AI基础

Lecture 10: Quantifying Uncertainty and Probabilistic Reasoning

Bin Yang
School of Data Science and Engineering
byang@dase.ecnu.edu.cn

[Some slides adapted from Philipp Koehn, JHU]

individual objects, categories of objects, and relations among objects.
Inheritance, default values,

Lecture 9 ILOs

$Bachelor = And(Unmarried, Adult, Male)$. DL: Categories

$Bachelor(x) \Leftrightarrow Unmarried(x) \land Adult(x) \land Male(x)$. FOL: Objects

- Knowledge representation
  - Events
  - Reasoning systems for categories
    - Semantic networks
    - Description logic
- Automated planning
  - Classic planning and Planning Domain Definition Lagrange (PDDL)
  - Algorithms for classic planning
    - Forward search (progression) and backward search (regression)
    - Heuristics: adding shortcut edges, reducing states
  - Hierarchical planning
  - Planning in nondeterministic domains
  - Scheduling
- Quantifying uncertainty
  - Acting under uncertainty
  - Probabilities

State: a conjunction of ground atomic fluents.
Action: precondition + effect

Applicable actions and relevant actions

plan first, schedule later
Critical path, slack
earliest possible start time, ES, latest possible start time, LE
Resource constraint (minimum slack heuristic)

Decision theory = probability theory + utility theory.
Maximum expected utility

Lecture 10 ILOs

- Quantifying uncertainty
  - Inference using full joint distributions
  - Bayes’ rule
  - The Wumpus world revisited
- Probabilistic programming
  - Bayesian networks
  - Exact Inference
  - Approximate inference
    - Direct sampling
    - Rejection sampling, Weight sampling
    - Markov Chain Monte Carlo

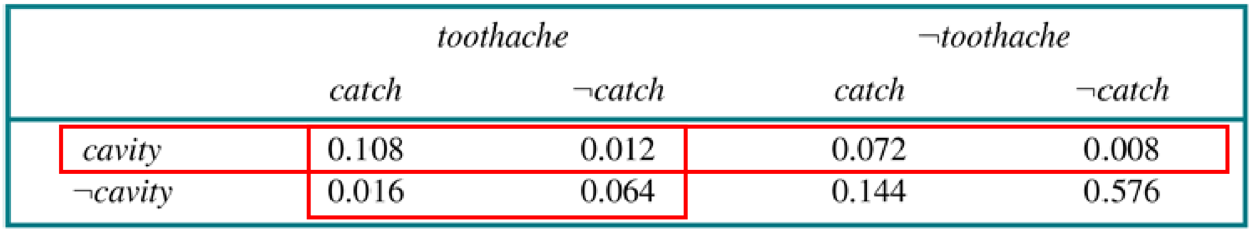
Outline

- Quantifying uncertainty
    - Inference using full joint distributions
    - Bayes’ rule
    - The Wumpus world revisited
- Probabilistic programming
    - Bayesian networks
    - Exact Inference
    - Approximate inference
        - Direct sampling
        - Rejection sampling, Weight sampling
        - Markov Chain Monte Carlo
- Short intro to our lab

Inference using full joint distributions

- A simple example
  - Three Boolean random variables
  - Toothache, cavity, catch
  - 2*2*2=8 possible worlds

Figure 12.3



	toothache		*¬toothache*	
	catch	*¬catch*	*catch*	*¬catch*
cavity	0.108	0.012	0.072	0.008
¬cavity	0.016	0.064	0.144	0.576

A full joint distribution for the *Toothache*, *Cavity*, *Catch* world.

- The probability associated with a proposition Φ is defined to be the sum of the probabilities of the worlds in which Φ holds.
  - identify those possible worlds in which the proposition is true and add up their probabilities

$$P(cavity \lor toothache) = 0.108 + 0.012 + 0.072 + 0.008 + 0.016 + 0.064 = 0.28.$$

$$P(cavity) = 0.108 + 0.012 + 0.072 + 0.008 = 0.2.$$

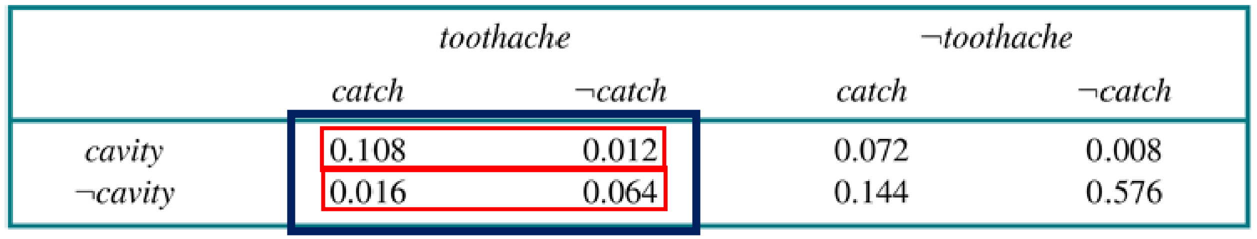
Marginal probability

For any proposition ϕ, $P(\phi) = \sum_{\omega \in \phi} P(\omega)$.

Conditional probabilities

$$P(a \mid b) = \frac{P(a \wedge b)}{P(b)} ,$$

Figure 12.3



	toothache		*¬toothache*	
	catch	*¬catch*	*catch*	*¬catch*
cavity	0.108	0.012	0.072	0.008
¬cavity	0.016	0.064	0.144	0.576

A full joint distribution for the *Toothache*, *Cavity*, *Catch* world.

- Computing conditional probabilities of some variables, given evidence about other variables.

$$P(cavity \mid toothache) = \frac{P(cavity \wedge toothache)}{P(toothache)} = \frac{0.108 + 0.012}{0.108 + 0.012 + 0.016 + 0.064} = 0.6.$$

$$P(\neg cavity \mid toothache) = \frac{P(\neg cavity \wedge toothache)}{P(toothache)} = \frac{0.016 + 0.064}{0.108 + 0.012 + 0.016 + 0.064} = 0.4.$$

$$\begin{aligned} \mathbf{P}(Cavity \mid toothache) &= \alpha \mathbf{P}(Cavity, toothache) \\ &= \alpha [\mathbf{P}(Cavity, toothache, catch) + \mathbf{P}(Cavity, toothache, \neg catch)] \\ &= \alpha [\langle 0.108, 0.016 \rangle + \langle 0.012, 0.064 \rangle] = \alpha \langle 0.12, 0.08 \rangle = \langle 0.6, 0.4 \rangle . \end{aligned}$$

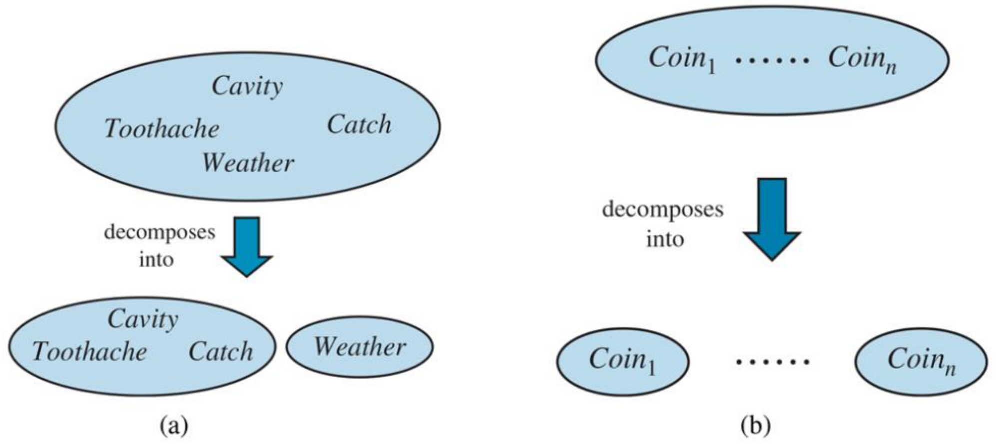
P(toothache) can be viewed as a normalization constant for the two conditional distributions. We use **α** to denote such constants.

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

X: query variable, e.g., cavity
e: observed variables, e.g., toothache
y: remaining unobserved variables, e.g., catch

Independence

- In addition to Boolean variables cavity, toothache, and catch, we have a fourth variable Weather
  - Weather={sun, rain, cloud, snow}
  - Four "editions" of the joint distribution table shown in previous slide, one for each weather.



Two examples of factoring a large joint distribution into smaller distributions, using absolute independence. (a) Weather and dental problems are independent. (b) Coin flips are independent.

$$P(toothache, catch, cavity, cloud)$$
$$= P(cloud \mid toothache, catch, cavity)P(toothache, catch, cavity).$$

$$P(cloud \mid toothache, catch, cavity) = P(cloud).$$

$$P(toothache, catch, cavity, cloud) = P(cloud)P(toothache, catch, cavity).$$

$$\mathbf{P}(Toothache, Catch, Cavity, Weather) = \mathbf{P}(Toothache, Catch, Cavity)\mathbf{P}(Weather).$$

$$\mathbf{P}(X \mid Y) = \mathbf{P}(X) \quad \text{or} \quad \mathbf{P}(Y \mid X) = \mathbf{P}(Y) \quad \text{or} \quad \mathbf{P}(X, Y) = \mathbf{P}(X)\mathbf{P}(Y).$$

	toothache		*¬toothache*	
	catch	*¬catch*	*catch*	*¬catch*
cavity	0.108	0.012	0.072	0.008
¬cavity	0.016	0.064	0.144	0.576

Outline

- Quantifying uncertainty
    - Inference using full joint distributions
    - Bayes’ rule
    - The Wumpus world revisited
- Probabilistic programming
    - Bayesian networks
    - Exact Inference
    - Approximate inference
        - Direct sampling
        - Rejection sampling, Weight sampling
        - Markov Chain Monte Carlo

Bayes' Rule

$$\mathbf{P}(Y \mid X) = \frac{\mathbf{P}(X \mid Y)\mathbf{P}(Y)}{\mathbf{P}(X)}.$$

- Computing P(Y|X) from P(X|Y), P(Y), and P(X)
    - Have good probability estimates for the three, and then compute the fourth.
- Causal direction
    - P(effect | cause)
    - Example: P(symptoms | disease)
- Diagnostic direction
    - P(cause | effect)
    - Example: P(disease | symptoms)
- Real world example
    - A doctor knows P(symptoms | disease), and want to derive a diagnosis P(disease | symptoms).

$$P(cause \mid effect) = \frac{P(effect \mid cause)P(cause)}{P(effect)}.$$

Example

$$P(cause \mid effect) = \frac{P(effect \mid cause) P(cause)}{P(effect)}.$$

$$P(s \mid m) = 0.7$$
$$P(m) = 1/50000$$
$$P(s) = 0.01$$
$$P(m \mid s) = \frac{P(s \mid m) P(m)}{P(s)} = \frac{0.7 \times 1/50000}{0.01} = 0.0014.$$

- A doctor knows that the disease **meningitis** causes a patient to have a **stiff neck**, say, 70% of the time.
- The doctor also knows some unconditional facts:
  - the prior probability that any patient has meningitis is 1/50000
  - the prior probability that any patient has a stiff neck is 1%
- Notice that even though a stiff neck is quite strongly indicated by meningitis (with probability 0.7), the probability of meningitis in patients with stiff necks remains small. This is because the prior probability of stiff necks (from any cause) is much higher than the prior for meningitis.

Diagnostic knowledge derived by Bayes rules vs. direct diagnostic knowledge

$$P(m \mid s) = \frac{P(s \mid m)P(m)}{P(s)}$$

- If the doctor has quantitative information in the diagnostic direction from symptoms to causes. Such a doctor has **no need** to use Bayes' rule.
  - E.g., the doctor knows that a stiff neck implies meningitis in 1 out of 5000 cases, i.e., P(m|s)=1/5000
- Diagnostic knowledge is often more fragile than causal knowledge.
  - If there is a sudden epidemic of meningitis, the unconditional probability of meningitis, P(m), will go up.
  - Diagnostic knowledge derived by Bayes rules: P(m|s) should go up proportionately with P(m). Causal knowledge P(s|m) is unaffected by the epidemic.
  - Direct diagnostic knowledge: has no idea of how to update
- The use of this kind of direct causal knowledge provides the crucial robustness needed to make probabilistic systems feasible in the real world.

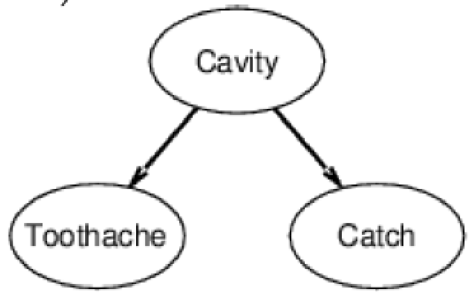
Naïve Bayes Models

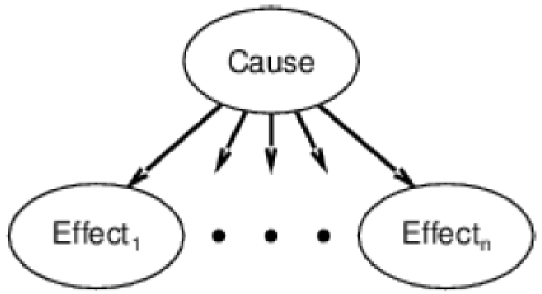
$$\mathbf{P}(X, Y \mid Z) = \mathbf{P}(X \mid Z)\mathbf{P}(Y \mid Z).$$

$$\mathbf{P}(toothache \wedge catch \mid Cavity) = \mathbf{P}(toothache \mid Cavity)\mathbf{P}(catch \mid Cavity).$$

- Conditional independence
- Naïve bayes:
  - We use "naive" because it is often used (as a simplifying assumption) in cases where the "effect" variables are not strictly independent given the cause variable.
  - In practice, naive Bayes systems often work very well, even when the conditional independence assumption is not strictly true.

$$\mathbf{P}(Cause, Effect_1, \ldots, Effect_n) = \mathbf{P}(Cause)\prod_i \mathbf{P}(Effect_i \mid Cause).$$





$$\mathbf{P}(Cause, Effect_1, \ldots, Effect_n) = \mathbf{P}(Cause)\prod_i \mathbf{P}(Effect_i \mid Cause).$$

Naïve Bayes Models

- e: observed effects
- y: unobserved effects

$$\mathbf{P}(Cause \mid \mathbf{e}) = \alpha \sum_{\mathbf{y}} \mathbf{P}(Cause, \mathbf{e}, \mathbf{y}).$$

$$\begin{aligned} \mathbf{P}(Cause \mid \mathbf{e}) &= \alpha \sum_{\mathbf{y}} \mathbf{P}(Cause)\,\mathbf{P}(\mathbf{y} | Cause) \left(\prod_j \mathbf{P}(e_j \mid Cause) \right) \\ &= \alpha\, \mathbf{P}(Cause) \left(\prod_j \mathbf{P}(e_j | Cause) \right) \boxed{\sum_{\mathbf{y}} \mathbf{P}(\mathbf{y} | Cause)} \\ &= \alpha\, \mathbf{P}(Cause) \prod_j \mathbf{P}(e_j \mid Cause) \end{aligned}$$

Conditional independence

Always being 1

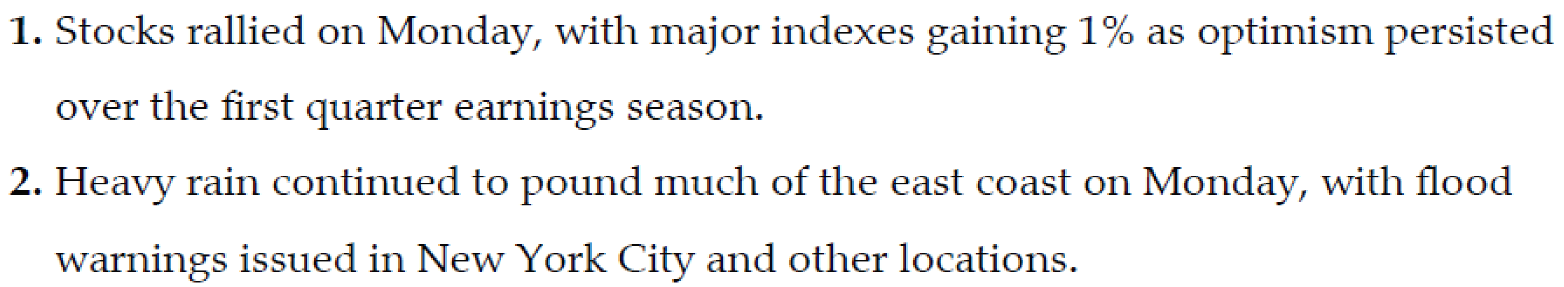
For each possible cause, multiply the prior probability of the cause by the product of the conditional probabilities of the observed effects given the cause; then normalize the result.

The run time of this calculation is linear in the number of observed effects and does not depend on the number of unobserved effects (which may be very large in domains such as medicine).

Example: text classification with naïve Bayes

- Given a text, decide which of a predefined set of classes or categories it belongs to.
- The "cause" is the category variable: news, sports, business, weather, etc.
- The "effect" variables are the presence or absence of certain key words HasWord_i

Two example sentences:



1. Stocks rallied on Monday, with major indexes gaining 1% as optimism persisted over the first quarter earnings season.

2. Heavy rain continued to pound much of the east coast on Monday, with flood warnings issued in New York City and other locations.

Example: text classification with naïve Bayes

$$\mathbf{P}(Cause \mid \mathbf{e}) = \alpha\, \mathbf{P}(Cause) \prod_j \mathbf{P}(e_j \mid Cause)$$

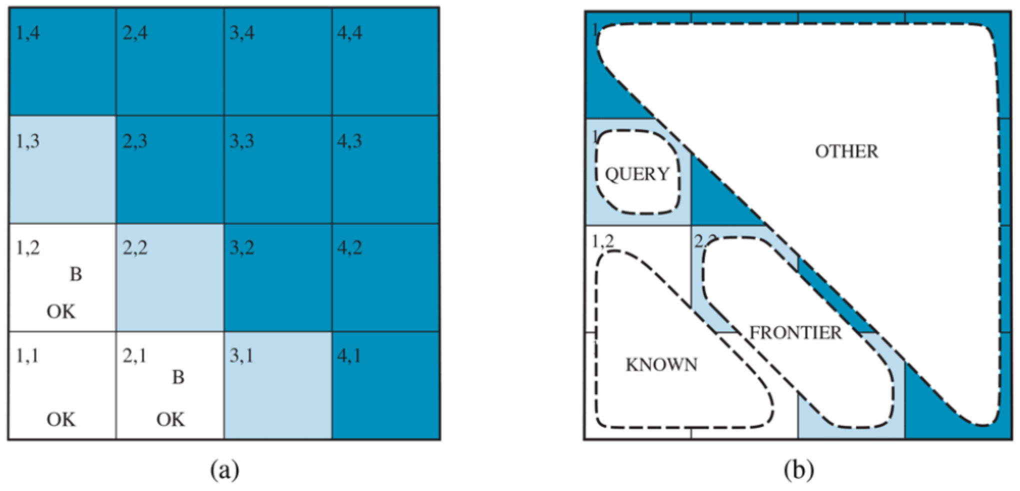
- Prior of causal, P(category)
  - For example, if 9% of articles are about weather, we set P(Category=weather)=0.09
- Causal knowledge, P(HasWord$_i$|category)
  - is estimated as the fraction of documents of each category that contain word i
  - say 37% of articles in the business category contain word 6, “stocks,” then we set P(HasWord$_6$ =true|category=business)=0.37
- To categorize a new document, we check which key words appear in the document and then apply the P(Cause | e) equation shown in the right upper corner to obtain the posterior probability distribution over categories.
  - If we have to predict just one category, we take the one with the highest posterior probability.

Outline

- Quantifying uncertainty
    - Inference using full joint distributions
    - Bayes' rule
    - The Wumpus world revisited
- Probabilistic programming
    - Bayesian networks
    - Exact Inference
    - Approximate inference
        - Direct sampling
        - Rejection sampling, Weight sampling
        - Markov Chain Monte Carlo

The Wumpus world revisited

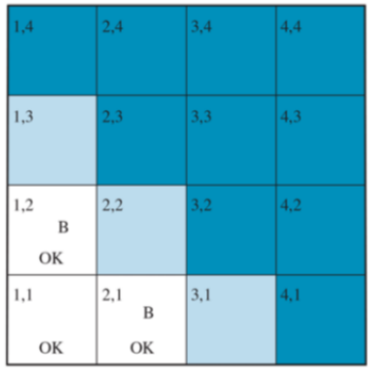
Figure 12.5



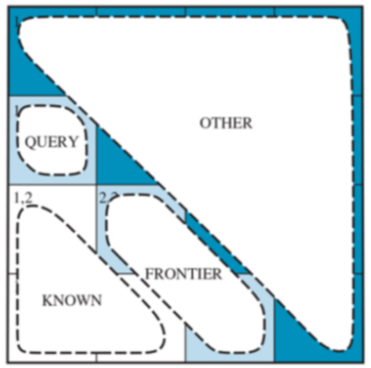
(a) After finding a breeze in both [1,2] and [2,1], the agent is stuck—there is no safe place to explore. (b) Division of the squares into *Known*, *Frontier*, and *Other*, for a query about [1,3].

- Breeze in both [1, 2] and [2, 1].
- the three unvisited but reachable squares [1,3], [2,2], and [3,1] might contain a pit.
- Pure logical inference can conclude nothing about which square is most likely to be safe, so a logical agent might have to choose randomly.
- We will see that a probabilistic agent can do much better than the logical agent.
  - How? By calculating the probability that each of the three squares contains a pit.

Specifying the probability model



(a)



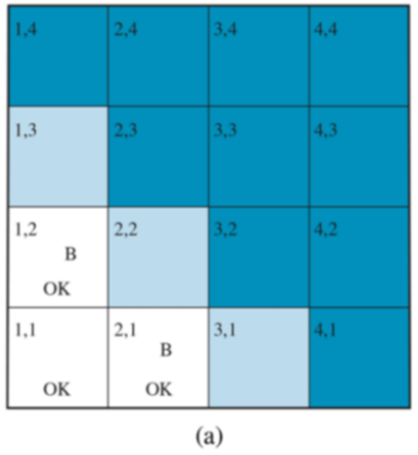
(b)

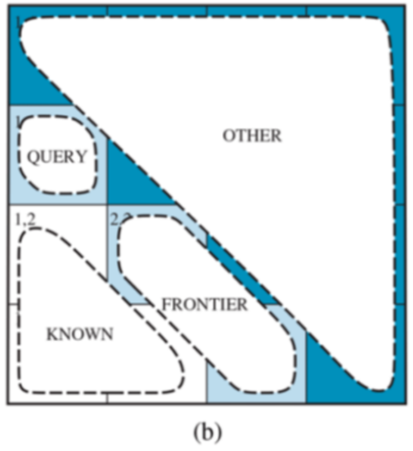
- A pit causes breezes in all neighboring squares
- Each square other than [1,1] contains a pit with probability 0.2.
- Random variables
  - P_{ij}: true iff [i, j] contains a pit
  - B_{ij}: true iff [i, j] is breezy, only for observed squares [1,1], [1,2], and [2,1]
- Full joint distribution

$$\mathbf{P}(P_{1,1},\ldots,P_{4,4},B_{1,1},B_{1,2},B_{2,1}) = \mathbf{P}(B_{1,1},B_{1,2},B_{2,1}|P_{1,1},\ldots,P_{4,4})\,\mathbf{P}(P_{1,1},\ldots,P_{4,4})\,.$$

- Prior: if there are n pits

$$\mathbf{P}(P_{1,1},\ldots,P_{4,4}) = \prod_{i,j=1,1}^{4,4}\mathbf{P}(P_{i,j}) = 0.2^n \times 0.8^{16-n}$$





Observations and query

- Know the following facts

$$b = \neg b_{1,1} \wedge b_{1,2} \wedge b_{2,1} \qquad known = \neg p_{1,1} \wedge \neg p_{1,2} \wedge \neg p_{2,1}$$

- Query: P(P$_{1,3}$ | known, b)
- Define Unknown = P$_{i,j}$ other than the query P$_{1,3}$ and known

$$\mathbf{P}(P_{1,3}|known, b) = \alpha \sum_{unknown} \mathbf{P}(P_{1,3}, unknown, known, b)$$

- Using conditional independency
  - observations are conditionally independent of other hidden squares given neighboring hidden squares
- Unknown=Frontier and Other

$$\mathbf{P}(P_{1,3}|\, known, b)$$

$$= \alpha \sum_{unknown} \mathbf{P}(P_{1,3}, known, b, unknown) \qquad \text{(from Equation (12.23))}$$

$$= \alpha \sum_{unknown} \mathbf{P}(b|P_{1,3}, known, unknown)\mathbf{P}(P_{1,3}, known, unknown) \quad \text{(product rule)}$$

$$= \alpha \sum_{frontier} \sum_{other} \mathbf{P}(b|\, known, P_{1,3}, frontier, other)\mathbf{P}(P_{1,3}, known, frontier, other)$$

Unknown=Frontier and Other

$$= \alpha \sum_{frontier} \sum_{other} \mathbf{P}(b|\, known, P_{1,3}, frontier)\mathbf{P}(P_{1,3}, known, frontier, other),$$

b is conditionally independent from other unknown variables given frontier

$$= \alpha \sum_{frontier} \mathbf{P}(b|\, known, P_{1,3}, frontier) \sum_{other} \mathbf{P}(P_{1,3}, known, frontier, other).$$

As the first term has no **other** anymore, sum over other only affects the second term

$$= \alpha \sum_{frontier} \mathbf{P}(b|known, P_{1,3}, frontier) \sum_{other} \mathbf{P}(P_{1,3})P(known)P(frontier)P(other)$$

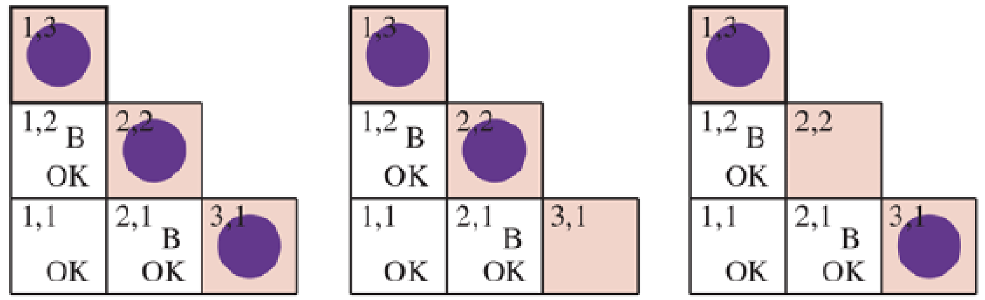
Independence among square having pits

$$= \alpha P(known)\mathbf{P}(P_{1,3}) \sum_{frontier} \mathbf{P}(b|\, known, P_{1,3}, frontier)P(frontier) \sum_{other} P(other)$$

Sum over other on other is 1

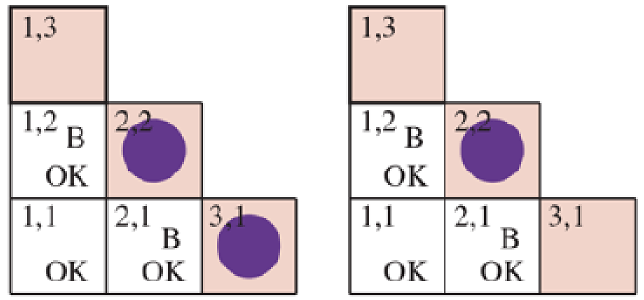
$$= \alpha' \mathbf{P}(P_{1,3}) \sum_{frontier} \mathbf{P}(b|\, known, P_{1,3}, frontier)P(frontier),$$

α’ is a new normalizing constant with P(known)



$0.2 \times 0.2 = 0.04$ $\quad 0.2 \times 0.8 = 0.16$ $\quad 0.8 \times 0.2 = 0.16$

(a)



$0.2 \times 0.2 = 0.04$ $\quad 0.2 \times 0.8 = 0.16$

(b)

(a) $P(P_{1,3})=0.2$
Each square other than [1,1]
contains a pit with probability 0.2.

(b) $P(P_{1,3})=1-0.2=0.8$

$$\mathbf{P}(P_{1,3}|\,known, b) = \alpha' \mathbf{P}(P_{1,3}) \sum_{frontier} \mathbf{P}(b|\,known, P_{1,3}, frontier) P(frontier),$$

$$\mathbf{P}(P_{1,3}|known, b) = \alpha' \langle 0.2(0.04 + 0.16 + 0.16),\ 0.8(0.04 + 0.16) \rangle \approx \langle 0.31, 0.69 \rangle\ .$$

[1, 3] has a pit with probability of 0.31

Mini quiz: what is the probability of [2, 2] and [3, 1] having a pit, respectively?

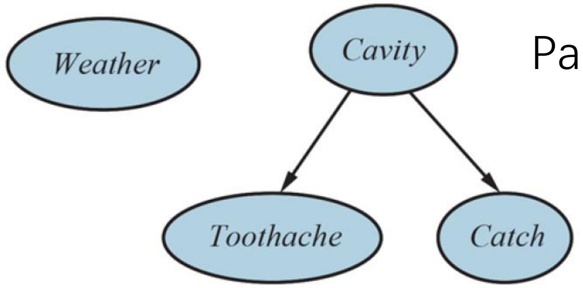
Outline

- Quantifying uncertainty
    - Inference using full joint distributions
    - Bayes' rule
    - The Wumpus world revisited
- Probabilistic programming
    - Bayesian networks
    - Exact Inference
    - Approximate inference
        - Direct sampling
        - Rejection sampling, Weight sampling
        - Markov Chain Monte Carlo

Representing knowledge in an uncertain domain

- ***Full joint probability distribution*** can answer any question about an uncertain domain, but can become intractably large as the number of variables grows.
- Independence and conditional independence relationships among variables can greatly reduce the number of probabilities that need to be specified in order to define the full joint distribution.
- We will now introduce a data structure called a **Bayesian network** to represent the dependencies among variables.
- Bayesian networks can represent essentially any full joint probability distribution and in many cases can do so very concisely.

Bayesian network



A simple Bayesian network in which *Weather* is independent of the other three variables and *Toothache* and *Catch* are conditionally independent, given *Cavity*.

- A Bayesian network is a directed graph in which each node is annotated with quantitative probability information.

1. Each node corresponds to a random variable, which may be discrete or continuous.

2. Directed links or arrows connect pairs of nodes. If there is an arrow from node X to node Y, X is said to be a *parent* of Y. The graph has no directed cycles and hence is a directed acyclic graph, or DAG.

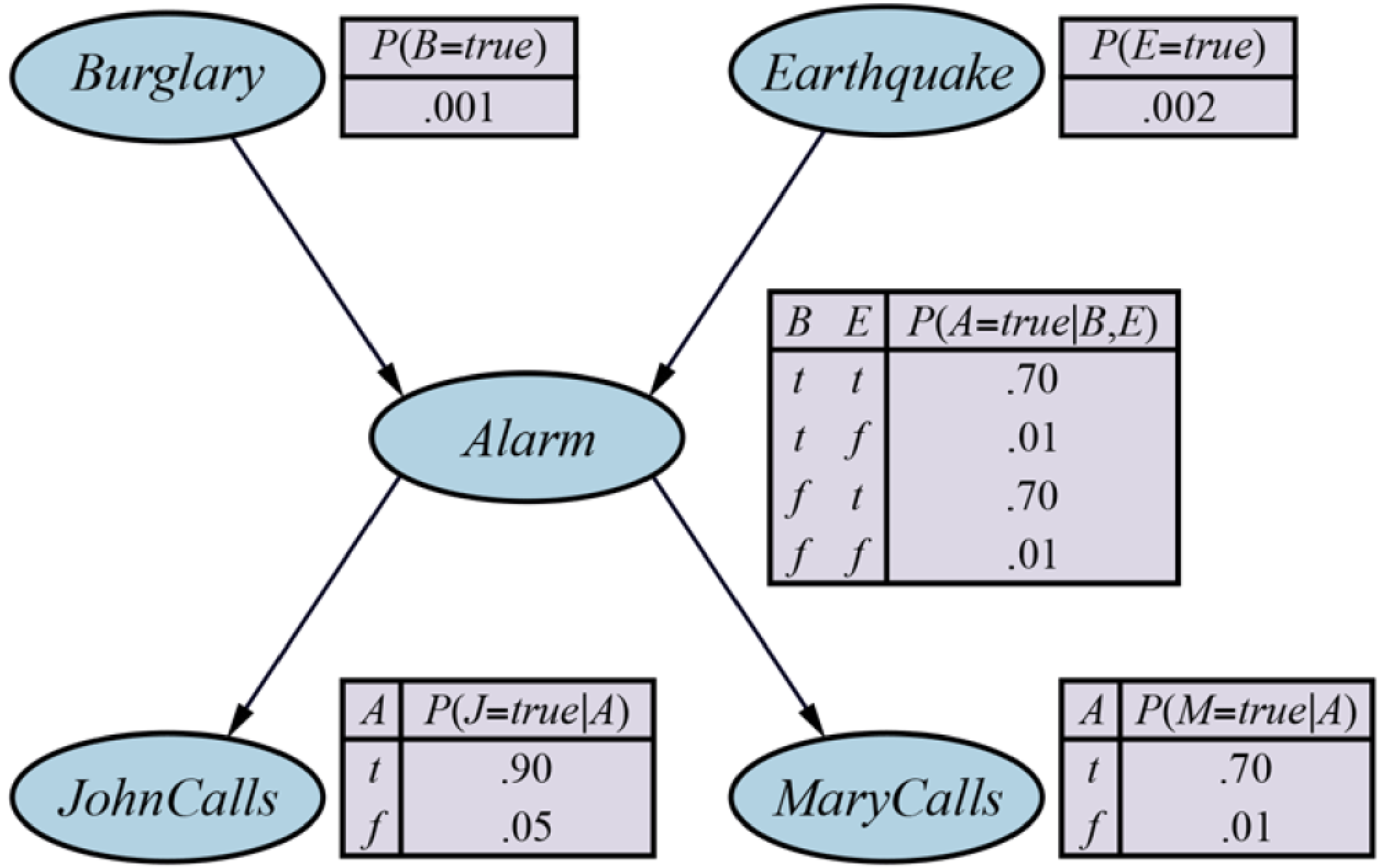
3. Each node X_i has associated probability information $\theta(X_i|Parents(X_i))$ that quantifies the effect of the parents on the node using a finite number of **parameters.**

Weather is independent of the other variables.
Toothache and **Catch** are conditionally independent, given **Cavity**. (The absence of a link between **Toothache** and **Catch**.)
Intuitively, the network represents the fact that **Cavity** is a ***direct cause*** of **Toothache** and **Catch**, whereas no direct causal relationship exists between **Toothache** and **Catch**.

Burglary example

- You have a new burglar alarm installed at home. It is fairly reliable at detecting a burglary, but is occasionally set off by minor earthquakes. (This example is due to Judea Pearl, a resident of earthquake-prone Los Angeles.)
- You have two neighbors, John and Mary, who have promised to call you at work when they hear the alarm.
- John nearly always calls when he hears the alarm, but sometimes confuses the telephone ringing with the alarm and calls then, too.
- Mary, on the other hand, likes rather loud music and often misses the alarm altogether.
- Given the evidence of who has or has not called, we would like to estimate the probability of a burglary.

Figure 13.2



P(B=true)
.001

P(E=true)
.002

B	E	P(A=true|B,E)
t	t	.70
t	f	.01
f	t	.70
f	f	.01

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

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

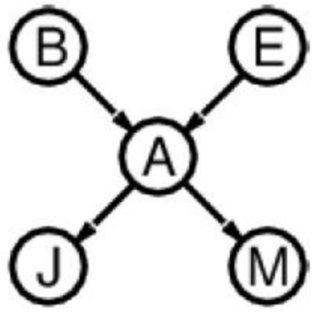
A typical Bayesian network, showing both the topology and the conditional probability tables (CPTs). In the CPTs, the letters *B*, *E*, *A*, *J*, and *M* stand for *Burglar*, *Earthquake*, *Alarm*, *JohnCalls*, and *MaryCalls*, respectively.

3. Each node X_i has associated probability information $\theta(X_i|Parents(X_i))$ that quantifies the effect of the parents on the node using a finite number of **parameters**.

- A conditional probability table (CPT) giving the distribution over X_i for each combination of its parent values
- Each row sums to 1.
  - For Boolean, omit one column: p and 1-p.
- A node without parents has only one row, prior.

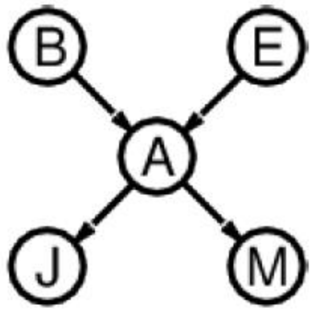
The probabilities actually summarize a potentially infinite set of circumstances in which the alarm might fail to go off (high humidity, power failure, dead battery, cut wires, a dead mouse stuck inside the bell, etc.)
John or Mary might fail to call and report it (out to lunch, on vacation, temporarily deaf, passing helicopter, etc.).
In this way, a small agent can cope with a very large world, at least approximately.

Analysis



- A conditional probability table for Boolean X_i with k Boolean parents has 2^k rows for the combinations of parent values
- Each row requires one number p for $X_i = true$ (the number for $X_i = false$ is just $1 - p$)
- If each variable has no more than k parents, the complete network requires $O(n \cdot 2^k)$ numbers
- I.e., grows linearly with n, vs. $O(2^n)$ for the full joint distribution
- For burglary net, $1 + 1 + 4 + 2 + 2 = 10$ numbers (vs. $2^5 - 1 = 31$)

The Semantics of Bayesian Networks



- The syntax of a Bayes net consists of a directed acyclic graph with some local probability information attached to each node.
- The semantics defines how the syntax corresponds to a joint distribution over the variables of the network.
- Global semantics defines the full joint distribution as the product of the local conditional distributions:

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i | parents(X_i))$$

- Example: the alarm has sounded, but neither a burglary nor an earthquake has occurred, and both John and Mary call.

$$\begin{aligned} P(j,m,a,\neg b,\neg e) &= P(j|a)P(m|a)P(a|\neg b \wedge \neg e)P(\neg b)P(\neg e) \\ &= 0.90 \times 0.70 \times 0.01 \times 0.999 \times 0.998 = 0.00628. \end{aligned}$$

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i | parents(X_i)).$$

Constructing Bayesian Networks

1. **NODES:** First determine the set of variables that are required to model the domain. Now order them, $\{X_1, \ldots, X_n\}$. Any order will work, but the resulting network will be more compact if the variables are ordered such that causes precede effects.
2. **LINKS:** For $i = 1$ to n do:
   - Choose a minimal set of parents for X_i from $X_1, \ldots, X_{i-1}$, such that Equation (13.3) is satisfied.
   - For each parent insert a link from the parent to X_i.
   - CPTs: Write down the conditional probability table, $\mathbf{P}(X_i | Parents(X_i))$.

(13.3)

$$\mathbf{P}(X_i | X_{i-1}, \ldots, X_1) = \mathbf{P}(X_i | Parents(X_i)),$$

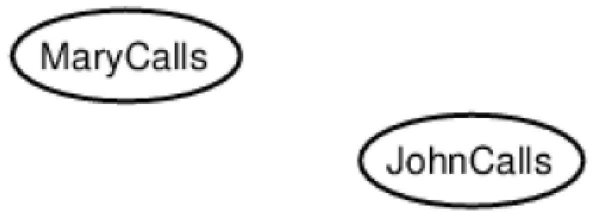
$$P(x_1, \ldots, x_n) = P(x_n | x_{n-1}, \ldots, x_1) P(x_{n-1}, \ldots, x_1).$$

$$\begin{aligned} P(x_1, \ldots, x_n) &= P(x_n | x_{n-1}, \ldots, x_1) P(x_{n-1} | x_{n-2}, \ldots, x_1) \cdots P(x_2 | x_1) P(x_1) \\ &= \prod_{i=1}^{n} P(x_i | x_{i-1}, \ldots, x_1). \end{aligned}$$

$$\mathbf{P}(X_i | X_{i-1}, \ldots, X_1) = \mathbf{P}(X_i | Parents(X_i)),$$

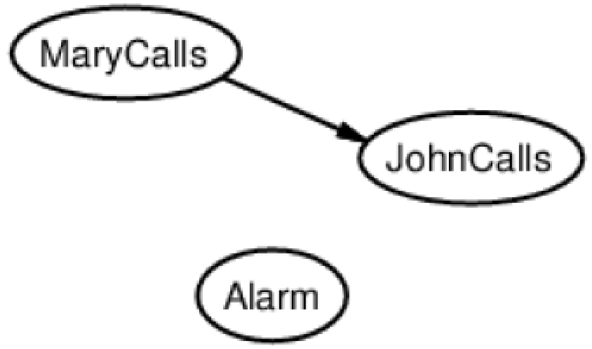
Compactness and node ordering

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



If Mary calls, that probably means the alarm has gone off,
which makes it more likely that John calls. Thus, No.
We need M as the parent of J.

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

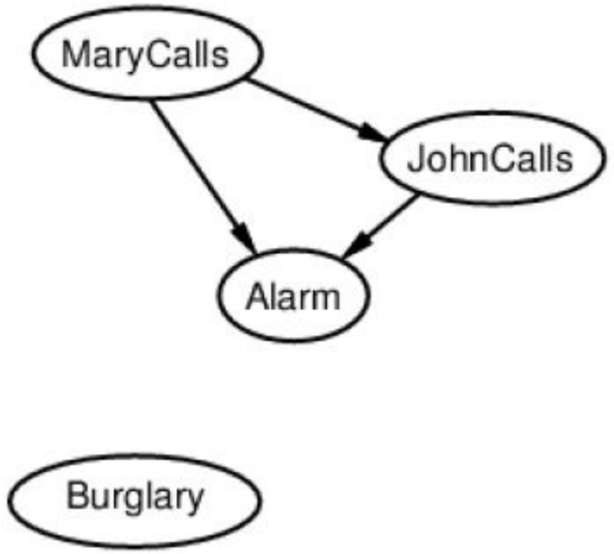


Clearly, if both call, it is more likely that the alarm has gone off than if just one or neither calls, so we need both M and J as parents of A. So, No.

$P(J|M) = P(J)$? No

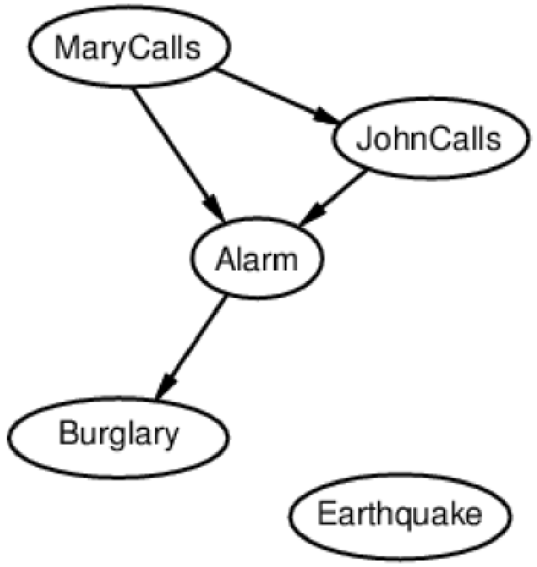
$P(A|J,M) = P(A|J)$? $P(A|J,M) = P(A)$?

If we know the alarm state, then the call from John or Mary might give us information about our phone ringing or Mary's music, but not about burglary.



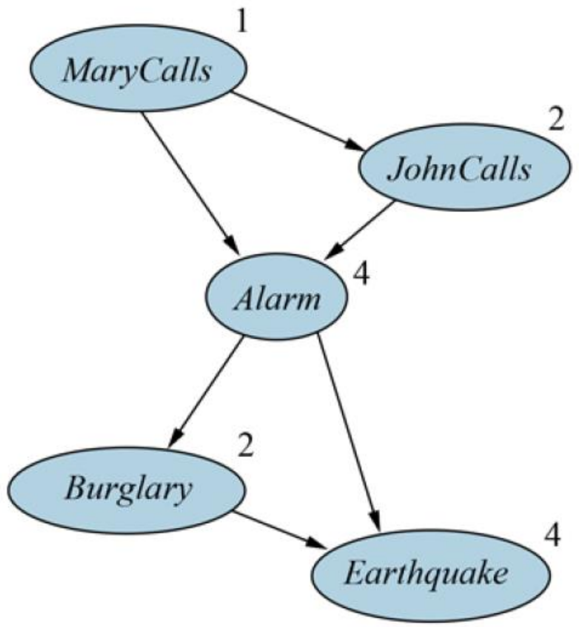
- $P(J|M) = P(J)$? No
- $P(A|J,M) = P(A|J)$? $P(A|J,M) = P(A)$? No
- $P(B|A,J,M) = P(B|A)$? Yes
- $P(B|A,J,M) = P(B)$? No

If the alarm is on, it is more likely that there has been an earthquake. (The alarm is an earthquake detector of sorts.) But if we know that there has been a burglary, then that explains the alarm, and the probability of an earthquake would be only slightly above normal. Hence, we need both *Alarm* and *Burglar* as parents.

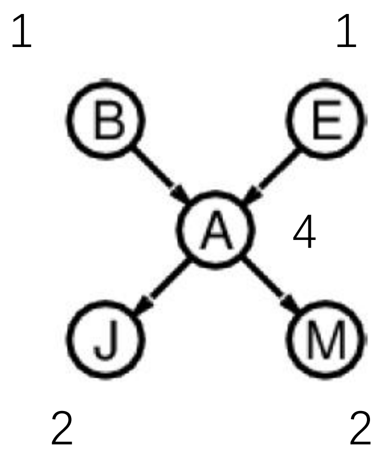


- $P(J|M) = P(J)$? No
- $P(A|J,M) = P(A|J)$? $P(A|J,M) = P(A)$? No
- $P(B|A,J,M) = P(B|A)$? Yes
- $P(B|A,J,M) = P(B)$? No
- $P(E|B,A,J,M) = P(E|A)$? No
- $P(E|B,A,J,M) = P(E|A,B)$? Yes

Compactness and node order

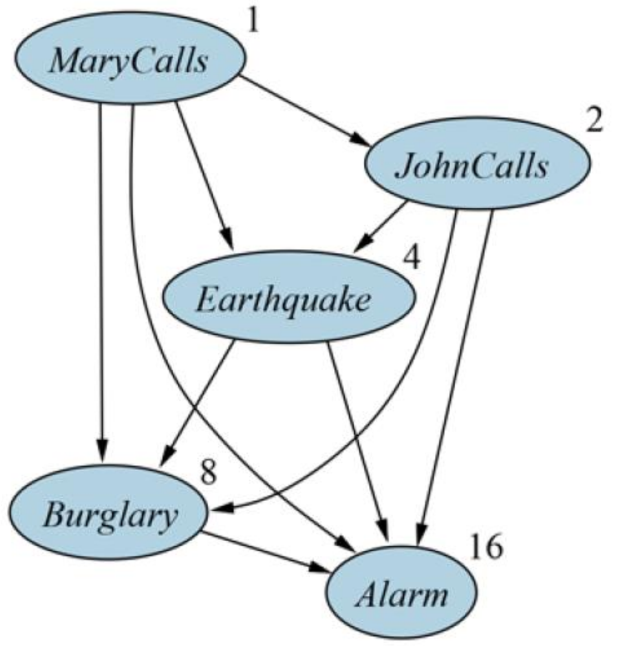


CPT has 1+2+4+2+4=13 rows.



CPT has 1+1+4+2+2=10 rows.

More compact.
Node order matters.



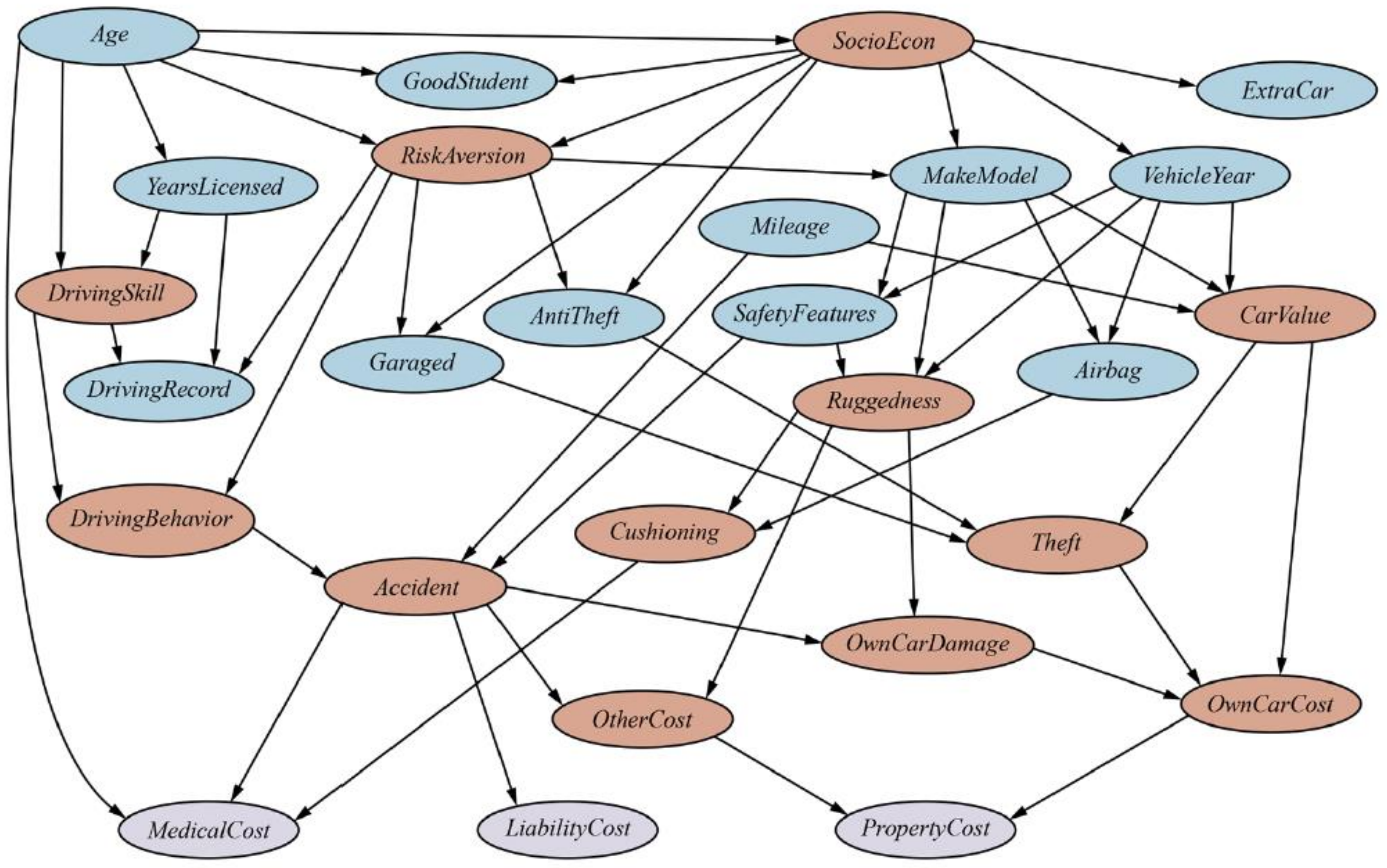
CPT has 1+2+4+8+16=15 rows.
No difference from full joint distribution.

Node order: the resulting network will be more compact if the variables are ordered such that **causes precede effects.**
If we stick to a causal model, we end up having to specify fewer numbers, and the numbers will often be easier to come up with.

Application form asks for the input information (Light Blue)

Case study: car insurance

- A car insurance company receives an application from an individual to insure a specific vehicle and must decide on the appropriate annual premium to charge, based on the anticipated claims it will pay out for this applicant.
- The task is to build a Bayes net that captures the causal structure of the domain and gives an accurate, well-calibrated distribution over the output variables given the evidence available from the application form.



A Bayesian network for evaluating car insurance applications.

The MedicalCost for any injuries sustained by the applicant
The LiabilityCost for lawsuits filed by other parties against the applicant and the company
The PropertyCost for vehicle damage to either party and vehicle loss by theft.

Outline

- Quantifying uncertainty
    - Inference using full joint distributions
    - Bayes' rule
    - The Wumpus world revisited
- Probabilistic programming
    - Bayesian networks
    - Exact Inference
    - Approximate inference
        - Direct sampling
        - Rejection sampling, Weight sampling
        - Markov Chain Monte Carlo

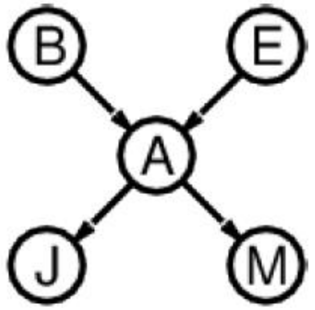
Exact Inference in Bayesian Networks

- The basic task for any probabilistic inference system is to compute the posterior probability distribution for a set of **query variables**, given some observed **event.**

$$\mathbf{P}(Burglary | JohnCalls = true, MaryCalls = true)$$

- {X} ∪ E ∪ Y
  - {X} query variable
  - E evidence variables
  - Y hidden variables
- Inference: P(X|e)
- Only one query variable at a time
  - **P(U, V|e)=P(V|e) P(U|V,e)**

Inference by enumeration



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

$$\mathbf{P}(Burglary|JohnCalls = true, MaryCalls = true)$$

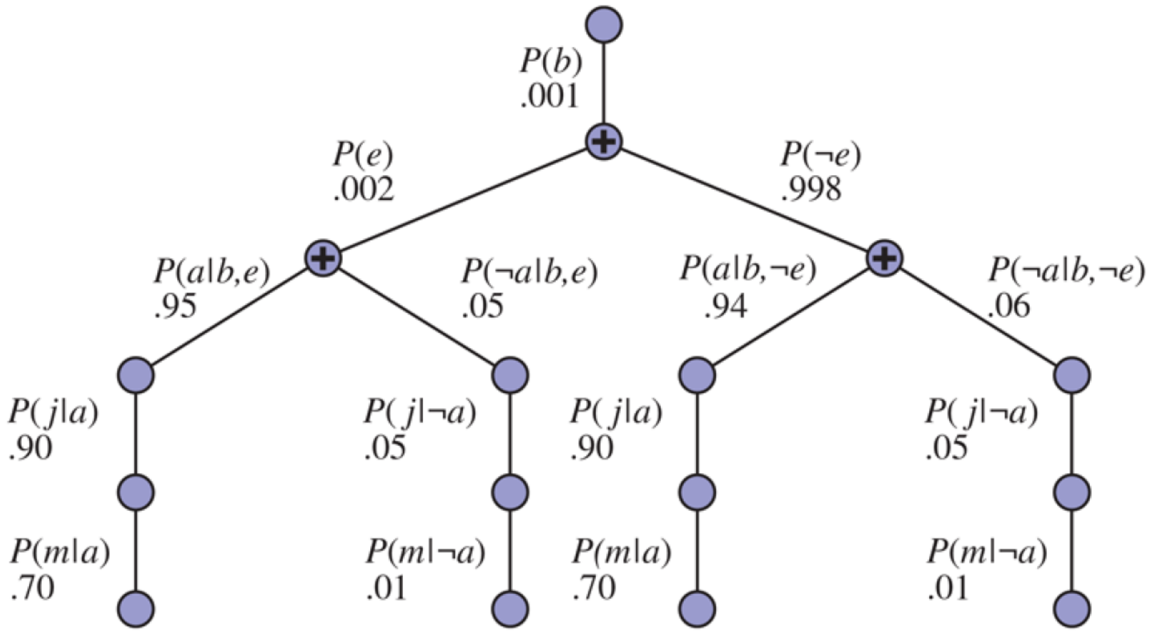
Earthquake and Alarm are hidden variables

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

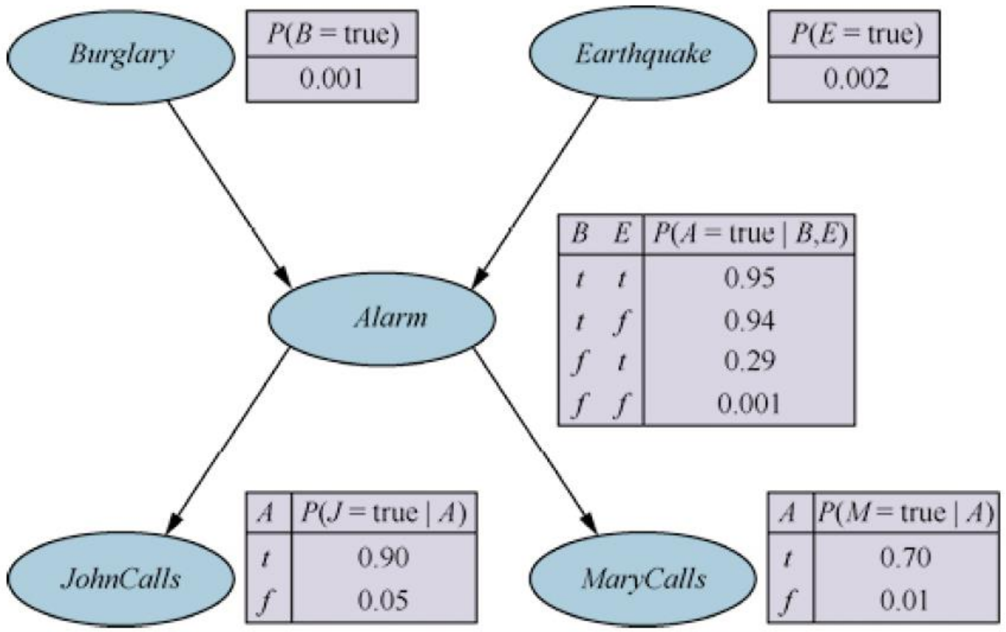
$$P(b|j,m) = \alpha \sum_{e}\sum_{a} P(b)P(e)P(a|b,e)P(j|a)P(m|a).$$

$$P(b|j,m) = \alpha\,P(b)\sum_{e} P(e)\sum_{a} P(a|b,e)P(j|a)P(m|a).$$

Figure 13.10



The structure of the expression shown in Equation (13.5). The evaluation proceeds top down, multiplying values along each path and summing at the "+" nodes. Notice the repetition of the paths for j and m.



P(B = true)
0.001

P(E = true)
0.002

B	E	P(A = true | B,E)
t	t	0.95
t	f	0.94
f	t	0.29
f	f	0.001

A	P(J = true | A)
t	0.90
f	0.05

A	P(M = true | A)
t	0.70
f	0.01

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

- DFS
    - Linear space complexity
    - $O(2^n)$ time complexity

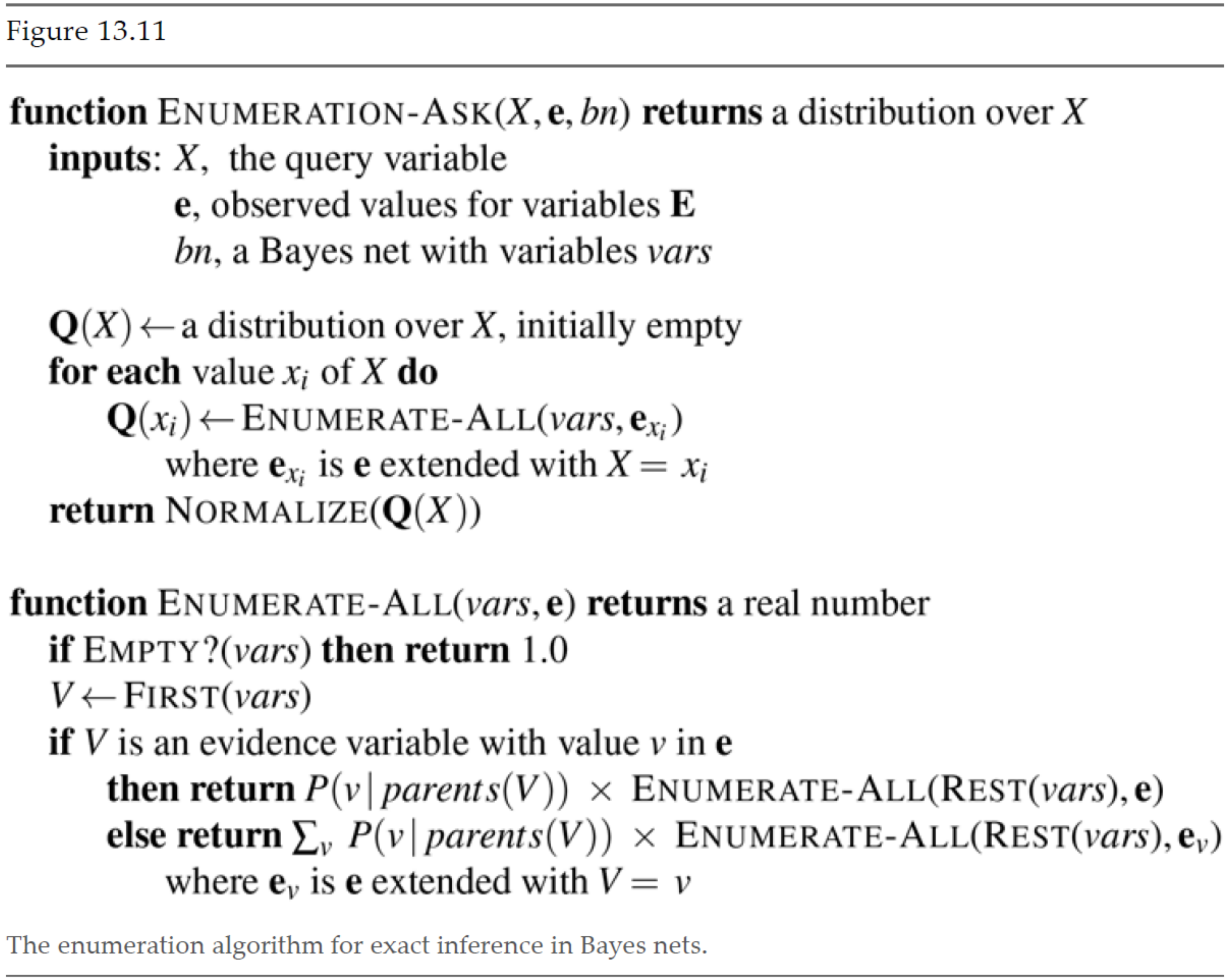


Figure 13.11

```
function ENUMERATION-ASK($X$, $\mathbf{e}$, $bn$) returns a distribution over $X$
    inputs: $X$, the query variable
            $\mathbf{e}$, observed values for variables $\mathbf{E}$
            $bn$, 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 $\mathbf{e}$ extended with $X = x_i$
    return NORMALIZE($\mathbf{Q}(X)$)

function ENUMERATE-ALL($vars$, $\mathbf{e}$) 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 $\mathbf{e}$
        then return $P(v \mid parents(V)) \times$ ENUMERATE-ALL(REST($vars$), $\mathbf{e}$)
        else return $\sum_v P(v \mid parents(V)) \times$ ENUMERATE-ALL(REST($vars$), $\mathbf{e}_v$)
            where $\mathbf{e}_v$ is $\mathbf{e}$ extended with $V = v$
```

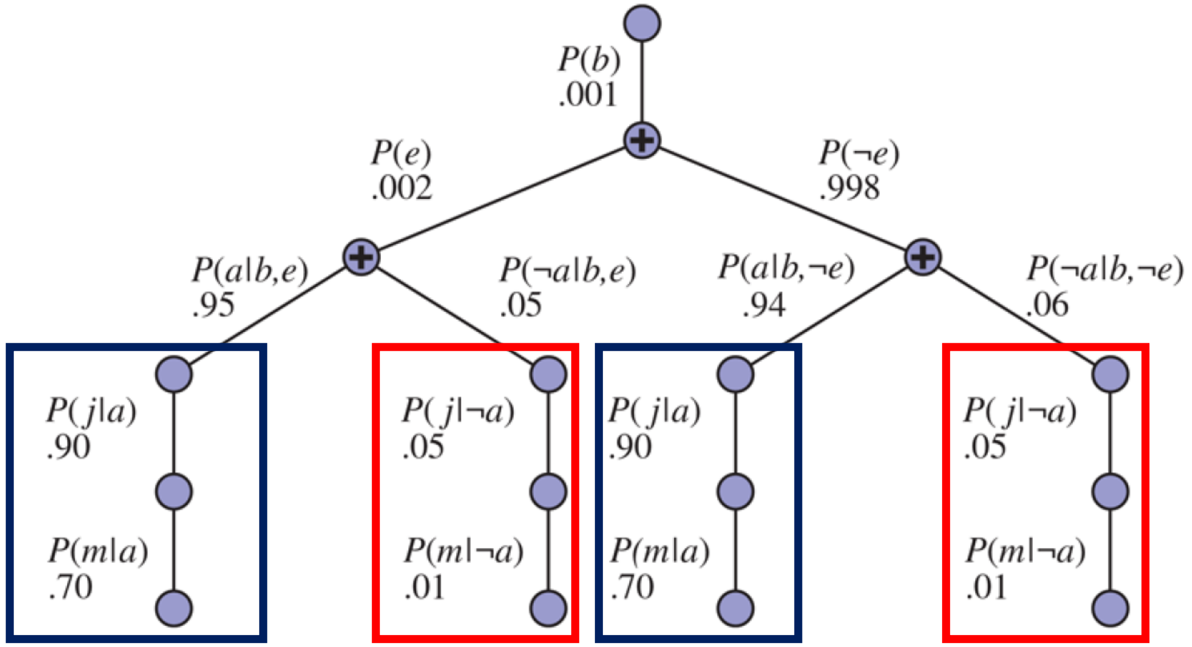
The enumeration algorithm for exact inference in Bayes nets.

Inference by Variable Elimination

- The enumeration algorithm can be improved substantially by eliminating repeated calculations.
- The idea is simple: do the calculation once and save the results for later use. This is a form of **dynamic programming**.
- **Variable elimination** works by evaluating expressions in right-to-left order (that is, bottom up in Figure 13.10).
- Intermediate results are stored, and summations over each variable are done only for those portions of the expression that depend on the variable.

$$\mathbf{P}(B|j,m) = \alpha \underbrace{\mathbf{P}(B)}_{\mathbf{f}_1(B)} \sum_e \underbrace{P(e)}_{\mathbf{f}_2(E)} \sum_a \underbrace{\mathbf{P}(a|B,e)}_{\mathbf{f}_3(A,B,E)} \underbrace{P(j|a)}_{\mathbf{f}_4(A)} \underbrace{P(m|a)}_{\mathbf{f}_5(A)}.$$

Figure 13.10



The structure of the expression shown in Equation (13.5). The evaluation proceeds top down, multiplying values along each path and summing at the "+" nodes. Notice the repetition of the paths for j and m.

$$\mathbf{P}(B|j,m) = \alpha\, \mathbf{f}_1(B) \times \sum_e \mathbf{f}_2(E) \times \sum_a \mathbf{f}_3(A,B,E) \times \mathbf{f}_4(A) \times \mathbf{f}_5(A).$$

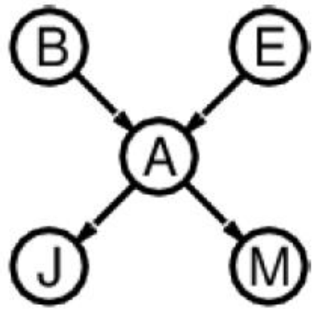
$$\begin{aligned}\mathbf{f}_6(B,E) &= \sum_a \mathbf{f}_3(A,B,E) \times \mathbf{f}_4(A) \times \mathbf{f}_5(A)\\ &= (\mathbf{f}_3(a,B,E) \times \mathbf{f}_4(a) \times \mathbf{f}_5(a)) + (\mathbf{f}_3(\neg a,B,E) \times \mathbf{f}_4(\neg a) \times \mathbf{f}_5(\neg a)).\end{aligned}$$

$$\mathbf{P}(B|j,m) = \alpha\, \mathbf{f}_1(B) \times \sum_e \mathbf{f}_2(E) \times \mathbf{f}_6(B,E).$$

$$\begin{aligned}\mathbf{f}_7(B) &= \sum_e \mathbf{f}_2(E) \times \mathbf{f}_6(B,E)\\ &= \mathbf{f}_2(e) \times \mathbf{f}_6(B,e) + \mathbf{f}_2(\neg e) \times \mathbf{f}_6(B,\neg e).\end{aligned}$$

$$\mathbf{P}(B|j,m) = \alpha\, \mathbf{f}_1(B) \times \mathbf{f}_7(B)$$

Irrelevant Variables



- Consider a query P(JohnCalls|Burgalry=true)

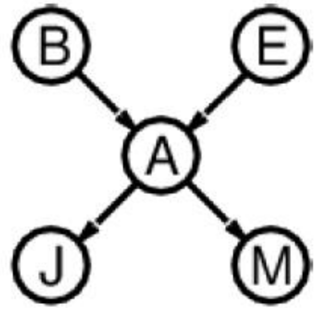
$$\mathbf{P}(J|b) = \alpha\, P(b) \sum_e P(e) \sum_a P(a|b,e)\mathbf{P}(J|a) \boxed{\sum_m P(m|a).}$$

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

- Y is irrelevant unless Y ∈ Ancestors({X} ∪ E)

$X = JohnCalls$, $\mathbf{E} = \{Burglary\}$
$Ancestors(\{X\} \cup \mathbf{E}) = \{Alarm, Earthquake\}$
$MaryCalls$ is irrelevant

Complexity of exact inference



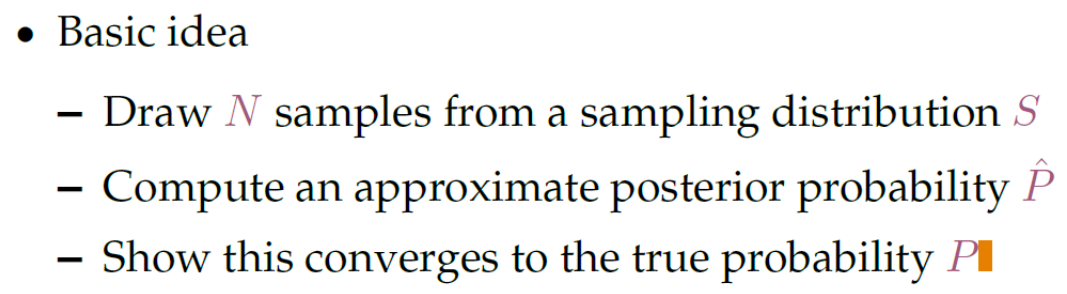
- Singly connected networks (polytrees)
    - There is **at most one undirected path** (i.e., ignoring the direction of the arrows) between any two nodes in the network.
    - The time and space complexity of exact inference in polytrees is linear in the size of the network. Here, the size is defined as the number of CPT entries; if the number of parents of each node is bounded by a constant, then the complexity will also be linear in the number of nodes.
- Multiply connected networks
    - NP-hard

Outline

- Quantifying uncertainty
  - Inference using full joint distributions
  - Bayes’ rule
  - The Wumpus world revisited
- Probabilistic programming
  - Bayesian networks
  - Exact Inference
  - Approximate inference
    - Direct sampling
    - Rejection sampling, Weight sampling
    - Markov Chain Monte Carlo

Approximate Inference for Bayesian Networks

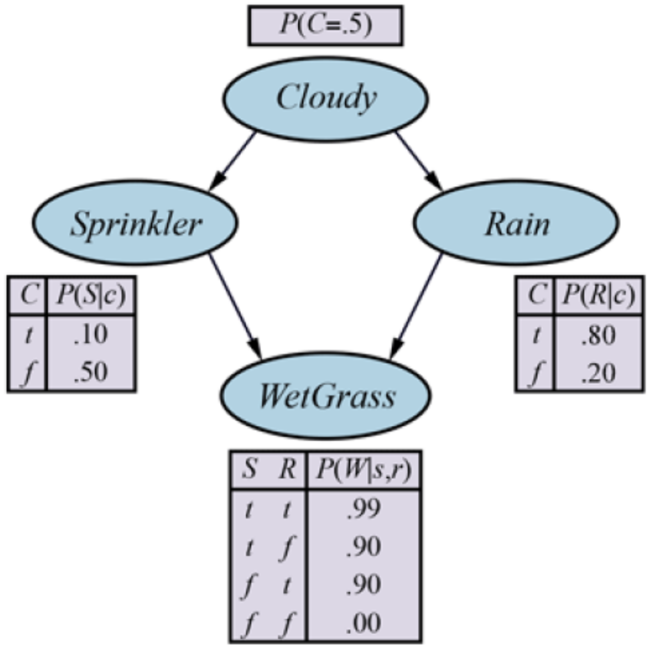
- Given the intractability of exact inference in large networks, we will now consider approximate inference methods.
- Randomized sampling algorithms, called Monte Carlo algorithms, provide approximate answers whose accuracy depends on the number of samples generated.
- Direct sampling
- Markov Chain sampling



- Basic idea
  - Draw N samples from a sampling distribution S
  - Compute an approximate posterior probability $\hat{P}$
  - Show this converges to the true probability P ▮

Direct sampling

- The primitive element in any sampling algorithm is the generation of samples from a known probability distribution.
  - For example, an unbiased coin can be thought of as a random variable with values <heads, tails> and a prior distribution <0.5, 0.5>.
  - Sampling this distribution gives 50% chance of heads and 50% of tails



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

A random sampling process for a Bayes net that has **no evidence** associated with it.
The idea is to sample each variable in turn, in **topological order**.

1. Sample from $\mathbf{P}(Cloudy) = \langle 0.5,0.5\rangle$, value is *true*.
2. Sample from $\mathbf{P}(Sprinkler|Cloudy = true) = \langle 0.1,0.9\rangle$, value is *false*.
3. Sample from $\mathbf{P}(Rain|Cloudy = true) = \langle 0.8,0.2\rangle$, value is *true*.
4. Sample from $\mathbf{P}(WetGrass|Sprinkler = false, Rain = true) = \langle 0.9,0.1\rangle$, value is *true*.

Cloudy, Sprinkler, Rain, WetGrass: [true, false, true, true]

- PRIOR-SAMPLE generates samples from the prior joint distribution specified by the network

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

Figure 13.16

```
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 $\mathbf{x}$
```

sampling algorithm that generates events from a Bayesian network. Each variable is sampled according to the conditional distribution given the values already sampled for the variable's parents.

$$\lim_{N\to\infty} \frac{N_{PS}(x_1,\ldots,x_n)}{N} = S_{PS}(x_1,\ldots,x_n) = P(x_1,\ldots,x_n).$$

For example, consider the event produced earlier: $[true,false,true,true]$. The sampling probability for this event is

$$S_{PS}(true,false,true,true) = 0.5 \times 0.9 \times 0.8 \times 0.9 = 0.324.$$

Hence, in the limit of large N, we expect 32.4% of the samples to be of this event.

Outline

- Quantifying uncertainty
    - Inference using full joint distributions
    - Bayes' rule
    - The Wumpus world revisited
- Probabilistic programming
    - Bayesian networks
    - Exact Inference
    - Approximate inference
        - Direct sampling
        - Rejection sampling, Weight sampling
        - Markov Chain Monte Carlo

Rejection sampling

- With evidence: reject samples disagreeing with evidence
  - First, it generates samples from the prior distribution specified by the network.
  - Then, it rejects all those that do not match the evidence.
- Often used in determine conditional probabilities P(X|e)
- For example, given an evidence Sprinkler=true, estimate P(Rain|Sprinkler=true)

  - E.g., estimate $\mathbf{P}(Rain|Sprinkler = true)$ using 100 samples
    27 samples have $Sprinkler = true$
    Of these, 8 have $Rain = true$ and 19 have $Rain = false$
  - $\hat{\mathbf{P}}(Rain|Sprinkler = true) = \textsc{Normalize}(\langle 8, 19 \rangle) = \langle 0.296, 0.704 \rangle$

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

$$\hat{\mathbf{P}}(X|\mathbf{e}) \approx \frac{\mathbf{P}(X,\mathbf{e})}{P(\mathbf{e})} = \mathbf{P}(X|\mathbf{e}).$$

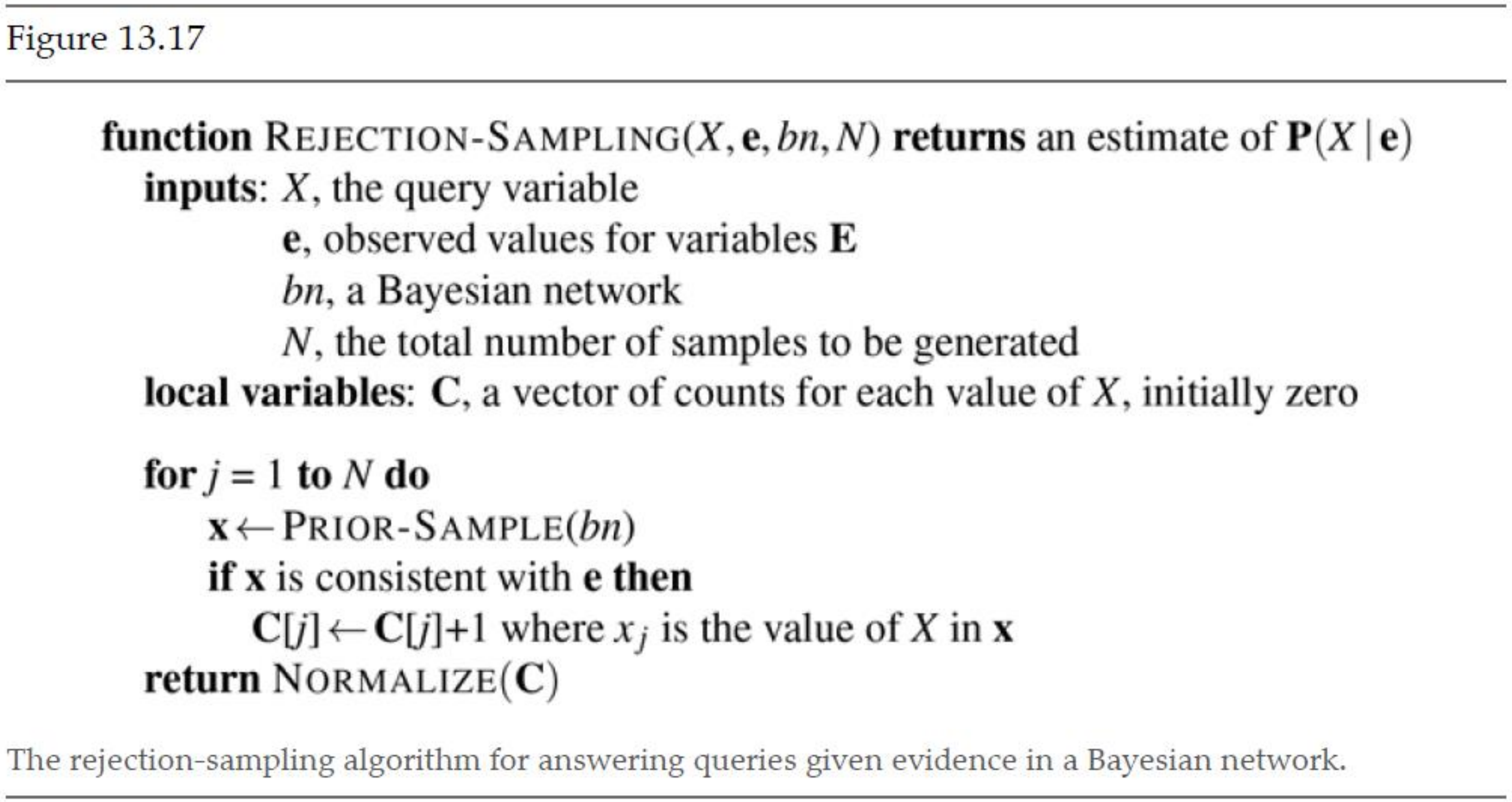


Figure 13.17

```
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($\mathbf{C}$)
```

The rejection-sampling algorithm for answering queries given evidence in a Bayesian network.

Now we know that rejection sampling converges to the correct answer, the next question is,
how fast does that happen?

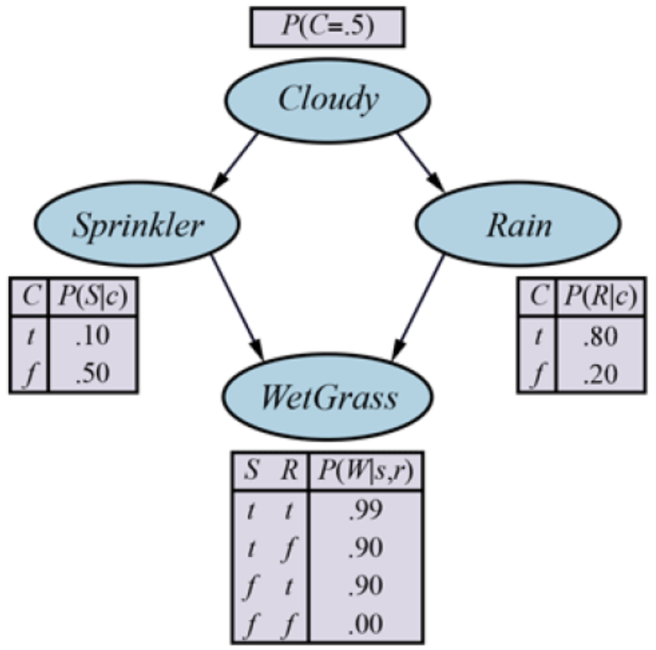
The complexity of rejection sampling depends primarily on the fraction of samples that are accepted.
This fraction is exactly equal to the prior probability of the evidence P(e).

Unfortunately, for complex problems with many evidence variables, this fraction is vanishingly small, e.g.,
the car insurance network.

Likelihood sampling

P(Rain|Cloudy = true, WetGrass = true)

- Fix evidence variables, sample only nonevidence variables.
- Each sample has a **weight** that is the likelihood accords the evidence.



1. *Cloudy* is an evidence variable with value *true*. Therefore, we set

$$w \leftarrow w \times P(Cloudy = true) = 0.5.$$

2. *Sprinkler* is not an evidence variable, so sample from $\mathbf{P}(Sprinkler|Cloudy = true) = \langle 0.1, 0.9 \rangle$; suppose this returns *false*.

3. *Rain* is not an evidence variable, so sample from $\mathbf{P}(Rain|Cloudy = true) = \langle 0.8, 0.2 \rangle$; suppose this returns *true*.

4. *WetGrass* is an evidence variable with value *true*. Therefore, we set

$$\begin{aligned} w &\leftarrow w \times P(WetGrass = true|Sprinkler = false, Rain = true) \\ &= 0.5 \times 0.9 = 0.45. \end{aligned}$$

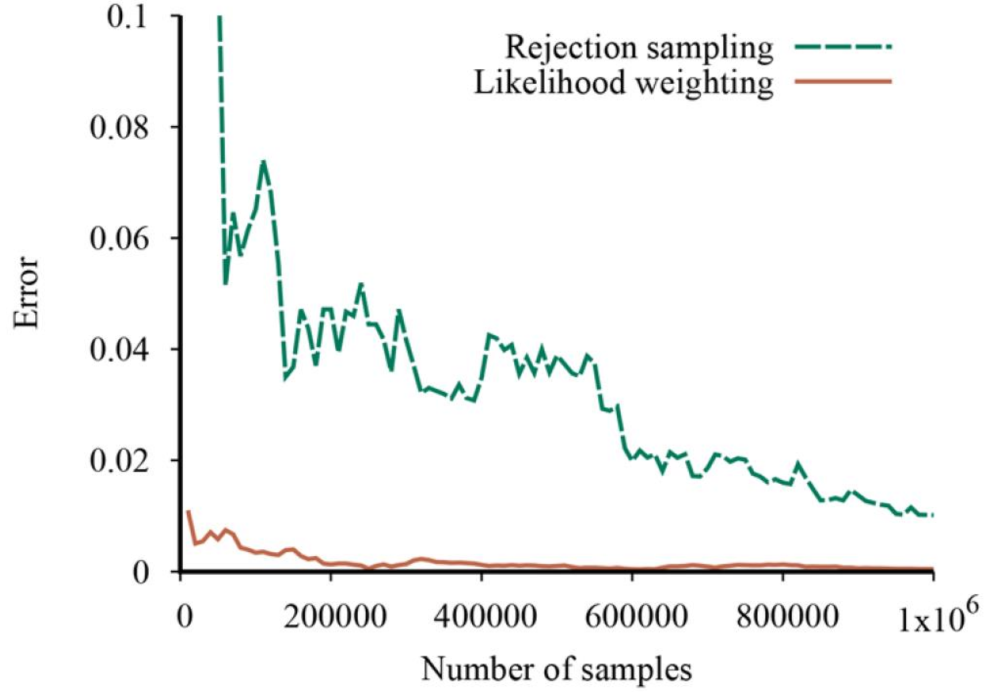
Figure 13.18

```
function LIKELIHOOD-WEIGHTING($X, \mathbf{e}, bn, N$) 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 specifying joint distribution $\mathbf{P}(X_1, \ldots, X_n)$
          $N$, the total number of samples to be generated
  local variables: $\mathbf{W}$, a vector of weighted counts for each value of $X$, initially zero

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

function WEIGHTED-SAMPLE($bn, \mathbf{e}$) returns an event and a weight
  $w \leftarrow 1$; $\mathbf{x} \leftarrow$ an event with $n$ elements, with values fixed from $\mathbf{e}$
  for $i = 1$ to $n$ do
      if $X_i$ is an evidence variable with value $x_{ij}$ in $\mathbf{e}$
          then $w \leftarrow w \times P(X_i = x_{ij} \mid parents(X_i))$
          else $\mathbf{x}[i] \leftarrow$ a random sample from $\mathbf{P}(X_i \mid parents(X_i))$
  return $\mathbf{x}$, $w$
```

The likelihood-weighting algorithm for inference in Bayesian networks. In WEIGHTED-SAMPLE, each nonevidence variable is sampled according to the conditional distribution given the values already sampled for the variable's parents, while a weight is accumulated based on the likelihood for each evidence variable.



Outline

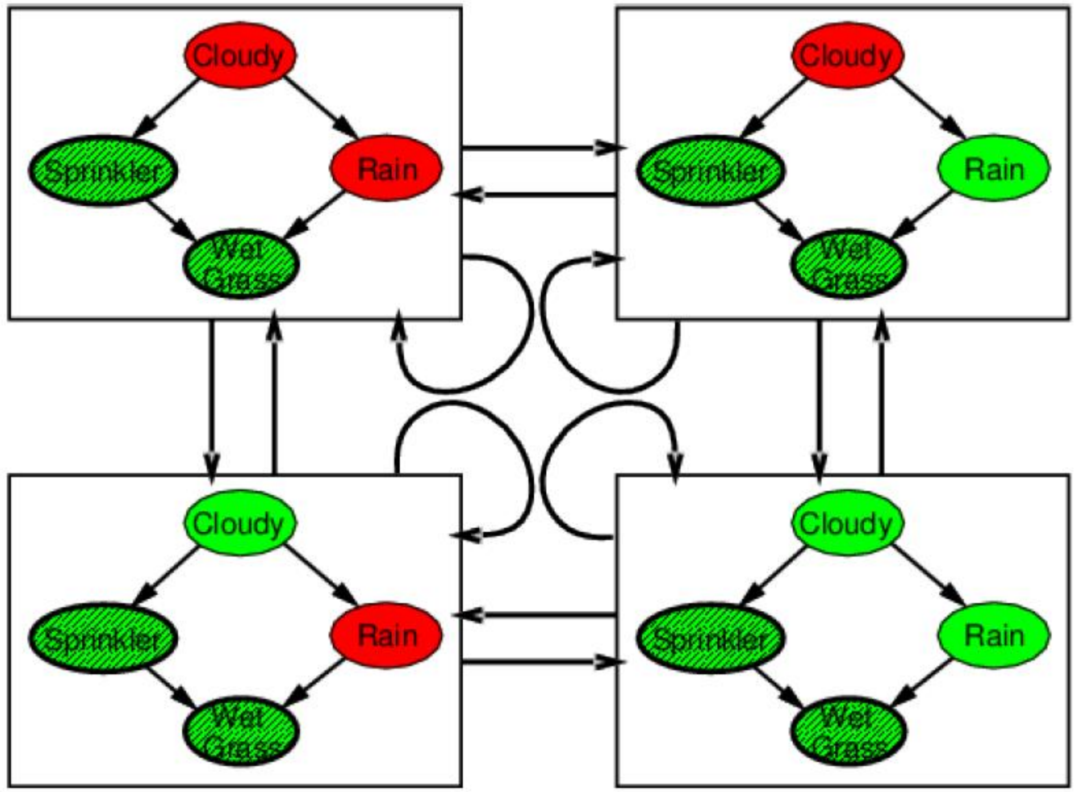
- Quantifying uncertainty
  - Inference using full joint distributions
  - Bayes' rule
  - The Wumpus world revisited
- Probabilistic programming
  - Bayesian networks
  - Exact Inference
  - Approximate inference
    - Direct sampling
    - Rejection sampling, Weight sampling
    - Markov Chain Monte Carlo

Inference by Markov chain simulation

- Markov Chain Monte Carlo (MCMC)
- Instead of generating each sample from scratch, MCMC algorithms generate a sample by making a random change to the preceding sample.
- Think of an MCMC algorithm as being in a particular current state that specifies a value for every variable and generating a next state by making random changes to the current state.
- Gibbs Sampling
    - starts with an arbitrary state (with the evidence variables fixed at their observed values) and generates a next state by randomly sampling a value for one of the nonevidence variables.

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

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



- Wander about for a while, average what you see

Lecture 10 ILOs

- Quantifying uncertainty
  - Inference using full joint distributions
  - Bayes’ rule
  - The Wumpus world revisited
- Probabilistic programming
  - Bayesian networks
  - Exact Inference
  - Approximate inference
    - Direct sampling
    - Rejection sampling, Weight sampling
    - Markov Chain Monte Carlo