Bayes’ Net Semantics

- A set of nodes, one per variable X
- A directed, acyclic graph
- A conditional distribution for each node
  - A collection of distributions over X, one for each combination of parents’ values
    \[ P(X|a_1 \ldots a_n) \]
  - CPT: conditional probability table
  - Description of a noisy “causal” process

A Bayes net = Topology (graph) + Local Conditional Probabilities
Probabilities in BNs

- For all joint distributions, we have (chain rule):
  \[ P(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} P(x_i|x_1, \ldots, x_{i-1}) \]

- Bayes’ nets implicitly encode joint distributions
  - As a product of local conditional distributions
  - To see what probability a BN gives to a full assignment, multiply all the relevant conditionals together:
  \[ P(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} P(x_i|\text{parents}(X_i)) \]
  
- This lets us reconstruct any entry of the full joint

Not every BN can represent every joint distribution
- The topology enforces certain conditional independencies

All Conditional Independences

- Given a Bayes net structure, can run d-separation to build a complete list of conditional independences that are necessarily true of the form
  \[ X_i \perp\!\!\!\perp X_j | \{X_{k_1}, \ldots, X_{k_n}\} \]

- This list determines the set of probability distributions that can be represented
Same Assumptions, Different Graphs?

- Can you have two different graphs that encode the same assumptions?
  - Yes!
  - Examples:

Example: Independence

- For this graph, you can fiddle with $\theta$ (the CPTs) all you want, but you won’t be able to represent any distribution in which the flips are dependent!

<table>
<thead>
<tr>
<th>$P(X_1)$</th>
<th>$P(X_2)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>h 0.5</td>
<td>h 0.5</td>
</tr>
<tr>
<td>t 0.5</td>
<td>t 0.5</td>
</tr>
</tbody>
</table>
Topology Limits Distributions

- Given some graph topology $G$, only certain joint distributions can be encoded.
- The graph structure guarantees certain (conditional) independences.
- (There might be more independence)
- Adding arcs increases the set of distributions, but has several costs
- Full conditioning can encode any distribution.

Causality?

- When Bayes’ nets reflect the true causal patterns:
  - Often simpler (nodes have fewer parents)
  - Often easier to think about
  - Often easier to elicit from experts
- BNs need not actually be causal
  - Sometimes no causal net exists over the domain
  - E.g. consider the variables Traffic and Drips
  - End up with arrows that reflect correlation, not causation
- What do the arrows really mean?
  - Topology may happen to encode causal structure
  - Topology only guaranteed to encode conditional independence
- *More about causality: [Causality – Judea Pearl]*
Bayes Nets Representation Summary

- Bayes nets compactly encode joint distributions
- Guaranteed independencies of distributions can be deduced from BN graph structure
- D-separation gives precise conditional independence guarantees from graph alone
- A Bayes’ net’s joint distribution may have further (conditional) independence that is not detectable until you inspect its specific distribution

Inference

- Inference: calculating some useful quantity from a joint probability distribution
- Examples:
  - Posterior probability:
    \[ P(Q|E_1 = e_1, \ldots E_k = e_k) \]
  - Most likely explanation:
    \[ \text{argmax}_q P(Q = q|E_1 = e_1 \ldots) \]
Inference by Enumeration

- Given unlimited time, inference in BNs is easy
- Recipe:
  - State the marginal probabilities you need
  - Figure out ALL the atomic probabilities you need
  - Calculate and combine them
- Example:

\[
P(+b|+j,+m) = \frac{P(+b,+j,+m)}{P(+j,+m)}
\]

Example: Enumeration

- In this simple method, we only need the BN to synthesize the joint entries

\[
P(+b,+j,+m) = \\ P(+b)P(+e)P(+a|+b,+e)P(+j|+a)P(+m|+a) + \\ P(+b)P(+e)P(-a|+b,+e)P(+j|a)P(+m|a) + \\ P(+b)P(-e)P(+a|+b,-e)P(+j|+a)P(+m|+a) + \\ P(+b)P(-e)P(-a|+b,-e)P(+j|+a)P(+m|a)
\]
Why is inference by enumeration so slow?
- You join up the whole joint distribution before you sum out the hidden variables
- You end up repeating a lot of work!

Idea: interleave joining and marginalizing!
- Called “Variable Elimination”
- Still NP-hard, but usually much faster than inference by enumeration

We'll need some new notation to define VE
Factor Zoo I

- Joint distribution: $P(X,Y)$
  - Entries $P(x,y)$ for all $x, y$
  - Sums to 1

<table>
<thead>
<tr>
<th></th>
<th>T</th>
<th>W</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>hot</td>
<td>sun</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>hot</td>
<td>rain</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>cold</td>
<td>sun</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>cold</td>
<td>rain</td>
<td>0.3</td>
<td></td>
</tr>
</tbody>
</table>

- Selected joint: $P(x,Y)$
  - A slice of the joint distribution
  - Entries $P(x,y)$ for fixed $x$, all $y$
  - Sums to $P(x)$

<table>
<thead>
<tr>
<th></th>
<th>T</th>
<th>W</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>cold</td>
<td>sun</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>cold</td>
<td>rain</td>
<td>0.3</td>
<td></td>
</tr>
</tbody>
</table>

Factor Zoo II

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

<table>
<thead>
<tr>
<th></th>
<th>T</th>
<th>W</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>hot</td>
<td>sun</td>
<td>0.8</td>
<td></td>
</tr>
<tr>
<td>hot</td>
<td>rain</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>cold</td>
<td>sun</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>cold</td>
<td>rain</td>
<td>0.6</td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th></th>
<th>T</th>
<th>W</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>cold</td>
<td>sun</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>cold</td>
<td>rain</td>
<td>0.6</td>
<td></td>
</tr>
</tbody>
</table>
Factor Zoo III

- Specified family: $P(y | X)$
  - Entries $P(y | x)$ for fixed $y$, but for all $x$
  - Sums to … who knows!

- In general, when we write $P(Y_1 \ldots Y_N | X_1 \ldots X_M)$
  - It is a “factor,” a multi-dimensional array
  - Its values are all $P(y_1 \ldots y_N | x_1 \ldots x_M)$
  - Any assigned $X$ or $Y$ is a dimension missing (selected) from the array

| $P(rain|T)$ |
|---|---|---|
| T | W | P |
| hot | rain | 0.2 |
| cold | rain | 0.6 |

Example: Traffic Domain

- Random Variables
  - R: Raining
  - T: Traffic
  - L: Late for class!

- First query: $P(L)$

<table>
<thead>
<tr>
<th>$P(R)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$+r$</td>
</tr>
<tr>
<td>$-r$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$P(T</th>
<th>R)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$+r$</td>
<td>$+t$</td>
</tr>
<tr>
<td>$+r$</td>
<td>$-t$</td>
</tr>
<tr>
<td>$-r$</td>
<td>$+t$</td>
</tr>
<tr>
<td>$-r$</td>
<td>$-t$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$P(L</th>
<th>R)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$+t$</td>
<td>$+l$</td>
</tr>
<tr>
<td>$+t$</td>
<td>$-l$</td>
</tr>
<tr>
<td>$-t$</td>
<td>$+l$</td>
</tr>
<tr>
<td>$-t$</td>
<td>$-l$</td>
</tr>
</tbody>
</table>
Variable Elimination Outline

- Track objects called factors
- Initial factors are local CPTs (one per node)

\[
\begin{array}{ccc}
P(R) & P(T|R) & P(L|T) \\
+\text{r} & 0.1 & +\text{t} & 0.8 & +\text{l} & 0.3 \\
-\text{r} & 0.9 & -\text{t} & 0.2 & -\text{l} & 0.7 \\
-\text{r} & 0.1 & -\text{t} & 0.9 & -\text{l} & 0.1 \\
+\text{t} & 0.9 & +\text{l} & 0.8 & +\text{r} & 0.3 \\
-\text{t} & 0.2 & -\text{l} & 0.7 & -\text{r} & 0.1 \\
-\text{t} & 0.9 & -\text{l} & 0.1 & -\text{r} & 0.3 \\
\end{array}
\]

- Any known values are selected
  - E.g. if we know \( L = +\ell \), the initial factors are

\[
\begin{array}{ccc}
P(R) & P(T|R) & P(+\ell|T) \\
+\text{r} & 0.1 & +\text{t} & 0.8 & +\text{l} & 0.3 \\
-\text{r} & 0.9 & -\text{t} & 0.2 & -\text{l} & 0.7 \\
-\text{r} & 0.1 & -\text{t} & 0.9 & -\text{l} & 0.1 \\
+\text{t} & 0.9 & +\text{l} & 0.8 & +\text{r} & 0.3 \\
-\text{t} & 0.2 & -\text{l} & 0.7 & -\text{r} & 0.1 \\
-\text{t} & 0.9 & -\text{l} & 0.1 & -\text{r} & 0.3 \\
\end{array}
\]

- VE: Alternately join factors and eliminate variables

Operation 1: Join Factors

- First basic operation: joining factors
- Combining factors:
  - Just like a database join
  - Get all factors over the joining variable
  - Build a new factor over the union of the variables involved

Example: Join on \( R \)

\[
\begin{array}{ccc}
P(R) & P(T|R) & P(R,T) \\
+\text{r} & 0.1 & +\text{t} & 0.8 & +\text{l} & 0.08 \\
-\text{r} & 0.9 & -\text{t} & 0.2 & -\text{l} & 0.02 \\
-\text{r} & 0.1 & -\text{t} & 0.9 & -\text{l} & 0.09 \\
+\text{t} & 0.9 & +\text{l} & 0.8 & +\text{r} & 0.3 \\
-\text{t} & 0.2 & -\text{l} & 0.7 & -\text{r} & 0.1 \\
-\text{t} & 0.9 & -\text{l} & 0.1 & -\text{r} & 0.3 \\
\end{array}
\]

- Computation for each entry: pointwise products
  \( \forall r, t : P(r, t) = P(r) \cdot P(t|r) \)
Example: Multiple Joins

\[ P(R) \]
\[
\begin{array}{c|c}
\text{R} & 0.1 \\
-\text{R} & 0.9 \\
\end{array}
\]

\[ P(T|R) \]
\[
\begin{array}{c|c}
\text{+R} & 0.8 \\
-\text{R} & 0.2 \\
\end{array}
\]

Join R

\[ P(R, T) \]
\[
\begin{array}{c|c|c}
\text{R} & \text{T} & 0.08 \\
\text{-R} & -\text{T} & 0.02 \\
\end{array}
\]

\[ P(L|T) \]
\[
\begin{array}{c|c|c}
\text{+R} & +\text{L} & 0.3 \\
-\text{R} & -\text{L} & 0.7 \\
\end{array}
\]

Example: Multiple Joins

\[ P(R, T) \]
\[
\begin{array}{c|c|c}
\text{R} & \text{T} & 0.08 \\
\text{-R} & -\text{T} & 0.02 \\
\end{array}
\]

Join T

\[ P(L|T) \]
\[
\begin{array}{c|c|c}
\text{+R} & +\text{L} & 0.3 \\
-\text{R} & -\text{L} & 0.7 \\
\end{array}
\]
Operation 2: Eliminate

- Second basic operation: marginalization
- Take a factor and sum out a variable
  - Shrinks a factor to a smaller one
  - A projection operation

Example:

\[
P(R, T) \\
\begin{array}{c|c|c}
+r & +t & 0.08 \\
+r & -t & 0.02 \\
-r & +t & 0.09 \\
-r & -t & 0.81 \\
\end{array}
\]

\[
\text{sum } R \quad P(T) \\
\begin{array}{c|c}
+t & 0.17 \\
-t & 0.83 \\
\end{array}
\]

Multiple Elimination

\[
P(R, T, L) \\
\begin{array}{c|c|c|c}
+r & +t & +l & 0.024 \\
+r & +t & -l & 0.056 \\
+r & -t & +l & 0.002 \\
+r & -t & -l & 0.018 \\
-r & +t & +l & 0.027 \\
-r & +t & -l & 0.063 \\
-r & -t & +l & 0.081 \\
-r & -t & -l & 0.729 \\
\end{array}
\]

Sum out R

\[
P(T, L) \\
\begin{array}{c|c|c}
+t & +l & 0.051 \\
+t & -l & 0.119 \\
-t & +l & 0.083 \\
-t & -l & 0.747 \\
\end{array}
\]

Sum out T

\[
P(L) \\
\begin{array}{c}
+l & 0.134 \\
-l & 0.886 \\
\end{array}
\]
P(L) : Marginalizing Early!

\[ P(R) \]
-+r 0.1  
-r 0.9

\[ P(T|R) \]

\[ \begin{array}{cc} +r & +t 0.8 \\ +r & -t 0.2 \\ -r & +t 0.1 \\ -r & -t 0.9 \end{array} \]

Join R

\[ P(R, T) \]

\[ \begin{array}{cc} +r & +t 0.08 \\ +r & -t 0.02 \\ -r & +t 0.09 \\ -r & -t 0.81 \end{array} \]

Sum out R

\[ P(T) \]

\[ \begin{array}{c} +t 0.17 \\ -t 0.83 \end{array} \]

Marginalizing Early (aka VE*)

\[ P(T) \]

\[ \begin{array}{c} +t 0.17 \\ -t 0.83 \end{array} \]

Join T

\[ P(T, L) \]

\[ \begin{array}{cc} +t & +l 0.051 \\ +t & -l 0.119 \\ -t & +l 0.083 \\ -t & -l 0.747 \end{array} \]

Sum out T

\[ P(L) \]

\[ \begin{array}{c} +l 0.134 \\ -l 0.886 \end{array} \]

* VE is variable elimination
Evidence

- If evidence, start with factors that select that evidence
  - No evidence uses these initial factors:

|   | $P(R)$ | $P(T|R)$ | $P(L|T)$ |
|---|---|---|---|
| $+$ | 0.1 | 0.8 | $+$ | 0.3 |
| $-$ | 0.9 | 0.2 | $+$ | 0.7 |
| $+$ | 0.1 | $-$ | $-$ | 0.1 |
| $+$ | 0.9 | $-$ | $-$ | 0.9 |

- Computing $P(L|+r)$, the initial factors become:

|   | $P(+r)$ | $P(T|r)$ | $P(L|T)$ |
|---|---|---|---|
| $+$ | 0.1 | 0.8 | $+$ | 0.3 |
| $+$ | 0.2 | $+$ | 0.7 |
| $+$ | 0.1 | $-$ | 0.1 |
| $+$ | 0.9 | $-$ | 0.9 |

- We eliminate all vars other than query + evidence

---

Evidence II

- Result will be a selected joint of query and evidence
  - E.g. for $P(L|+r)$, we’d end up with:

|   | $P(+r,L)$ | Normalize | $P(L|+r)$ |
|---|---|---|---|
| $+$ | $+$ | 0.026 | $+$ | 0.26 |
| $+$ | $-$ | 0.074 | $-$ | 0.74 |

- To get our answer, just normalize this!

- That’s it!
General Variable Elimination

- Query: \( P(Q|E_1 = e_1, \ldots E_k = e_k) \)

- 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

Variable Elimination Bayes Rule

\[
P(B) \rightarrow P(a|B)
\]

Starting Table:

<table>
<thead>
<tr>
<th>B</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>+b</td>
<td>0.1</td>
</tr>
<tr>
<td>−b</td>
<td>0.9</td>
</tr>
</tbody>
</table>

Starting Table:

<table>
<thead>
<tr>
<th>B</th>
<th>A</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>+b</td>
<td>+a</td>
<td>0.8</td>
</tr>
<tr>
<td>+b</td>
<td>a</td>
<td>0.2</td>
</tr>
<tr>
<td>−b</td>
<td>+a</td>
<td>0.1</td>
</tr>
<tr>
<td>−b</td>
<td>a</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Joint Table:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>+a</td>
<td>+b</td>
<td>0.08</td>
</tr>
<tr>
<td>+a</td>
<td>−b</td>
<td>0.09</td>
</tr>
</tbody>
</table>

Normalized Table:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>+a</td>
<td>+b</td>
<td>8/17</td>
</tr>
<tr>
<td>+a</td>
<td>−b</td>
<td>9/17</td>
</tr>
</tbody>
</table>
Example

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

| \( P(B) \) | \( P(E) \) | \( P(A|B, E) \) | \( P(j|A) \) | \( P(m|A) \) |

Choose A

\[ \begin{align*}
  &P(A|B, E) \\
  &P(j|A) \\
  &P(m|A)
\end{align*} \]

\[ \times \quad \sum \]

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

Example

| \( P(B) \) | \( P(E) \) | \( P(j, m|B, E) \) |

Choose E

\[ \begin{align*}
  &P(E) \\
  &P(j, m|B, E)
\end{align*} \]

\[ \times \quad \sum \]

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

Finish with B

\[ \begin{align*}
  &P(B) \\
  &P(j, m|B)
\end{align*} \]

\[ \times \quad \text{Normalize} \quad \sum \]

\[ P(B|j, m) \]
Variable Elimination

What you need to know:
- Should be able to run it on small examples, understand the factor creation/reduction flow
- Better than enumeration: saves time by marginalizing variables as soon as possible rather than at the end

We will see special cases of VE later
- On tree-structured graphs, variable elimination runs in polynomial time, like tree-structured CSPs
- You’ll have to implement a tree-structured special case to track invisible ghosts (Project 4)