Where are we?

Last time . . .

- Introduced Bayesian networks
- Allow for compact representation of JPDs
- Methods for efficient representations of CPTs
- But how hard is inference in BNs?

Today . . .

- **Inference in Bayesian networks**
Inference in BNs

- Basic task: compute posterior distribution for set of query variables given some observed event (i.e. assignment of values to evidence variables)
- Formally: determine $P(X|e)$ given query variables $X$, evidence variables $E$ (and non-evidence or hidden variables $Y$)
- Example:
  $P(Burglary|JohnCalls = true, MaryCalls = true) = \langle 0.284, 0.716 \rangle$
- First we will discuss exact algorithms for computing posterior probabilities then approximate methods later
Inference by enumeration

- We have seen that any conditional probability can be computed from a full JPD by summing terms:
  \[ P(X|e) = \alpha P(X, e) = \alpha \sum_y P(X, e, y) \]
- Since BN gives complete representation of full JPD, we must be able to answer a query by computing sums of products of conditional probabilities from the BN.
- Consider query:
  \[ P(Burglary|JohnCalls = true, MaryCalls = true) = P(B|j, m) \]
  \[ P(B|j, m) = \alpha P(B, j, m) = \alpha \sum_e \sum_a P(B, e, a, j, m) \]
Inference by enumeration

- Recall \( P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i | \text{parents}(X_i)) \)
- We can use CPTs to simplify this exploiting BN structure
- For \( \text{Burglary} = \text{true} \):

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

- But we can improve efficiency of this by moving terms outside that don’t depend on sums

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

- To compute this, we need to loop through variables in order and multiply CPT entries; for each summation we need to loop over variable’s possible values
Example

- New burglar alarm has been fitted, fairly reliable but sometimes reacts to earthquakes
- Neighbours John and Mary promise to call when they hear alarm
- John sometimes mistakes phone for alarm, and Mary listens to loud music and sometimes doesn’t hear alarm
The variable elimination algorithm

- Enumeration method is computationally quite hard.
- You often compute the same thing several times; e.g. \( P(j|a)P(m|a) \) and \( P(j|\neg a)P(m|\neg a) \) for each value of \( e \)
- Evaluation of expression shown in the following tree:
The variable elimination algorithm

- Idea of **variable elimination**: avoid repeated calculations
- Basic idea: store results after doing calculation once
- Works bottom-up by evaluating subexpressions
- Assume we want to evaluate

\[
P(B|j, m) = \alpha P(B) \sum_{e} P(e) \sum_{a} P(a|B, e) P(j|a) P(m|a)
\]

- We’ve annotated each part with a **factor**.
- A factor is a **matrix**, indexed with its argument variables. E.g:
  - Factor \( f_5(A) \) corresponds to \( P(m|a) \) and depends just on \( A \) because \( m \) is fixed (it’s a \( 2 \times 1 \) matrix).
    \[ f_5(A) = \langle P(m|a), P(m|\neg a) \rangle \]
  - \( f_3(A, B, E) \) is a \( 2 \times 2 \times 2 \) matrix for \( P(a|B, e) \)
The variable elimination algorithm

\[ P(B|j, m) = \alpha f_1(B) \times \sum_e f_2(E) \sum_a f_3(A, B, E) \times f_4(A) \times f_5(A) \]

- Summing out \( A \) produces a \( 2 \times 2 \) matrix

  (via pointwise product):

  \[
  f_6(B, E) = \sum_a f_3(A, B, E) \times f_4(A) \times f_5(A) \\
  = (f_3(a, B, E) \times f_4(a) \times f_5(a)) + (f_3(\neg a, B, E) \times f_4(\neg a) \times f_5(\neg a))
  \]

- So now we have

  \[ P(B|j, m) = \alpha f_1(B) \times \sum_e f_2(E) \times f_6(B, E) \]

- Sum out \( E \) in the same way:

  \[ f_7(B) = (f_2(e) \times f_6(B, e)) + (f_2(\neg e) \times f_6(B, \neg e)) \]

- Using \( f_1(B) = P(B) \), we can finally compute

  \[ P(B|j, m) = \alpha f_1(B) \times f_7(B) \]

- Remains to define pointwise product and summing out
An example

- Pointwise product yields product for union of variables in its arguments:

\[ f(X_1 \ldots X_i, Y_1 \ldots Y_j, Z_1 \ldots Z_k) = f_1(X_1 \ldots X_i, Y_1 \ldots Y_j)f_2(Y_1 \ldots Y_j, Z_1 \ldots Z_k) \]

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>(f_1(A, B))</th>
<th></th>
<th></th>
<th>(f_2(B, C))</th>
<th></th>
<th></th>
<th>(f(A, B, C))</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>0.3</td>
<td>T</td>
<td>T</td>
<td>0.2</td>
<td>T</td>
<td>T</td>
<td>0.3 (\times) 0.2</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>0.7</td>
<td>T</td>
<td>F</td>
<td>0.8</td>
<td>T</td>
<td>T</td>
<td>0.3 (\times) 0.8</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>0.9</td>
<td>F</td>
<td>T</td>
<td>0.6</td>
<td>T</td>
<td>F</td>
<td>0.7 (\times) 0.6</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>0.1</td>
<td>F</td>
<td>F</td>
<td>0.4</td>
<td>T</td>
<td>F</td>
<td>0.7 (\times) 0.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

- For example \(f(T, T, F) = f_1(T, T) \times f_2(T, F)\)
An example

- Summing out is similarly straightforward
- Trick: any factor that does not depend on the variable to be summed out can be moved outside the summation process
- For example

\[
\sum_e f_2(E) \times f_3(A, B, E) \times f_4(A) \times f_5(A) = f_4(A) \times f_5(A) \times \sum_e f_2(E) \times f_3(A, B, E)
\]

- Matrices are only multiplied when we need to sum out a variable from the accumulated product
Another Example: $P(J|b) = \langle P(j|b), P(\neg j|b) \rangle$

$$
J|b) = \alpha \sum_e \sum_a \sum_m P(J, b, e, a, m) = \alpha \sum_e \sum_a \sum_m P(b)P(e)P(a|b, e)P(J|a)P(m|a) = \alpha' \sum_e P(e) \sum_a P(a|b, e) P(J|a) \sum_m P(m|a) = 1
$$

$$
= \alpha' \sum_e f_1(E) \sum_a f_2(A, E) f_3(J, A) 2 \times 1 \times 2 \times 2
= \alpha' f_4(J, E) 2 \times 1 \times 2 \times 2
= \alpha' f_5(J)
$$

Can eliminate all variables that aren’t ancestors of query or evidence variables!
Summary

- Inference in Bayesian Networks
- Exact methods: enumeration, variable elimination algorithm
- Computationally intractable in the worst case
- Next time: Approximate inference in Bayesian Networks