Graphical Models

Lecture 10:
Variable Elimination, continued

Andrew McCallum
mccallum@cs.umass.edu

Thanks to Noah Smith and Carlos Guestrin for some slide materials.
Last Time

• Probabilistic inference is the goal: $P(X \mid E = e)$.
  – #P-complete in general

• Do it anyway! Variable elimination ...
Markov Chain Example

\[ P(B) = \sum_{a \in \text{Val}(A)} P(A = a) P(B \mid A = a) \]

\[ P(C) = \sum_{b \in \text{Val}(B)} P(B = b) P(C \mid B = b) \]

\[ P(D) = \sum_{c \in \text{Val}(C)} P(C = c) P(D \mid C = c) \]
Last Time

• Probabilistic inference is the goal: \( P(X \mid E = e) \).
  – #P-complete in general

• Do it anyway! Variable elimination ...
  – Work on factors (algebra of factors)
  – Generally: “sum-product” inference \( \sum_{Z} \prod_{\phi \in \Phi} \phi \)
**Products of Factors**

- Given two factors with different scopes, we can calculate a new factor equal to their products.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>$\phi_1(A, B)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>30</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B</th>
<th>C</th>
<th>$\phi_2(B, C)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>100</td>
</tr>
</tbody>
</table>

$\phi_1(A, B) \cdot \phi_2(B, C) = \phi_3(A, B, C)$

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>$\phi_3(A, B, C)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3000</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>30</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>500</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1000</td>
</tr>
</tbody>
</table>
Factor Marginalization

• Given \( \mathbf{X} \) and \( \mathbf{Y} \) (\( \mathbf{Y} \not\in \mathbf{X} \)), we can turn a factor \( \phi(\mathbf{X}, \mathbf{Y}) \) into a factor \( \psi(\mathbf{X}) \) via marginalization:

\[
\psi(\mathbf{X}) = \sum_{y \in \text{Val}(Y)} \phi(\mathbf{X}, y)
\]

| P(C | A, B) | 0, 0 | 0, 1 | 1, 0 | 1,1 |
|---------|------|------|------|-----|
| 0       | 0.5  | 0.4  | 0.2  | 0.1 |
| 1       | 0.5  | 0.6  | 0.8  | 0.9 |

“summing out” B

\[
\begin{array}{|c|c|c|}
\hline
A & C & \psi(A, C) \\
\hline
0 & 0 & 0.9 \\
0 & 1 & 0.3 \\
1 & 0 & 1.1 \\
1 & 1 & 1.7 \\
\hline
\end{array}
\]
Last Time

• Probabilistic inference is the goal: \( P(\mathbf{X} \mid \mathbf{E} = \mathbf{e}) \).
  – \#P-complete in general

• Do it anyway! Variable elimination …
  – Work on factors (algebra of factors)
  – How to eliminate one variable
    (marginalize a product of factors)
Eliminating One Variable

Input: Set of factors $\Phi$, variable $Z$ to eliminate
Output: new set of factors $\Psi$

1. Let $\Phi' = \{\phi \in \Phi \mid Z \in \text{Scope}(\phi)\}$
2. Let $\Psi = \{\phi \in \Phi \mid Z \not\in \text{Scope}(\phi)\}$
3. Let $\psi$ be $\Sigma_Z \prod_{\phi \in \Phi'} \phi$
4. Return $\Psi \cup \{\psi\}$
Last Time

• Probabilistic inference is the goal: $P(X | E = e)$.
  – #P-complete in general

• Do it anyway! Variable elimination ...
  – Work on factors (algebra of factors)
  – Generally: “sum-product” inference $\sum_{Z} \prod_{\phi \in \Phi} \phi$
  – How to eliminate one variable
    (marginalize a product of factors)
  – How to eliminate a bunch of variables
Variable Elimination

Input: Set of factors $\Phi$, ordered list of variables $Z$ to eliminate

Output: new factor $\psi$

1. For each $Z_i \in Z$ (in order):
   
   – Let $\Phi = \text{Eliminate-One}(\Phi, Z_i)$

2. Return $\prod_{\phi \in \Phi} \phi$
Today

- Variable elimination for inference (with evidence)
- Complexity analysis of VE
- Elimination orderings
Probabilistic Inference

• Assume we are given a graphical model.

• Want:

\[
P(X \mid E = e) = \frac{P(X, E = e)}{P(E = e)} \propto P(X, E = e)
\]

\[
= \sum_{y \in \text{Val}(Y)} P(X, E = e, Y = y)
\]
Adding Evidence

• Conditional distributions are Gibbs; can be represented as factor graphs!

• Everything is essentially the same, but we reduce the factors to match the evidence.
  – Previously normalized factors may not be normalized any longer, but this is not a problem.

• Prune anything not on an active trail to query variables.
Example

• Query: $P(\text{Flu} \mid \text{runny nose})$

• Let’s reduce to $R = \text{true (runny nose)}$. 

<table>
<thead>
<tr>
<th>$P(R \mid S)$</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example

• Query: P(Flu | runny nose)

• Let’s reduce to R = true (runny nose).

| P(R | S) | 0 | 1 |
|-------|---|---|
|       |   |   |

<table>
<thead>
<tr>
<th>S</th>
<th>R</th>
<th>φ_{SR} (S, R)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Example

• Query: $P(\text{Flu} \mid \text{runny nose})$

• Let’s reduce to $R = \text{true (runny nose)}$. 

\[
\begin{array}{c|c|c}
\text{S} & \text{R} & \phi_{SR}(S, R) \\
\hline
0 & 0 & \ \\
0 & 1 & \ \\
1 & 0 & \ \\
1 & 1 & \ \\
\end{array}
\]

\[
\begin{array}{c|c|c}
\text{S} & \text{R} & \phi'_S(S) \\
\hline
0 & 1 & \ \\
1 & 1 & \ \\
\end{array}
\]
Example

• Query: $P(\text{Flu} \mid \text{runny nose})$

• Let’s reduce to $R = \text{true (runny nose)}$. 

<table>
<thead>
<tr>
<th>$P(R \mid S)$</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S</th>
<th>R</th>
<th>$\phi_{SR} (S, R)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S</th>
<th>R</th>
<th>$\phi'_S (S)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Example

- Query: $P(\text{Flu} \mid \text{runny nose})$
- Let’s reduce to $R = \text{true (runny nose)}$. 

<table>
<thead>
<tr>
<th>S</th>
<th>R</th>
<th>$\phi'_s (S)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Example

• Query: 
  \( P(\text{Flu} \mid \text{runny nose}) \)

• Now run variable elimination all the way down to one factor (for F).

H can be pruned for the same reasons as before.
Example

• Query: $P(\text{Flu} \mid \text{runny nose})$

• Now run variable elimination all the way down to one factor (for F).
Example

• Query:
P(Flu | runny nose)

• Now run variable elimination all the way down to one factor (for F).
Example

• Query: $P(\text{Flu} \mid \text{runny nose})$

• Now run variable elimination all the way down to one factor (for $F$).

Take final product.
Example

• Query:
  \[ P(\text{Flu} \mid \text{runny nose}) \]

• Now run variable elimination all the way down to one factor.
Variable Elimination for Conditional Probabilities

Input: Graphical model, set of query variables $Q$, evidence $E = e$

Output: factor $\phi$ and scalar $\alpha$

1. $\Phi =$ factors in the model
2. Reduce factors in $\Phi$ by $E = e$
3. Choose variable ordering on $Z = X \setminus Q \setminus E$
4. $\phi = \text{Variable-Elimination}(\Phi, Z)$
5. $\alpha = \sum_{z \in \text{Val}(Z)} \phi(z)$
6. Return $\phi, \alpha$
Note

• For Bayesian networks, the final factor will be $P(Q, E = e)$ and the sum $\alpha = P(E = e)$.

• This equates to a Gibbs distribution with partition function $= \alpha$. 
Complexity of
Variable Elimination
Complexity of Variable Elimination

- $n =$ number of random variables
- $m =$ number of factors
- In step $i$, we multiply all factors relating to $X_i$, resulting in $\psi_i$, and sum out $X_i$, giving a new factor $\tau_i$.
  - $N_i =$ number of entries in $\psi_i$
  - $N_{\text{max}} = \max_i N_i$
- If we eliminate everything, $m$ initial factors plus $n$ new ones (the $\tau_i$).
Complexity of Variable Elimination

• $m + n$ factors
• Each is multiplied once, then removed.
Recall: Eliminating One Variable

Input: Set of factors \( \Phi \), variable \( Z \) to eliminate
Output: new set of factors \( \Psi \)

1. Let \( \Phi' = \{ \phi \in \Phi \mid Z \in \text{Scope}(\phi) \} \)
2. Let \( \Psi = \{ \phi \in \Phi \mid Z \not\in \text{Scope}(\phi) \} \)
3. Let \( \psi \) be \( \sum_z \prod_{\phi \in \Phi'} \phi \)
4. Return \( \Psi \cup \{ \psi \} \)
Complexity of Variable Elimination

• $m + n$ factors
• Each is multiplied once to produce some $\psi_i$, then removed to produce $\tau_i$.
  – $(m + n) N_i$ multiplications for $X_i$
  – $O(mN_{\text{max}})$
• Marginalization (summing) touches each entry in each $\psi_i$ once:
  – $N_i$ additions for $X_i$
  – $O(nN_{\text{max}})$
Complexity of Variable Elimination

- Overall: $O(mN_{\max})$
  - Bayesian network: $m = n$
  - Markov network: $m \geq n$
Complexity of Variable Elimination

- Overall: $O(mN_{\text{max}})$
- The size $N_{\text{max}}$ of the intermediate factors $\psi_i$ is what makes this blow up.
  - $v$ values per random variable
  - $k_i$ variables for factor $\psi_i$
  - $N_i = v^{k_i}$

- But really, how bad is it?
Analyzing VE via the Graph

• Assume a factor graph representation.

• One step of VE, on $X_i$:
  – Create a single factor $\psi$ that includes $X_i$ and its neighbors ($Y$ that share factors with $X_i$).
  – Marginalize $X_i$ out of $\psi$, giving new factor $\tau$.
  – If we go back to a Markov network, we have now introduced new edges!
Example

- Factor graph.
Example

• Markov network.

Flu
All.
S.I.
R.N.
H.
Example

• Eliminate S.
Example

- Eliminate S.

- $\psi = \phi_{FAS} \cdot \phi_{SR} \cdot \phi_{SH}$
Example

• Eliminate S.

• $\psi = \phi_{FAS} \cdot \phi_{SR} \cdot \phi_{SH}$
Example

- Eliminate $S$.

- $\psi = \phi_{FAS} \cdot \phi_{SR} \cdot \phi_{SH}$

- $\tau = \sum_S \psi$
Example

• Eliminate $S$.

• $\psi = \Phi_{FAS} \cdot \Phi_{SR} \cdot \Phi_{SH}$

• $\tau = \sum S \psi$
Example

- Eliminate $S$.
- $\psi = \phi_{FAS} \cdot \phi_{SR} \cdot \phi_{SH}$
- $\tau = \sum_S \psi$
- Back to Markov net?
Example

removed stuff

“fill edges”
Insight

• Each VE step is a transformation on the graph.
  – We’ve been drawing it on slides this way all along!
• We can put the full sequence of graphs together into a single structure.
Union of the Graphs ...
Union of the Graphs
Induced Graph

• Take the union over all of the undirected graphs from each step: induced graph.
  
  (1) The scope of every intermediate factor is a clique in this graph.
  
  (2) Every maximal clique in the graph is the scope of some intermediate factor.

• Important:
  different ordering → different induced graph ...
Proof (1)

• The scope of every intermediate factor is a clique in the induced graph.
  – Consider $\psi(X_1, \ldots, X_k)$, an intermediate factor.
  – In the corresponding Markov network, all of the $X_i$ are connected (they share a factor).
  – Hence they form a clique.
Proof (2)

• Every maximal clique in the induced graph is the scope of some intermediate factor.
  – Consider maximal clique \( Y = \{Y_1, ..., Y_k\} \)
  – Let \( Y_1 \) be the first one eliminated, with resulting product-of-factors \( \psi \).
  – All edges relating to \( Y_1 \) are introduced \textit{before} it is eliminated.
  – \( Y_1 \) and \( Y_i \) share an edge, so they share a factor that gets multiplied into \( \psi \); so \( \psi \) includes all of \( Y \).
  – Any other variable \( X \) can’t be in the scope of \( \psi \), because it would have to be linked to all of \( Y \), so that \( Y \) wouldn’t be a maximal clique.
Ordering: \{S, \ldots\}
Ordering: \( \{H, R, S, A, F\} \)
Ordering: \{H, R, S, A, F\}
Ordering: \{H, R, S, A, F\}
Ordering: \{H, R, S, A, F\}
Ordering: \{H, R, S, A, F\}
Ordering: \{H, R, S, A, F\}

induced graph = original graph
“Induced Width”

- Number of nodes in the largest clique of the induced graph, minus one.
  - Relative to an ordering!

```
Flu     All.
|       |
S.I.    |
|       |
R.N.    H.
```

```
Flu     All.
|       |
S.I.    |
|
R.N.    H.
```
“Induced Width”

• Number of nodes in the largest clique of the induced graph, minus one.
  – Relative to an ordering!

• “Tree width” = minimum width over all possible orderings.
  – Bound on the best performance we can hope for …
    VE runtime is *exponential* in treewidth!

Why minus one?
Tree
width
Example

![Diagram of treewidth example]
Treewidth Example
Treewidth Example

[Diagram of a graph with four nodes connected in a square pattern]
Treewidth Example
Treewidth Example
Treewidth Example
Finding Elimination Orderings

• NP-complete:
  “Is there an elimination ordering such that induced width \( \leq K \)?”

• Nonetheless, some convenient cases arise.
You’d like to be able to look at the *original* graph and easily say something about the difficulty of inference
Chordal Graphs

- Undirected graph whose minimal cycles are not longer than 3.
Chordal Graphs

• Induced graphs are always chordal!
Chordal Graphs

- Induced graphs are always chordal!

Lemma: cannot add any edges incident on $X_i$ after it is eliminated.

When we eliminate C, edges A-C and C-D must exist.

After elimination A-D will exist.
Chordal Graphs

- Induced graphs are always chordal!

Lemma: cannot add any edges incident on $X_i$ after it is eliminated.

induced graph? not chordal
Theorem

- Chordal graphs always admit an elimination ordering that doesn’t introduce any fill edges into the graph.
  - No fill edges: no blowup.
  - Inference becomes *linear* in size of the factors already present!
Clique Tree

Every maximal clique becomes a vertex.

Tree structure.

Lecture 5
Clique Tree

For each edge, intersection of r.v.s separates the rest in $\mathcal{H}$.

sep$_H$(A, D | B, C)

sep$_H$(B, E | C, D)

sep$_H$(C, F | D, E)
Clique Tree

• Does a clique tree exist?
  – Yes, if the undirected graph $\mathcal{H}$ is chordal!
Theorem

• Chordal graphs always admit an elimination ordering that doesn’t introduce any fill edges into the graph.

• Proof by induction on the number of nodes in the tree:
  – Take a leaf $C_i$ in the clique tree.
  – Eliminate a variable in $C_i$ (but not $C_i$’s neighbor).
    • No fill edges.
    • Still chordal.
Heuristics for Variable Elimination Ordering other than using a clique tree
Alternative Ordering Heuristic: Maximum Cardinality

• Start with undirected graph on $X$, all nodes unmarked.

• For $i = |X|$ to 1:
  – Let $Y$ be the unmarked variable in $X$ with the largest number of marked neighbors
  – $\pi(Y) = i$
  – Mark $Y$.

• Eliminate using permutation $\pi$.
  – i.e. $\pi$ maps each variable to an integer;
    eliminate the variables in order of those integers 1, 2, 3...
Alternative Ordering Heuristic: Maximum Cardinality

• “Maximum Cardinality” permutation will not introduce any fill edges for chordal graphs.
• Don’t need the clique tree.
• Can also use it on non-chordal graphs!
  – Better ideas exist, though; greedy algorithms that try to add a small number of edges at a time.
Bayesian Networks Again

• Recall: If undirected graph $\mathcal{H}$ is chordal, then there is a Bayesian network structure $G$ that is a P-map for $\mathcal{H}$.

• Chordal graphs correspond to Bayesian networks with a **polytree** structure.
  – At most one trail between any pair of nodes.

“polytree” = directed graph with at most one undirected path between any two vertices; equiv: directed acyclic graph (DAG) for which there are no undirected cycles either.
Example Polytree

From Wikipedia :-)
Inference in Polytrees

• Linear in conditional probability table sizes!
  – Consider the skeleton.
  – Pick a root; form a tree.
  – Work in from “leaves.”

• In the corresponding undirected graph, no fill edges are introduced.
Variable Elimination Summary

• In general, exponential requirements in induced width corresponding to the ordering you choose.
• It’s NP-hard to find the best elimination ordering.
• If you can avoid fill edges (or “big” intermediate factors), you can make inference linear in the size of the original factors.
  – Chordal graphs
  – Polytrees