Bayes Nets

CS 486/686: Introduction to Artificial Intelligence
Outline

• Inference in Bayes Nets
• Variable Elimination
Inference in Bayes Nets

• Independence allows us to compute prior and posterior probabilities quite effectively

• We will start with a couple simple examples

  - Networks without loops
    - A loop is a cycle in the underlying undirected graph
Forward Inference

Note: all (final) terms are CPTs in the BN
Note: only ancestors of J considered

\[ P(J) = \]
Forward Inference with “Upstream Evidence”

\[ P(J \mid ET) = \]
Forward Inference with Multiple Parents

P(Fev)=?
Forward Inference with Evidence

\[ P(\text{Fev}|ts,\sim m)=? \]
Simple Backward Inference

• When evidence is downstream of a query variable, must reason “backwards”. This requires Bayes Rule

\[ P(ET|j) = \]
Backward Inference

- Same idea applies when several pieces of evidence lie “downstream”

\[ P(ET|j,fev) = ? \]
Variable Elimination

What about general BN?

\[ P(H|A,F) = ? \]
Variable Elimination

• Simply applies the summing-out rule (marginalization) repeatedly

• Exploits independence in network and distributes the sum inward
  - Basically doing dynamic programming
Factors

• A function $f(X_1,\ldots,X_k)$ is called a factor
  - View this as a table of numbers, one for each instantiation of the variables
  - Exponential in $k$

• Each CPT in a BN is a factor
  - $P(C|A,B)$ is a function of 3 variables, $A$, $B$, $C$
    - Represented as $f(A,B,C)$

• Notation: $f(X,Y)$ denotes a factor over variables $X \cup Y$
  - $X$ and $Y$ are sets of variables
Product of Two Factors

• Let $f(X,Y)$ and $g(Y,Z)$ be two factors with variables $Y$ in common

• The product of $f$ and $g$, denoted by $h=fg$ is
  - $h(X,Y,Z)=f(X,Y) \times g(Y,Z)$

<table>
<thead>
<tr>
<th>$f(A,B)$</th>
<th>$g(B,C)$</th>
<th>$h(A,B,C)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ab$</td>
<td>$bc$</td>
<td>$0.63$</td>
</tr>
<tr>
<td>$a~b$</td>
<td>$b~c$</td>
<td>$0.08$</td>
</tr>
<tr>
<td>$a~b$</td>
<td>$b~c$</td>
<td>$0.28$</td>
</tr>
<tr>
<td>$a~b$</td>
<td>$b~c$</td>
<td>$0.48$</td>
</tr>
<tr>
<td>$a~b$</td>
<td>$b~c$</td>
<td>$0.27$</td>
</tr>
<tr>
<td>$a~b$</td>
<td>$b~c$</td>
<td>$0.02$</td>
</tr>
<tr>
<td>$a~b$</td>
<td>$b~c$</td>
<td>$0.12$</td>
</tr>
<tr>
<td>$a~b$</td>
<td>$b~c$</td>
<td>$0.12$</td>
</tr>
</tbody>
</table>
Summing a Variable Out of a Factor

• Let $f(X, Y)$ be a factor with variable $X$ and variable set $Y$

• We sum out variable $X$ from $f$ to produce $h=\sum_x f$ where $h(Y)=\sum_{x \in \text{Dom}(X)} f(x, Y)$

<table>
<thead>
<tr>
<th>f(A,B)</th>
<th>h(B)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ab</td>
<td>b</td>
</tr>
<tr>
<td>0.9</td>
<td>1.3</td>
</tr>
<tr>
<td>a~b</td>
<td>~b</td>
</tr>
<tr>
<td>0.1</td>
<td>0.7</td>
</tr>
<tr>
<td>~ab</td>
<td></td>
</tr>
<tr>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td><del>a</del>b</td>
<td></td>
</tr>
<tr>
<td>0.6</td>
<td></td>
</tr>
</tbody>
</table>
Restricting a Factor

• Let $f(X, Y)$ be a factor with variable $X$

• We restrict factor $f$ to $X=x$ by setting $X$ to the value $x$ and “deleting”. Define $h=f_{X=x}$ as: $h(Y)=f(x, Y)$

<table>
<thead>
<tr>
<th>f(A,B)</th>
<th>h(B) = f_{A=a}</th>
</tr>
</thead>
<tbody>
<tr>
<td>ab</td>
<td>0.9</td>
</tr>
<tr>
<td>b</td>
<td>0.9</td>
</tr>
<tr>
<td>a~b</td>
<td>0.1</td>
</tr>
<tr>
<td>~b</td>
<td>0.1</td>
</tr>
<tr>
<td>~ab</td>
<td>0.4</td>
</tr>
<tr>
<td><del>a</del>b</td>
<td>0.6</td>
</tr>
</tbody>
</table>
Variable Elimination: No Evidence

• Computing prior probability of query variable $X$ can be seen as applying these operations on factors

$$\begin{align*}
P(C) &= \sum_{A,B} P(C|B) \sum_{A} P(B|A) P(A) \\
&= \sum_{B} P(C|B) \sum_{A} f_2(A,B) f_1(A) \\
&= \sum_{B} f_3(B,C) f_4(B) \\
&= f_5(C)
\end{align*}$$

Define new factors: $f_4(B) = \sum_{A} f_2(A,B) f_1(A)$ and $f_5(C) = \sum_{B} f_3(B,C) f_4(B)$
Variable Elimination: No Evidence

\[ f_1(A) \quad f_2(A,B) \quad f_3(B,C) \]

<table>
<thead>
<tr>
<th>f_1(A)</th>
<th>f_2(A,B)</th>
<th>f_3(B,C)</th>
<th>f_4(B)</th>
<th>f_5(C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>ab</td>
<td>bc</td>
<td>b</td>
<td>c</td>
</tr>
<tr>
<td>~a</td>
<td>a~b</td>
<td>b~c</td>
<td>~b</td>
<td>~c</td>
</tr>
<tr>
<td>~ab</td>
<td><del>a</del>b</td>
<td>~bc</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><del>a</del>b</td>
<td><del>b</del>c</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Variable Elimination: No Evidence

\[ P(D) = \sum_{A,B,C} P(D|C) P(C|B,A) P(B) P(A) \]
\[ = \sum_C f_4(C,D) \sum_B f_2(B) \sum_A f_3(A,B,C) f_1(A) \]
\[ = \sum_C f_4(C,D) \sum_B f_2(B) f_5(B,C) \]
\[ = \sum_C f_4(C,D) f_6(C) \]
\[ = f_7(D) \]

Define new factors: \( f_5(B,C), f_6(C), f_7(D) \), in the obvious way.
Variable Elimination: One View

• Write out desired computation using chain rule, exploiting independence relations in networks
• Arrange terms in convenient fashion
• Distribution each sum (over each variable) in as far as it will go
• Apply operations “inside out”, repeatedly elimination and creating new factors
  - Note that each step eliminates a variable
The Algorithm

• Given query variable Q, remaining variables Z. Let F be the set of factors corresponding to CPTs for \{Q\} \cup Z.

1. Choose an elimination ordering Z_1, ..., Z_n of variables in Z.
2. For each Z_j -- in the order given -- eliminate Z_j \in Z as follows:
   (a) Compute new factor g_j = \sum_{Z_j} f_1 \times f_2 \times ... \times f_k, where the f_i are the factors in F that include Z_j
   (b) Remove the factors f_i (that mention Z_j) from F and add new factor g_j to F
3. The remaining factors refer only to the query variable Q. Take their product and normalize to produce P(Q)
**Example Again**

<table>
<thead>
<tr>
<th>Factors:</th>
<th>$f_1(A)$ $f_2(B)$ $f_3(A,B,C)$ $f_4(C,D)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query:</td>
<td>$P(D)$?</td>
</tr>
<tr>
<td>Elim. Order:</td>
<td>A, B, C</td>
</tr>
</tbody>
</table>

---

Step 1: Add $f_5(B,C) = \sum_A f_3(A,B,C) f_1(A)$
Remove: $f_1(A), f_3(A,B,C)$

Step 2: Add $f_6(C) = \sum_B f_2(B) f_5(B,C)$
Remove: $f_2(B), f_5(B,C)$

Step 3: Add $f_7(D) = \sum_C f_4(C,D) f_6(C)$
Remove: $f_4(C,D), f_6(C)$

Last factor $f_7(D)$ is (possibly unnormalized) probability $P(D)$
Variable Elimination: Evidence

- Computing posterior of query variable given evidence is similar; suppose we observe C=c:

\[ P(A|c) = \frac{P(A) P(c|A)}{\sum_B P(c|B) P(B|A)} = \frac{f_1(A)}{\sum_B f_4(B) f_2(A,B)} = \frac{f_1(A) f_5(A)}{f_6(A)} \]

New factors: \( f_4(B) = f_3(B,c); \quad f_5(A) = \sum_B f_2(A,B) f_4(B); \]
\[ f_6(A) = f_1(A) f_5(A) \]
The Algorithm (with Evidence)

- Given query variable Q, evidence variables E (observed to be e), remaining variables Z. Let F be the set of factors corresponding to CPTs for \{Q\} \cup Z.

1. Replace each factor \( f \in F \) that mentions a variable(s) in E with its restriction \( f_{E=e} \) (somewhat abusing notation).
2. Choose an elimination ordering \( Z_1, \ldots, Z_n \) of variables in Z.
3. Run variable elimination as above.
4. The remaining factors refer only to the query variable Q. Take their product and normalize to produce \( P(Q) \)
Example

Factors: $f_1(A)$, $f_2(B)$, $f_3(A, B, C)$, $f_4(C, D)$
Query: $P(A)$?
Evidence: $D = d$
Elim. Order: C, B
Some Notes on VE

• After each iteration $j$ (elimination of $Z_j$) factors remaining in set $F$ refer only to variables $Z_{j+1},...,Z_n$ and $Q$

- No factor mentions an evidence variable after the initial restriction

• Number of iterations is linear in number of variables
Some Notes on VE

• Complexity is linear in number of variables and exponential in size of the largest factor
  - Recall each factor has exponential size in its number of variables
  - Can’t do any better than size of BN (since its original factors are part of the factor set)
  - When we create new factors, we might make a set of variables larger
Elimination Ordering: Polytrees

- Inference is linear in size of the network
  - Ordering: eliminate only “singly-connected” nodes
  - Result: no factor ever larger than original CPTs
- What happens if we eliminate B first?
Effect of Different Orderings

• Suppose query variable is D. Consider different orderings for this network
  • A,F,H,G,B,C,E: Good
  • E,C,A,B,G,H,F: Bad
• Certain variables have no impact on the query

- In ABC network, computing $P(A)$ with no evidence requires elimination of $B$ and $C$
  - But when you sum out these variables, you compute a trivial factor
  - Eliminating $C$: $g(C) = \sum f(B,C) = \sum_c \Pr(C|B)$.
  - Note that $P(clb) + P(\sim clb) = 1$ and $P(cl\sim b) + P(\sim cl\sim b) = 1$
Relevance: A Sound Approximation

• Can restrict our attention to relevant variables

• Given query Q, evidence E
  - Q is relevant
  - If any node Z is relevant, its parents are relevant
  - If $E \subseteq E$ is a descendant of a relevant node, then E is relevant
Example

- $P(F)$
- $P(\text{FIE})$
- $P(\text{FIE,C})$
Probabilistic Inference

• Applications of BN in AI are virtually limitless

• Examples
  - mobile robot navigation
  - speech recognition
  - medical diagnosis, patient monitoring
  - fault diagnosis (e.g. car repairs)
  - etc
Where do BNs Come From?

• Handcrafted
  - Interact with a domain expert to
    - Identify dependencies among variables (causal structure)
    - Quantify local distributions (CPTs)

• Empirical data, human expertise often used as a guide
Where do BNs Come From?

- Recent emphasis on learning BN from data
  - Input: a set of cases (instantiations of variables)
  - Output: network reflecting empirical distribution
  - Issues: identifying causal structure, missing data, discovery of hidden (unobserved) variables, incorporating prior knowledge (bias) about structure