Announcements

- **Midterm: Wednesday 7pm-9pm**
  - See midterm prep page (posted on Piazza, inst.eecs page)
  - Four rooms; your room determined by *last two digits of your SID*:
    - 00-32: Dwinelle 155
    - 33-45: Genetics and Plant Biology 100
    - 63-99: Pimentel 1
- Discussions this week *by topic*
- **Survey: complete it before midterm; 80% participation = +1pt**
Bayes nets encode joint distributions as product of conditional distributions on each variable:

\[ P(X_1, \ldots, X_n) = \prod_i P(X_i | Parents(X_i)) \]
Conditional independence semantics

- Every variable is conditionally independent of its non-descendants given its parents
- Conditional independence semantics $\iff$ global semantics
Example

- JohnCalls independent of Burglary given Alarm?
  - Yes
- JohnCalls independent of MaryCalls given Alarm?
  - Yes
- Burglary independent of Earthquake?
  - Yes
- Burglary independent of Earthquake given Alarm?
  - NO!
  - Given that the alarm has sounded, both burglary and earthquake become more likely
  - But if we then learn that a burglary has happened, the alarm is explained away and the probability of earthquake drops back
A variable’s Markov blanket consists of parents, children, children’s other parents

Every variable is conditionally independent of all other variables given its Markov blanket
CS 188: Artificial Intelligence

Bayes Nets: Exact Inference

Instructor: Sergey Levine and Stuart Russell--- University of California, Berkeley
Bayes Nets

Part I: Representation

Part II: Exact inference

- Enumeration (always exponential complexity)
- Variable elimination (worst-case exponential complexity, often better)
- Inference is NP-hard in general

Part III: Approximate Inference

Later: Learning Bayes nets from data
Inference

- Inference: calculating some useful quantity from a probability model (joint probability distribution)

- Examples:
  - Posterior marginal probability
    - $P(Q \mid e_1, \ldots, e_k)$
    - E.g., what disease might I have?
  - Most likely explanation:
    - $\text{argmax}_{q,r,s} P(Q=q,R=r,S=s \mid e_1, \ldots, e_k)$
    - E.g., what did he say?
Inference by Enumeration in Bayes Net

- Reminder of inference by enumeration:
  - Any probability of interest can be computed by summing entries from the joint distribution.
  - Entries from the joint distribution can be obtained from a BN by multiplying the corresponding conditional probabilities.

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

- So inference in Bayes nets means computing sums of products of numbers: sounds easy!!

- Problem: sums of **exponentially many** products!
Can we do better?

- Consider \(uwy + uwz + uxy + uxz + vwy + vwz + vxy + vxz\)
  - 16 multiplies, 7 adds
  - Lots of repeated subexpressions!
- Rewrite as \((u+v)(w+x)(y+z)\)
  - 2 multiplies, 3 adds
- \(\sum_{e,a} P(B) P(e) P(a | B, e) P(j | a) P(m | a)\)
  - \(= P(B)P(e)P(a | B, e)P(j | a)P(m | a) + P(B)P(\neg e)P(a | B, \neg e)P(j | a)P(m | a)\)
  - \(+ P(B)P(e)P(\neg a | B, e)P(j | \neg a)P(m | \neg a) + P(B)P(\neg e)P(\neg a | B, \neg e)P(j | \neg a)P(m | \neg a)\)

Lots of repeated subexpressions!
Variable elimination: The basic ideas

- Move summations inwards as far as possible
  - \[ P(B \mid j, m) = \alpha \sum_{e,a} P(B) \ P(e) \ P(a \mid B,e) \ P(j \mid a) \ P(m \mid a) \]
  - \[ = \alpha \ P(B) \ \sum_{e} P(e) \ \sum_{a} P(a \mid B,e) \ P(j \mid a) \ P(m \mid a) \]

- Do the calculation from the inside out
  - I.e., sum over \( a \) first, then sum over \( e \)
  - Problem: \( P(a \mid B,e) \) isn’t a single number, it’s a bunch of different numbers depending on the values of \( B \) and \( e \)
  - Solution: use arrays of numbers (of various dimensions) with appropriate operations on them; these are called factors
Factor Zoo
Factor Zoo I

- **Joint distribution**: $P(X,Y)$
  - Entries $P(x,y)$ for all $x, y$
  - $|X| \times |Y|$ matrix
  - Sums to 1

- **Projected joint**: $P(x,Y)$
  - A slice of the joint distribution
  - Entries $P(x,y)$ for one $x$, all $y$
  - $|Y|$-element vector
  - Sums to $P(x)$

<table>
<thead>
<tr>
<th>A \ J</th>
<th>true</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>0.09</td>
<td>0.01</td>
</tr>
<tr>
<td>false</td>
<td>0.045</td>
<td>0.855</td>
</tr>
</tbody>
</table>

$P(A,J)$

Number of variables (capitals) = dimensionality of the table
Factor Zoo II

- Single conditional: \( P(Y | x) \)
  - Entries \( P(y | x) \) for fixed \( x \), all \( y \)
  - Sums to 1

- Family of conditionals: \( P(X | Y) \)
  - Multiple conditionals
  - Entries \( P(x | y) \) for all \( x, y \)
  - Sums to \( |Y| \)

<table>
<thead>
<tr>
<th>( A ) ( \setminus ) ( J )</th>
<th>true</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>0.9</td>
<td>0.1</td>
</tr>
</tbody>
</table>

\[ P(J|a) - P(J|\neg a) \]

<table>
<thead>
<tr>
<th>( A ) ( \setminus ) ( J )</th>
<th>true</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>0.9</td>
<td>0.1</td>
</tr>
<tr>
<td>false</td>
<td>0.05</td>
<td>0.95</td>
</tr>
</tbody>
</table>
Operation 1: Pointwise product

- First basic operation: **pointwise product** of factors (similar to a **database join**, **not** matrix multiply!)
  - New factor has **union** of variables of the two original factors
  - Each entry is the product of the corresponding entries from the original factors

- Example: \( P(J|A) \times P(A) = P(A,J) \)

|          | \( P(A) \) |          | \( P(J|A) \) |          | \( P(A,J) \) |
|----------|------------|----------|-------------|----------|--------------|
|          | true | false   |          | true | false   |          | true | false   |          | true | false   |          | false | true | false   |          |
| A \( \setminus \) J | 0.1  | 0.9     |          | 0.9  | 0.1     |          | 0.09 | 0.01    |          | 0.045 | 0.855  |
Example: Making larger factors

Example: \( P(A,J) \times P(A,M) = P(A,J,M) \)

<table>
<thead>
<tr>
<th>A \ J</th>
<th>true</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>0.09</td>
<td>0.01</td>
</tr>
<tr>
<td>false</td>
<td>0.045</td>
<td>0.855</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A \ M</th>
<th>true</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>0.07</td>
<td>0.03</td>
</tr>
<tr>
<td>false</td>
<td>0.009</td>
<td>0.891</td>
</tr>
</tbody>
</table>

\[
P(A,J,M)
\]

<table>
<thead>
<tr>
<th>J \ M</th>
<th>true</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td></td>
<td></td>
</tr>
<tr>
<td>false</td>
<td>.0003</td>
<td></td>
</tr>
</tbody>
</table>

A=false

A=true
Example: Making larger factors

- Example: \( P(U, V) \times P(V, W) \times P(W, X) = P(U, V, W, X) \)
- Sizes: [10,10] x [10,10] x [10,10] = [10,10,10,10]
- I.e., 300 numbers blows up to 10,000 numbers!
- Factor blowup can make VE very expensive
Operation 2: Summing out a variable

- Second basic operation: **summing out** (or eliminating) a variable from a factor
  - Shrinks a factor to a smaller one
  - Example: $\sum_j P(A,J) = P(A,j) + P(A,\neg j) = P(A)$

<table>
<thead>
<tr>
<th>A \ J</th>
<th>true</th>
<th>false</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>0.09</td>
<td>0.01</td>
</tr>
<tr>
<td>false</td>
<td>0.045</td>
<td>0.855</td>
</tr>
</tbody>
</table>

$P(A,J)$

Sum out $J$

<table>
<thead>
<tr>
<th>P(A)</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
</tr>
<tr>
<td>false</td>
</tr>
</tbody>
</table>
Summing out from a product of factors

- Project the factors each way first, then sum the products
- Example: \( \sum_a P(a \mid B,e) \times P(j \mid a) \times P(m \mid a) \)
  
  \[ = P(a \mid B,e) \times P(j \mid a) \times P(m \mid a) + \]
  
  \[ P(\neg a \mid B,e) \times P(j \mid \neg a) \times P(m \mid \neg a) \]
Variable Elimination
Variable Elimination

- Query: $P(Q|E_1=e_1,.., E_k=e_k)$

- Start with initial factors:
  - Local CPTs (but instantiated by evidence)

- While there are still hidden variables (not Q or evidence):
  - Pick a hidden variable H
  - Join all factors mentioning H
  - Eliminate (sum out) H

- Join all remaining factors and normalize
function VariableElimination\( (Q, e, bn) \) returns a distribution over \( Q \)

\[ 
\text{factors} \leftarrow [ ]
\]

for each \( \text{var} \) in ORDER\( (bn.\text{vars}) \) do

\[ 
\text{factors} \leftarrow [\text{MAKE-FACTOR}(\text{var}, e) | \text{factors}]
\]

if \( \text{var} \) is a hidden variable then

\[ 
\text{factors} \leftarrow \text{SUM-OUT}(\text{var}, \text{factors})
\]

return NORMALIZE(POINTWISE-PRODUCT(\text{factors}))
Example

Query $P(B \mid j,m)$

Choose A

$P(A \mid B,E) \times \sum P(j \mid A) P(m \mid A) = P(j,m \mid B,E)$

$P(B) P(E) P(j,m \mid B,E)$
Example

Choose E

\[
P(E) \times P(j, m|B, E)
\]

\[
\sum P(j, m|B)
\]

Finish with B

\[
P(B) \times P(j, m|B)
\]

\[
P(j, m, B) \rightarrow \text{Normalize} \rightarrow P(B | j, m)
\]
Order matters

- Order the terms Z, A, B C, D
  - \[ P(D) = \alpha \sum_{z,a,b,c} P(z) P(a|z) P(b|z) P(c|z) P(D|z) \]
  - \[ = \alpha \sum_z P(z) \sum_a P(a|z) \sum_b P(b|z) \sum_c P(c|z) P(D|z) \]
  - Largest factor has 2 variables (D,Z)
- Order the terms A, B C, D, Z
  - \[ P(D) = \alpha \sum_{a,b,c,z} P(a|z) P(b|z) P(c|z) P(D|z) P(z) \]
  - \[ = \alpha \sum_a \sum_b \sum_c \sum_z P(a|z) P(b|z) P(c|z) P(D|z) P(z) \]
  - Largest factor has 4 variables (A,B,C,D)
- In general, with \( n \) leaves, factor of size \( 2^n \)
The computational and space complexity of variable elimination is determined by the largest factor (and it’s space that kills you).

The elimination ordering can greatly affect the size of the largest factor.
- E.g., previous slide’s example $2^n$ vs. 2

Does there always exist an ordering that only results in small factors?
- No!
Worst Case Complexity? Reduction from SAT

- CNF clauses:
  1. \( A \lor B \lor C \)
  2. \( C \lor D \lor \neg A \)
  3. \( B \lor C \lor \neg D \)

- \( P(\text{AND}) > 0 \) iff clauses are satisfiable
  - \( \Rightarrow \) NP-hard

- \( P(\text{AND}) = S \times 0.5^n \) where \( S \) is the number of satisfying assignments for clauses
  - \( \Rightarrow \) \#P-hard
A polytree is a directed graph with no undirected cycles.

For poly-trees the complexity of variable elimination is \textit{linear in the network size} if you eliminate from the leave towards the roots.

- This is essentially the same theorem as for tree-structured CSPs.
Bayes Nets

- Part I: Representation

- Part II: Exact inference
  - Enumeration (always exponential complexity)
  - Variable elimination (worst-case exponential complexity, often better)
  - Inference is NP-hard in general

- Part III: Approximate Inference

- Later: Learning Bayes nets from data