INFECTION IN BAYESIAN NETWORKS

CHAPTER 14.4–5
Outline

♦ Exact inference by enumeration
♦ Exact inference by variable elimination
♦ Approximate inference by stochastic simulation
♦ Approximate inference by Markov chain Monte Carlo
Inference tasks

Simple queries: compute posterior marginal $P(X_i|E = e)$
  e.g., $P(\text{NoGas}|\text{Gauge} = \text{empty}, \text{Lights} = \text{on}, \text{Starts} = \text{false})$

Conjunctive queries: $P(X_i, X_j|E = e) = P(X_i|E = e)P(X_j|X_i, E = e)$

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

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:
\[ P(B|j, m) = \frac{P(B, j, m)}{P(j, m)} = \alpha P(B, j, m) = \alpha \sum_e \sum_a P(B, e, a, j, m) \]

Rewrite full joint entries using product of CPT entries:
\[ P(B|j, m) = \alpha \sum_e \sum_a P(B)P(e)P(a|B, e)P(j|a)P(m|a) = \alpha P(B) \sum_e P(e) \sum_a P(a|B, e)P(j|a)P(m|a) \]

Recursive depth-first enumeration: \( O(n) \) space, \( O(d^n) \) time
function \textsc{Enumeration-Ask}(X, e, bn) returns a distribution over $X$

\textbf{inputs:} $X$, the query variable  
\hspace{9pt} $e$, observed values for variables $E$  
\hspace{9pt} $bn$, a Bayesian network with variables \{X\} $\cup$ $E$ $\cup$ $Y$

$Q(X) \leftarrow$ a distribution over $X$, initially empty  
\textbf{for each} value $x_i$ of $X$ \textbf{do}  
\hspace{12pt} extend $e$ with value $x_i$ for $X$  
\hspace{12pt} $Q(x_i) \leftarrow$ \textsc{Enumerate-All}(\textsc{Vars}[bn], $e$)  
\textbf{return} \textsc{Normalize}($Q(X)$)

\textbf{function \textsc{Enumerate-All}(vars, e) returns} a real number

\textbf{if} \textsc{Empty?}(vars) \textbf{then return} 1.0

$Y \leftarrow$ \textsc{First}(vars)

\textbf{if} $Y$ has value $y$ in $e$  
\hspace{12pt} \textbf{then return} $P(y \mid Pa(Y)) \times$ \textsc{Enumerate-All}(\textsc{Rest}(vars), $e$)  
\hspace{12pt} \textbf{else return} $\sum_y P(y \mid Pa(Y)) \times$ \textsc{Enumerate-All}(\textsc{Rest}(vars), $e_y$)  
\hspace{12pt} where $e_y$ is $e$ extended with $Y = y$
Evaluation tree

P(b) = 0.001
P(e) = 0.002
P(¬e) = 0.998
P(a|b,e) = 0.95
P(¬a|b,e) = 0.05
P(a|b,¬e) = 0.94
P(¬a|b,¬e) = 0.06
P(j|a) = 0.90
P(j|¬a) = 0.05
P(j|a) = 0.90
P(m|a) = 0.70
P(m|¬a) = 0.01
P(m|a) = 0.70
P(m|¬a) = 0.01

Enumeration is inefficient: repeated computation
e.g., computes \( P(j|a) P(m|a) \) for each value of \( e \)
Inference by variable elimination

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

\[
P(B|j, m) = \alpha P(B) \sum_e P(e) \sum_a P(a|B, e) P(j|a) P(m|a) \\
= \alpha P(B) \sum_e P(e) \sum_a P(a|B, e) P(j|a) f_M(a) \\
= \alpha P(B) \sum_e P(e) \sum_a f_A(a, b, e) f_J(a) f_M(a) \\
= \alpha P(B) \sum_e P(e) f_{\bar{A}JM}(b, e) \quad \text{(sum out } A) \\
= \alpha P(B) f_{\bar{E}\bar{A}JM}(b) \quad \text{(sum out } E) \\
= \alpha f_B(b) \times f_{\bar{E}\bar{A}JM}(b)
\]
**Variable elimination: Basic operations**

**Summing out** a variable from a product of factors:
- move any constant factors outside the summation
- add up submatrices in pointwise product of remaining factors

\[
\sum_x f_1 \times \cdots \times f_k = f_1 \times \cdots \times f_i \sum_x f_{i+1} \times \cdots \times f_k = f_1 \times \cdots \times f_i \times f_{\bar{X}}
\]

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., \( f_1(a, b) \times f_2(b, c) = f(a, b, c) \)
Variable elimination algorithm

```plaintext
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 P(X₁, . . . , Xₙ)
    factors ← []; vars ← Reverse(Vars[bn])
    for each var in vars do
        factors ← [Make-Factor(var, e)|factors]
        if var is a hidden variable then factors ← Sum-Out(var, factors)
    return Normalize(Pointwise-Product(factors))
```
Irrelevant variables

Consider the query $P(JohnCalls|Burglary = \text{true})$

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

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

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

Here, $X = JohnCalls$, $E = \{Burglary\}$, and
$\text{Ancestors}({X} \cup E) = \{\text{Alarm}, \text{Earthquake}\}$
so $MaryCalls$ is irrelevant

(Compare this to backward chaining from the query in Horn clause KBs)
Defn: moral graph of Bayes net: marry all parents and drop arrows
Defn: A is m-separated from B by C iff separated by C in the moral graph
Thm 2: Y is irrelevant if m-separated from X by E

For \( P(\text{JohnCalls}|\text{Alarm} = \text{true}) \), both Burglary and Earthquake are irrelevant
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 $\Rightarrow$ NP-hard
- equivalent to counting 3SAT models $\Rightarrow$ $\#P$-complete

1. $A \lor B \lor C$
2. $C \lor D \lor \neg A$
3. $B \lor C \lor \neg D$
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 $P$

Outline:
- Sampling from an empty network
- Rejection sampling: reject samples disagreeing with evidence
- Likelihood weighting: use evidence to weight samples
- Markov chain Monte Carlo (MCMC): sample from a stochastic process whose stationary distribution is the true posterior
Sampling from an empty network

**function** Prior-Sample(*bn*) **returns** an event sampled from *bn*

**inputs**: *bn*, a belief network specifying joint distribution \( P(X_1, \ldots, X_n) \)

\[ x \leftarrow \text{an event with } n \text{ elements} \]

**for** \( i = 1 \) **to** \( n \) **do**

\[ x_i \leftarrow \text{a random sample from } P(X_i | \text{parents}(X_i)) \]

\( \text{given the values of } Parents(X_i) \text{ in } x \)

**return** \( x \)
Example

\[
P(C) = 0.50
\]

\[
P(S|C) =
\begin{array}{c|c}
C & P(S|C) \\
\hline
T & 0.10 \\
F & 0.50 \\
\end{array}
\]

\[
P(R|C) =
\begin{array}{c|c}
C & P(R|C) \\
\hline
T & 0.80 \\
F & 0.20 \\
\end{array}
\]

\[
P(W|S,R) =
\begin{array}{c|c|c}
S & R & P(W|S,R) \\
\hline
T & T & 0.99 \\
T & F & 0.90 \\
F & T & 0.90 \\
F & F & 0.01 \\
\end{array}
\]
Example

- **Cloudy**
  - \( P(C) = 0.50 \)

### Conditional Probabilities

| C | \( P(S|C) \) |
|---|-------------|
| T | 0.10        |
| F | 0.50        |

| C | \( P(R|C) \) |
|---|-------------|
| T | 0.80        |
| F | 0.20        |

| S | R | \( P(W|S,R) \) |
|---|---|----------------|
| T | T | 0.99           |
| T | F | 0.90           |
| F | T | 0.90           |
| F | F | 0.01           |

Chapter 14.4–5  16
Example

\begin{table}[h]
\centering
\begin{tabular}{|c|c|}
\hline
C & P(S|C) \\
\hline
T & .10 \\
F & .50 \\
\hline
\end{tabular}
\end{table}

\begin{table}[h]
\centering
\begin{tabular}{|c|c|}
\hline
C & P(R|C) \\
\hline
T & .80 \\
F & .20 \\
\hline
\end{tabular}
\end{table}

\begin{table}[h]
\centering
\begin{tabular}{|c|c|}
\hline
S & R & P(W|S,R) \\
\hline
T & T & .99 \\
T & F & .90 \\
F & T & .90 \\
F & F & .01 \\
\hline
\end{tabular}
\end{table}
### Example

#### Probabilities:
- $P(C) = 0.50$
- $P(R|C) = 0.80$
- $P(S|C) = 0.10$
- $P(R|C) = 0.20$

#### Conditional Probabilities:

| S | R | $P(W|S,R)$ |
|---|---|------------|
| T | T | 0.99       |
| T | F | 0.90       |
| F | T | 0.90       |
| F | F | 0.01       |

#### Variables:
- Cloudy
- Sprinkler
- Wet Grass
- Rain
Example

\[ P(C) = 0.50 \]

\[ P(S|C) \]

| C | P(S|C) |
|---|-------|
| T | 0.10  |
| F | 0.50  |

\[ P(R|C) \]

| C | P(R|C) |
|---|-------|
| T | 0.80  |
| F | 0.20  |

\[ P(W|S,R) \]

| S | R | P(W|S,R) |
|---|---|---------|
| T | T | 0.99    |
| T | F | 0.90    |
| F | T | 0.90    |
| F | F | 0.01    |
Example

| C | P(S|C) |
|---|-------|
| T | 0.10  |
| F | 0.50  |

| C | P(R|C) |
|---|-------|
| T | 0.80  |
| F | 0.20  |

| S | R | P(W|S,R) |
|---|---|---------|
| T | T | 0.99    |
| T | F | 0.90    |
| F | T | 0.90    |
| F | F | 0.01    |
Example

\[ P(C) \]
\[ .50 \]

\[ P(S|C) \]
\[ \begin{array}{ll}
T & .10 \\
F & .50 \\
\end{array} \]

\[ P(R|C) \]
\[ \begin{array}{ll}
T & .80 \\
F & .20 \\
\end{array} \]

\[ P(W|S,R) \]
\[ \begin{array}{ccc}
S & R & P(W|S,R) \\
TT & .99 & \\
TF & .90 & \\
FT & .90 & \\
FF & .01 & \\
\end{array} \]
Sampling from an empty network contd.

Probability that PriorSample generates a particular event

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

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 we have

\[
\lim_{N \to \infty} \hat{P}(x_1, \ldots, x_n) = \lim_{N \to \infty} \frac{N_{PS}(x_1, \ldots, x_n)}{N} \\
= S_{PS}(x_1, \ldots, x_n) \\
= P(x_1 \ldots x_n)
\]

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**

\( \hat{P}(X|e) \) estimated from samples agreeing with \( e \)

```plaintext
function REJECTION-SAMPLING(\( X, e, bn, N \)) returns an estimate of \( P(X|e) \)

local variables: \( N \), a vector of counts over \( X \), initially zero

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

return \( \text{NORMALIZE}(N[X]) \)
```

E.g., estimate \( P(Rain|Sprinkler = true) \) using 100 samples
27 samples have \( Sprinkler = true \)
Of these, 8 have \( Rain = true \) and 19 have \( Rain = false \).

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

Similar to a basic real-world empirical estimation procedure
Analysis of rejection sampling

\[ \hat{P}(X|e) = \alpha N_{PS}(X, e) \quad \text{(algorithm defn.)} \]
\[ = N_{PS}(X, e)/N_{PS}(e) \quad \text{(normalized by } N_{PS}(e)\text{)} \]
\[ \approx P(X, e)/P(e) \quad \text{(property of PRIORSAMPLE)} \]
\[ = P(X|e) \quad \text{(defn. of conditional probability)} \]

Hence rejection sampling returns consistent posterior estimates

Problem: hopelessly expensive if \( P(e) \) is small

\( P(e) \) drops off exponentially with number of evidence variables!
Likelihood weighting

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

**function** LIKELIHOOD-WEIGHTING($X, e, bn, N$) **returns** an estimate of $P(X|e)$

**local variables:** $W$, a vector of weighted counts over $X$, initially zero

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

**return** $\text{Normalize}(W[X])$

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

$x \leftarrow$ 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 $w \leftarrow w \times P(X_i = x_i | \text{parents}(X_i))$
        else $x_i \leftarrow$ a random sample from $P(X_i | \text{parents}(X_i))$

**return** $x, w$
Likelihood weighting example

$\begin{array}{c|c}
\text{C} & \text{P(S|C)} \\
\hline
\text{T} & .10 \\
\text{F} & .50 \\
\end{array}$

$\begin{array}{c|c}
\text{C} & \text{P(R|C)} \\
\hline
\text{T} & .80 \\
\text{F} & .20 \\
\end{array}$

$\begin{array}{c|c|c}
\text{S} & \text{R} & \text{P(W|S,R)} \\
\hline
\text{T} & \text{T} & .99 \\
\text{T} & \text{F} & .90 \\
\text{F} & \text{T} & .90 \\
\text{F} & \text{F} & .01 \\
\end{array}$

$w = 1.0$
Likelihood weighting example

\[ P(C) \]
- \[ w = 1.0 \]

\[
\begin{array}{c|c}
C & P(S|C) \\
\hline
T & .10 \\
F & .50 \\
\end{array}
\]

\[
\begin{array}{c|c}
C & P(R|C) \\
\hline
T & .80 \\
F & .20 \\
\end{array}
\]

\[
\begin{array}{c|c|c}
S & R & P(W|S,R) \\
\hline
T & T & .99 \\
T & F & .90 \\
F & T & .90 \\
F & F & .01 \\
\end{array}
\]
Likelihood weighting example

\[
\begin{array}{c|c}
\text{C} & \text{P(S|C)} \\
\hline
\text{T} & .10 \\
\text{F} & .50 \\
\end{array}
\]

\[
\begin{array}{c|c}
\text{C} & \text{P(R|C)} \\
\hline
\text{T} & .80 \\
\text{F} & .20 \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{S} & \text{R} & \text{P(W|S,R)} \\
\hline
\text{T} & \text{T} & .99 \\
\text{T} & \text{F} & .90 \\
\text{F} & \text{T} & .90 \\
\text{F} & \text{F} & .01 \\
\end{array}
\]

\[w = 1.0\]
Likelihood weighting example

\[
w = 1.0 \times 0.1
\]
$w = 1.0 \times 0.1$
Likelihood weighting example

\[ C \quad P(S|C) \quad C \quad P(R|C) \]

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>.10</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>.50</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>.80</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>.20</td>
<td></td>
</tr>
</tbody>
</table>

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

\[ w = 1.0 \times 0.1 \]
Likelihood weighting example

| C | P(S|C) | S  | R  | P(W|S,R) |
|---|-------|----|----|----------|
| T | .10   | T  | T  | .99      |
| F | .50   | T  | F  | .90      |
|   |       | F  | T  | .90      |
|   |       | F  | F  | .01      |

\[ w = 1.0 \times 0.1 \times 0.99 = 0.099 \]
Likelihood weighting analysis

Sampling probability for WeightedSample is
\[ S_{WS}(z, e) = \prod_{i=1}^{l} P(z_i|\text{parents}(Z_i)) \]
Note: pays attention to evidence in ancestors only
\[ \Rightarrow \text{somewhere “in between” prior and posterior distribution} \]

Weight for a given sample \( z, e \) is
\[ w(z, e) = \prod_{i=1}^{m} P(e_i|\text{parents}(E_i)) \]

Weighted sampling probability is
\[ S_{WS}(z, e)w(z, e) = \prod_{i=1}^{l} P(z_i|\text{parents}(Z_i)) \prod_{i=1}^{m} P(e_i|\text{parents}(E_i)) \]
\[ = P(z, e) \text{ (by standard global semantics of network)} \]

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|e)
    local variables: N[X], a vector of counts over 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 Y
    for j = 1 to N do
        for each Z_i in Z do
            sample the value of Z_i in x from P(Z_i|mb(Z_i))
            given the values of MB(Z_i) in x
            N[x] ← N[x] + 1 where x is the value of X in x
    return Normalize(N[X])
```

Can also choose a variable to sample at random each time
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 $P(Rain|Sprinkler = true, WetGrass = true)$

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

E.g., visit 100 states

31 have $Rain = true$, 69 have $Rain = false$

$\hat{P}(Rain|Sprinkler = true, WetGrass = true)$

$= \text{Normalize}(\langle 31, 69 \rangle) = \langle 0.31, 0.69 \rangle$

Theorem: chain approaches **stationary distribution**: 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 and Rain*

Markov blanket of *Rain is* 
*Cloudy, Sprinkler, and WetGrass*

Probability given the Markov blanket is calculated as follows:

\[
P(x'_i | mb(X_i)) = P(x'_i | parents(X_i)) \prod_{Z_j \in \text{Children}(X_i)} P(z_j | parents(Z_j))
\]

Easily implemented in message-passing parallel systems, brains

Main computational problems:

1) Difficult to tell if convergence has been achieved
2) Can be wasteful if Markov blanket is large:

\[
P(X_i | mb(X_i)) \text{ won’t change much (law of large numbers)}
\]
Summary

Exact inference by variable elimination:
- polytime on polytrees, NP-hard on general graphs
- space = time, very sensitive to topology

Approximate inference by LW, MCMC:
- LW does poorly when there is lots of (downstream) evidence
- LW, MCMC generally insensitive to topology
- Convergence can be very slow with probabilities close to 1 or 0
- Can handle arbitrary combinations of discrete and continuous variables