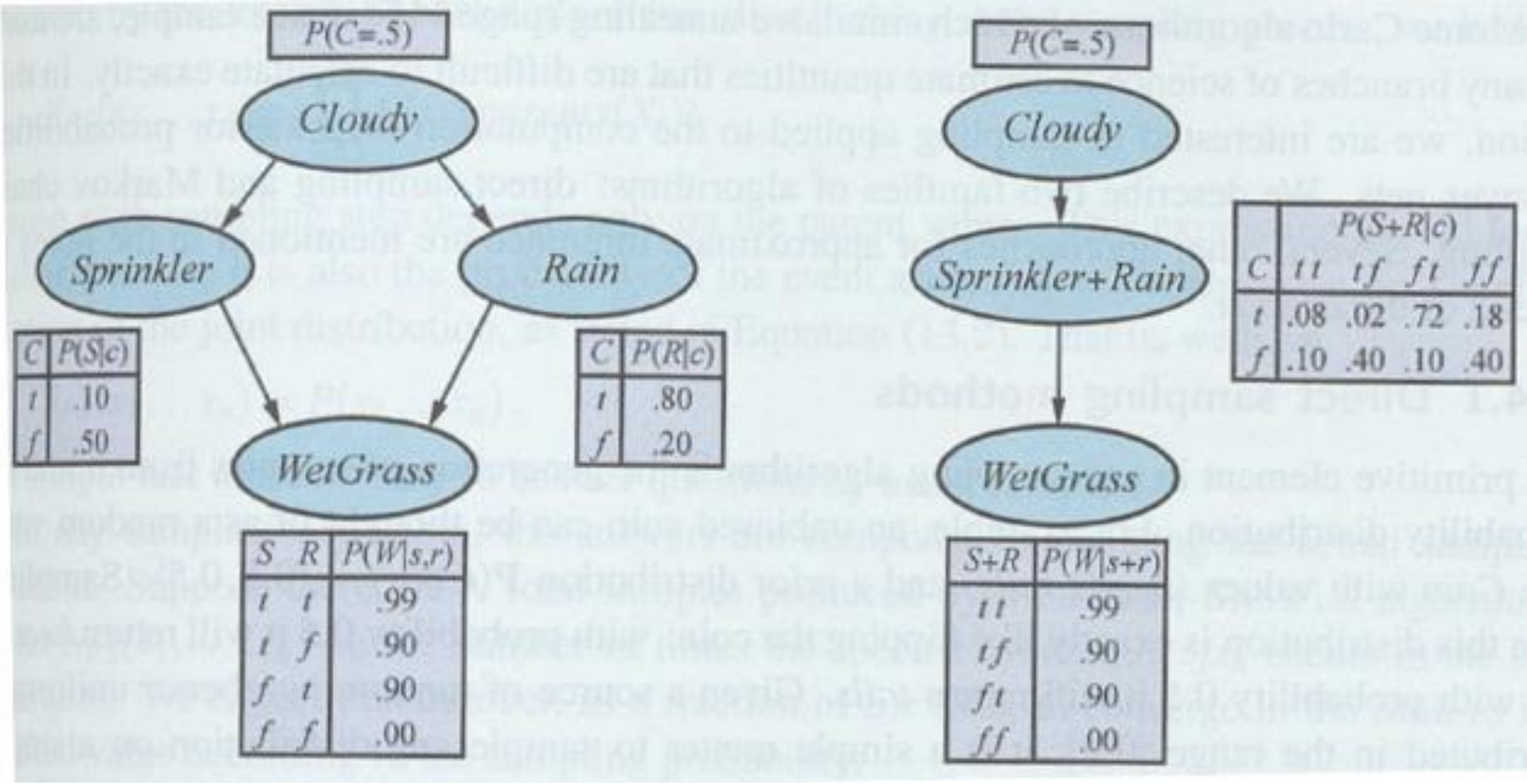
function ENUMERATE-ALL($vars$, **e**) **returns** a real number
  if EMPTY?($vars$) **then return** 1.0
  $V \leftarrow$ FIRST($vars$)
  if V is an evidence variable with value v in **e**
    then return $P(v \mid parents(V)) \times$ ENUMERATE-ALL(REST($vars$), **e**)
    else return $\sum_v P(v \mid parents(V)) \times$ ENUMERATE-ALL(REST($vars$), $\mathbf{e}_v$)
      where $\mathbf{e}_v$ is **e** extended with $V = v$

Improvements for Enumerate-Joint-Ask

- Ideas:
  - Move terms in the sums forward
  - Perform all computations just once and save them
  - Recognize and eliminate irrelevant variables, whose sum in a product equals 1 (e.g. $\sum_m P(m|a)$ in a query P (J | B = true)
- Many implementations, e.g. variable elimination (s. textbook)
- Complexity of variable elimination:
  - With singly connected networks (polytrees): O(n)
  - With multiply connected networks: exponential (as expected, since propositional logic can be imitated)

Clustering Algorithms

- Multiply connected networks can be transformed in singly connected networks by clustering nodes to „meganodes“
- Clustering might also be helpful to answer several queries in one call of the algorithm
- Left: Multiply connected network
- Right: Equivalent singly connected network by joining the nodes Sprinkler and Rain to a meganode „Sprinkler+Rain“ with a larger probability table
- Special purpose inference algorithm necessary (because of constraints on sharing variables in several meganodes)



Approximate Inference for Bayesian Networks

- **Idea:** For a query, perform many simulation runs with random numbers based on the probabilities in the Bayesian net and count the different answers to the query.
  - Also called **Monte Carlo** algorithms
  - With enough samples, the true probability can be approximated arbitrary close
    - Exception: Bayesian nets with deterministic conditional distributions
- Approaches for Bayesian Nets:
  - Direct sampling
  - Rejection sampling
  - Importance sampling, e.g. likelihood weighting
  - Markov chain simulation, e.g. Gibbs sampling

Sampling Approaches

Choose values for the nodes in topological order in accordance with their probability tables and the values of the parents (with a random number generator) und iterate as often, until an answer to the query is approximated accurate enough.

- **Direct sampling:** For rare events many runs are necessary

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

- **Rejection sampling:** All runs irrelevant for the query are immediately discarded (if an evidence variable get a wrong value)
- **Likelihood weighting** (subtype of importance sampling): The evidence variables e are preset and the runs are weighted according to the probability of setting e depending on their parents

Likelihood Weighting

- Each run (sample) gets a weight w by multiplying the weights from each evidence variable computed from $P(X_i | parents(X_i))$
- The results of the runs for the query variable X are accumulated based on their weights

function LIKELIHOOD-WEIGHTING(X, **e**, bn, N) **returns** an estimate of $\mathbf{P}(X | \mathbf{e})$
  inputs: X, the query variable
    e, observed values for variables **E**
    bn, a Bayesian network specifying joint distribution $\mathbf{P}(X_1, \ldots, X_n)$
    N, the total number of samples to be generated
  local variables: **W**, a vector of weighted counts for each value of X, initially zero

  for $j = 1$ **to** N **do**
    x, $w \leftarrow$ WEIGHTED-SAMPLE(bn, **e**)
    $\mathbf{W}[j] \leftarrow \mathbf{W}[j] + w$ where x_j is the value of X in **x**
  return NORMALIZE(**W**)

function WEIGHTED-SAMPLE(bn, **e**) **returns** an event and a weight

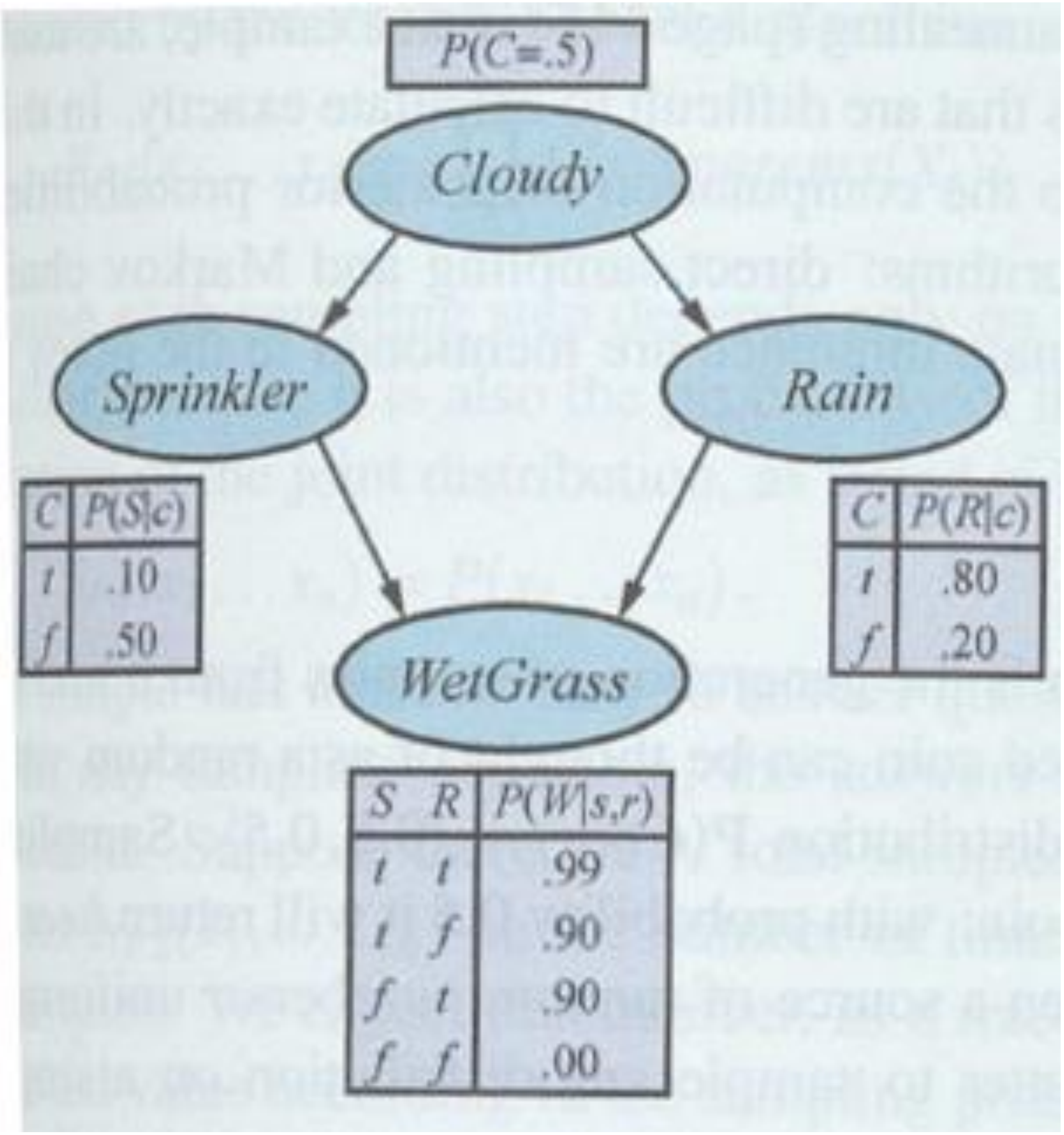
  $w \leftarrow 1$; $\mathbf{x} \leftarrow$ an event with n elements, with values fixed from **e**
  for $i = 1$ **to** n **do**
    if X_i is an evidence variable with value x_{ij} in **e**
      then $w \leftarrow w \times P(X_i = x_{ij} | parents(X_i))$
      else $\mathbf{x}[i] \leftarrow$ a random sample from $\mathbf{P}(X_i | parents(X_i))$
  return **x**, w

Example for Likelihood Weighting

- Query: **P**(Rain|Cloudy=true, WetGrass=true)
- Showing one run (function „Weighted-Sample“) in topological order:

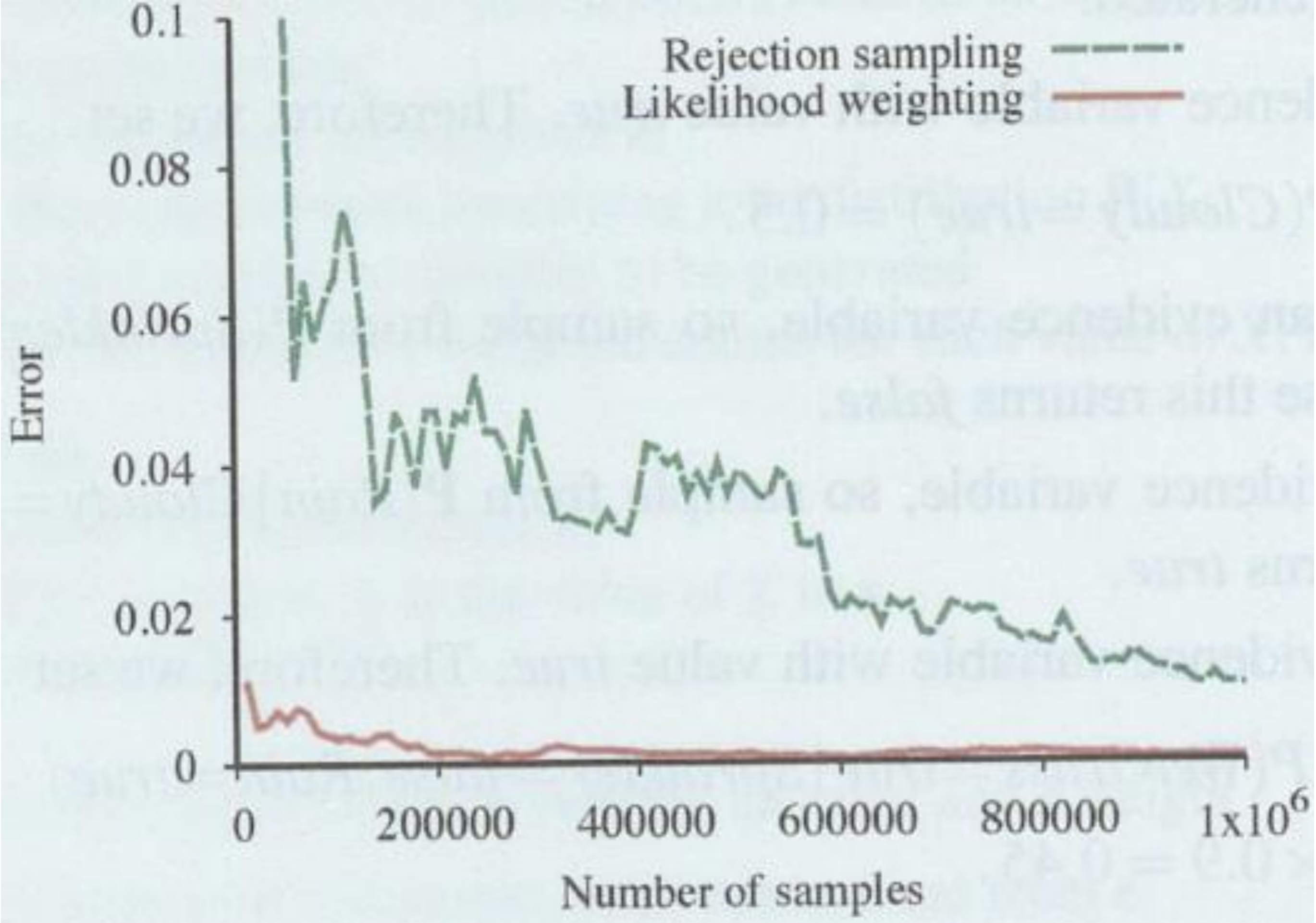
1. Cloudy is an evidence variable with value true. Setting Cloudy=true is weighted with probability 0.5, thus w = 1 * 0.5 = 0.5
2. Sprinkler is not an evidence variable, it is sampled from ⟨0.1, 0.9⟩, suppose it returns false
3. Rain is not an evidence variable, it is sampled from ⟨0.8, 0.2⟩, suppose it returns true
4. WetGrass is an evidence variable with value true. Setting WetGrass=ture is weighted with probability 0.9, thus w = 0.5 * 0.9 = 0.45

- This run yields Rain=true with weight 0.45



Efficiency of Likelihood Weighting

- In comparison to rejection sampling, likelihood weighting is usually much more efficient, because each run counts (left: for car insurance network)
  - With many evidence variables the performance degradates, because each run will have very low weights
    - In particular, if the evidence variables are in the lower part of the network („downstream“), because then the non-evidence variables in the upper part cannot be guided



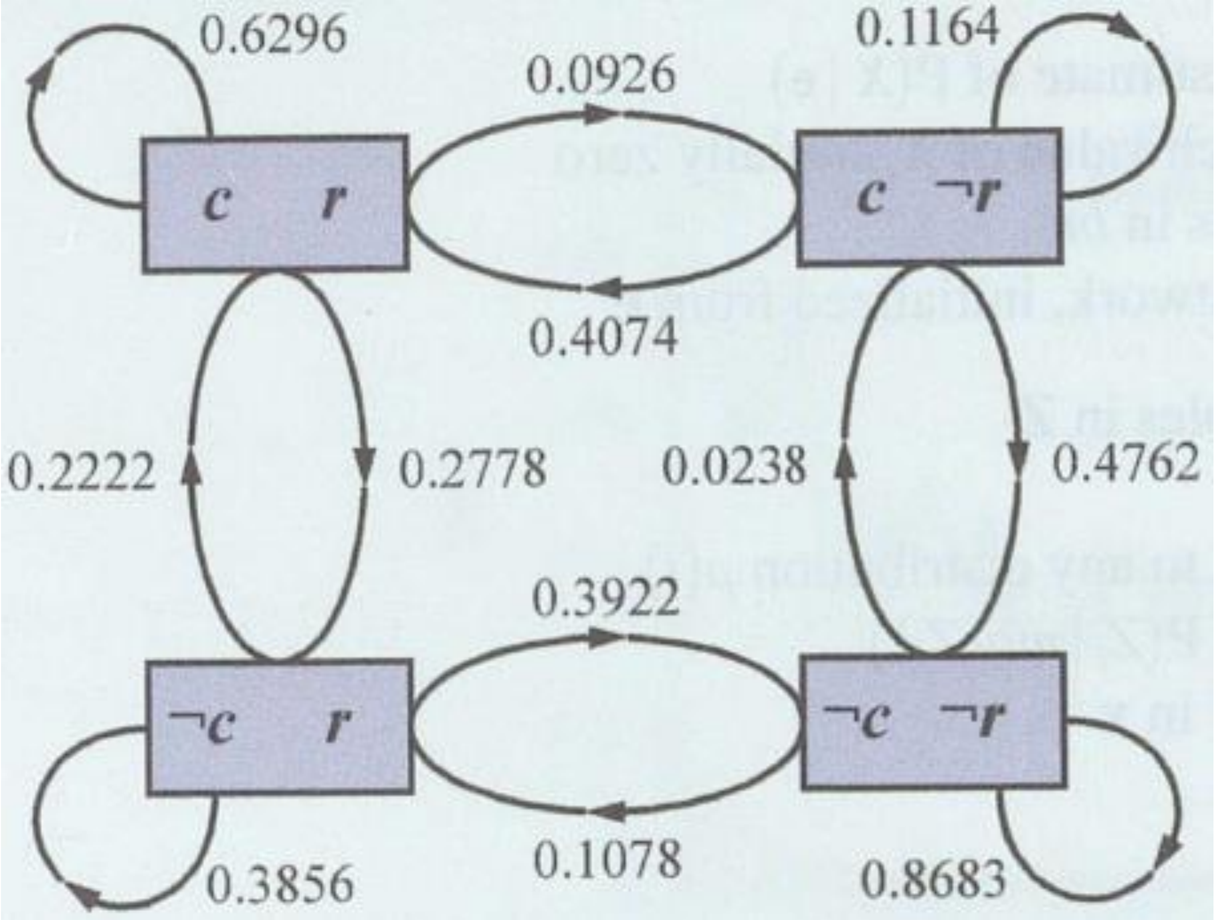
- In certain cases, even likelihood weighting produces runs with zero weight, e.g. in the Sprinkler example, if Sprinkler = false and Rain = false, then Wetgrass = true with probability 0

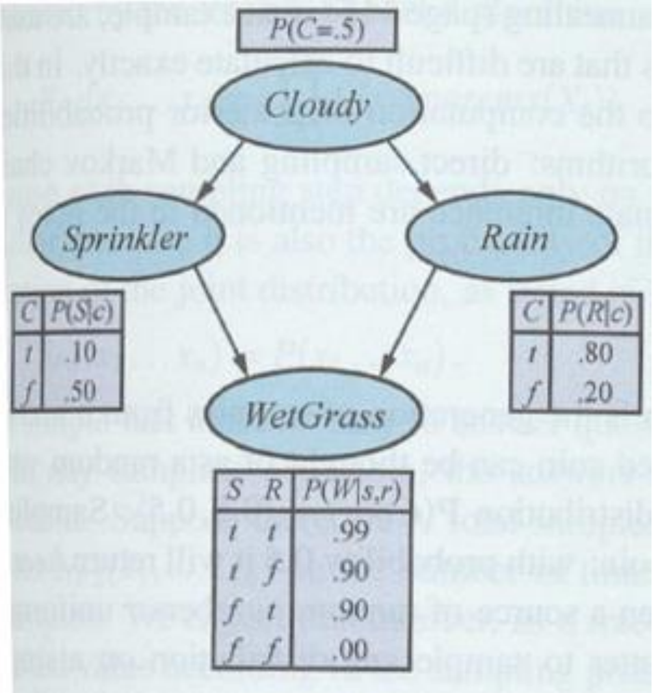
Markov Chain Simulation

- Sampling methods generate a sample in each run from scratch.
- Markov chain Monte Carlo (MCMC) algorithms generate a sample by making a random change to the preceding sample
  - In the long run, MCMC, stays in each state proportional to its a-posteriori probability
  - For large networks potentially much more efficient, because states can be „reused“

MCMC algorithm: Gibbs Sampling

- Initializiation: Compute a complete run (e.g. at random or with likelihood weighting)
- Continuation: Choose non-evidence variable and recompute its value with a random generator accoring to the probability of its markov blanket (i.e. its parents, children and children‘s parents)
  - The new state counts as a full run
  - Iteration, until enough samples are computed





Example: The states and transition probabilities for the query **P**(Rain|Sprinkler=true, WetGrass=true) computed from the markov blanket of the network including self loops

Gibbs Sampling

- X: query variable
- e: evidence variables
- bn: Bayesian net
- N: # Iterations

- mb(Z_i) = Markov blanket of variable Z_i

- Instead of choosing any variable from Z_i cycling through the variables works also.

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

  initialize **x** with random values for the variables in **Z**
  for $k = 1$ **to** N **do**
    choose any variable Z_i from **Z** according to any distribution $\rho(i)$
    set the value of Z_i in **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 **x**
  return NORMALIZE(**C**)

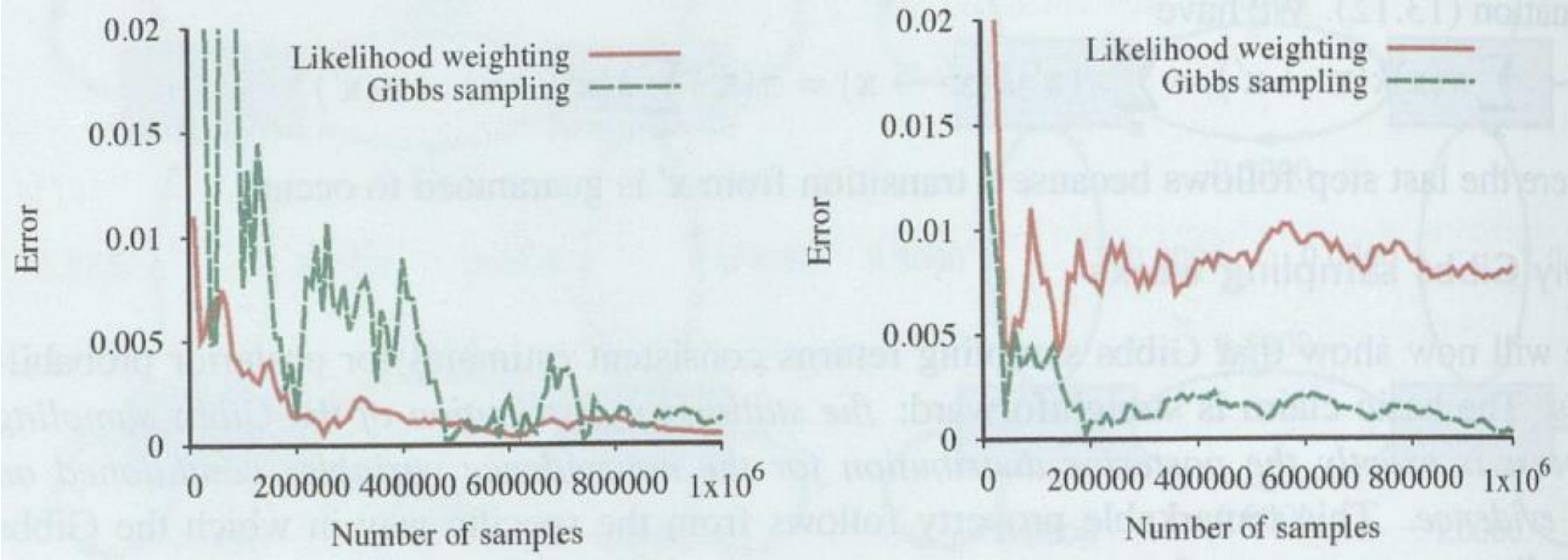
Example for MCMC

- Query: P (Rain|Sprinkler, WetGrass):
  - Evidence variables: Sprinkler = true, WetGrass = true
  - Free variables: Cloudy, Rain. Assume current values of true and false
    - Current variable assignment: [Cloudy, Sprinkler, Rain, WetGrass] (T,T,F,T)
  - N repetitions of following steps
    - New sample for Cloudy with P(Cloudy|Sprinkler, ¬Rain); assume Cloudy = F
      - New variable assignment: (F,T,F,T)
    - New sample for Rain with P(Rain|¬Cloudy, Sprinkler, ¬WetGrass); assume Rain = T
      - New variable assignment: (F,T,T,T)
  - Counting frequency of Rain: assume with 80 samples: (Rain=T) = 20 and (Rain=F) = 60
  - Normalized probability of rain 20/80 = 25%
- The new value of a variable X_i is sampled according to its markov blanket (mb):

$$P(x_i \mid mb(X_i)) = \alpha P(x_i \mid parents(X_i)) \prod_{Y_j \in Children(X_i)} P(y_j \mid parents(Y_j))$$

Comparison of Gibbs Sampling and Likelihood Weighting

- Left: Car insurance network for the standard query on Property Cost
- Right: Car insurance network with observed output variables and Age as query variable



Complexity of Gibbs Sampling

- Advantages:
  - Calculating the Markov blanket of a variable is independant of the size of the network
  - It does not share the problem of likelihood weighting with downstream evidence
- Problems:
  - Deterministic CPTs (conditional probability tables) might block transition between states
    - Example: Assume, it rains if and only if it is cloudy
      - Only two joint states for Cloudy and Rain have non-zero probability: [T, T] and [F,F]
      - Starting in [T, T], the state [F, F] can never be reached because the intermediate states [T, F] and [F, T] have probability zero
- Solutions:
  - Change deterministic CPTs to nearly deterministic (low convergence)
  - Block sampling: Sample multiple variables simultaneously
  - **Metropolis-Hastings sampling:** Combine Gibbs sampling and likihood weighting to produce completely new initializations of variables

Compiling Approximate Inference

- All sampling algorithms operate on a Bayesian network as data structure
  - Operations like finding the node‘s parents are repeated extreme often
  - Since the network structure remains fixed, it can be compiled
    - Tremendous speed up (2 – 3 orders of magnitude)
- Example for compilation of computing the markov blanket:
  - set the value of Earthquake in x by sampling from **P**(Earthquake|mb(Earthquake))

$$P(x_i \mid mb(X_i)) = \alpha P(x_i \mid parents(X_i)) \prod_{Y_j \in Children(X_i)} P(y_j \mid parents(Y_j))$$

- Compilation to model-specific sampling code for the Earthquake variable depending on Alarm and Burglary:
- Requires similar compilations for each variable of the network (not pretty, but fast)

```
r ← a uniform random sample from [0, 1]
if Alarm = true
   then if Burglary = true
      then return [r < 0.0020212]
      else return [r < 0.36755]
   else if Burglary = true
      then return [r < 0.0016672]
      else return [r < 0.0014222]
```

Causal Networks

- Causal networks are a subtype of Bayesian networks, where all relations are causal
  - Example: Sprinkler $\rightarrow$ WetGrass, but not Sprinkler $\rightarrow$ Cloudy
  - If turning on the sprinkler, the probability of WetGrass changes, but not of Cloudy
- **Causal networks allow to predict interventions** which should ignore „uncausal“ propagations
  - However, in the real world the reasons of an action (like turning on the sprinkler) might be unclear and are potentially influenced by other factors (like Cloudy), the „back door“
  - The back door can be blocked by d-separation, i.e. selecting variables Z (e.g. Rain) blocking all paths from the variable (e.g. sprinkler) to its parents (e.g. Cloudy), which must be set.
  - Useful for causal analysis in a wide range of non-experimental or quasi-experimental settings, like e.g. randomized controlled trials („if this happens instead, what would the probability have been?“)