The Elimination Algorithm

Probabilistic Graphical Models (10-708)

Lecture 4, Sep 26, 2007

Eric Xing

Reading: J-Chap 3, KF-Chap. 8, 9

Questions?
Probabilistic Inference

- We now have compact representations of probability distributions: \textbf{Graphical Models}

- A GM $G$ describes a unique probability distribution $P$

- How do we answer \textit{queries} about $P$?

- We use \textit{inference} as a name for the process of computing answers to such queries

Query 1: Likelihood

- Most of the queries one may ask involve \textit{evidence}
  - Evidence $e$ is an assignment of values to a set $E$ variables in the domain
  - Without loss of generality $E = \{X_{k+1}, \ldots, X_n\}$

- Simplest query: compute probability of evidence

$$P(e) = \sum_{x_1} \cdots \sum_{x_k} P(x_1, \ldots, x_k, e)$$

- this is often referred to as computing the \textit{likelihood} of $e$
Query 2: Conditional Probability

- Often we are interested in the **conditional probability distribution** of a variable given the evidence

\[ P(X | e) = \frac{P(X, e)}{P(e)} = \frac{P(X, e)}{\sum_x P(X = x, e)} \]

- this is the **a posteriori belief** in \( X \), given evidence \( e \)

- We usually query a subset \( Y \) of all domain variables \( X = \{Y, Z\} \) and "don't care" about the remaining, \( Z \):

\[ P(Y | e) = \sum_z P(Y, Z = z | e) \]

- the process of summing out the "don't care" variables \( Z \) is called **marginalization**, and the resulting \( P(Y | e) \) is called a **marginal** prob.

Applications of a posteriori Belief

- **Prediction**: what is the probability of an outcome given the starting condition

- the query node is a descendent of the evidence

- **Diagnosis**: what is the probability of disease/fault given symptoms

- the query node an ancestor of the evidence

- **Learning** under partial observation

  - fill in the unobserved values under an "EM" setting (more later)

- The directionality of information flow between variables is not restricted by the directionality of the edges in a GM

  - probabilistic inference can combine evidence form all parts of the network
Query 3: Most Probable Assignment

In this query we want to find the most probable joint assignment (MPA) for some variables of interest.

Such reasoning is usually performed under some given evidence $\mathbf{e}$, and ignoring (the values of) other variables $\mathbf{z}$:

$$
\text{MPA}(Y \mid \mathbf{e}) = \arg \max_{y \in Y} P(y \mid \mathbf{e}) = \arg \max_{y \in Y} \sum_{z} P(y, z \mid \mathbf{e})
$$

This is the maximum a posteriori configuration of $y$.

Applications of MPA

- Classification
  - find most likely label, given the evidence
- Explanation
  - what is the most likely scenario, given the evidence

Cautionary note:

- The MPA of a variable depends on its "context"---the set of variables been jointly queried

<table>
<thead>
<tr>
<th>$x$</th>
<th>$y$</th>
<th>$P(x,y)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0.35</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0.05</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0.3</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.3</td>
</tr>
</tbody>
</table>

Example:

- MPA of $X$?
- MPA of $(X, Y)$?
Complexity of Inference

Thm:
Computing $P(X = x \mid e)$ in a GM is NP-hard

- Hardness does not mean we cannot solve inference
  - It implies that we cannot find a general procedure that works efficiently for arbitrary GMs
  - For particular families of GMs, we can have provably efficient procedures

Approaches to inference

- Exact inference algorithms
  - The elimination algorithm
  - Message-passing algorithm (sum-product, belief propagation)
  - The junction tree algorithms

- Approximate inference techniques
  - Stochastic simulation / sampling methods
  - Markov chain Monte Carlo methods
  - Variational algorithms
Marginalization and Elimination

- A signal transduction pathway:

\[ A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \]

What is the likelihood that protein E is active?

- Query: \( P(e) \)

\[
P(e) = \sum_d \sum_c \sum_b \sum_a P(a,b,c,d,e)
\]

By chain decomposition, we get

\[
= \sum_d \sum_c \sum_b \sum_a P(a)P(b|a)P(c|b)P(d|c)P(e|d)
\]

A naïve summation needs to enumerate over an exponential number of terms.

Elimination on Chains

- Rearranging terms ...

\[
P(e) = \sum_d \sum_c \sum_b \sum_a P(a)P(b|a)P(c|b)P(d|c)P(e|d)
\]

\[
= \sum_d \sum_c \sum_b \sum_a P(c|b)P(d|c)P(e|d)\sum_a P(a)P(b|a)
\]
Now we can perform innermost summation

\[
P(e) = \sum_d \sum_c \sum_b P(c \mid b) P(d \mid c) P(e \mid d) \sum_a P(a) P(b \mid a)
\]

\[
= \sum_d \sum_c \sum_b P(c \mid b) P(d \mid c) P(e \mid d) p(b)
\]

This summation "eliminates" one variable from our summation argument at a "local cost".

Rearranging and then summing again, we get

\[
P(e) = \sum_d \sum_c \sum_b P(c \mid b) P(d \mid c) P(e \mid d) p(b)
\]

\[
= \sum_d \sum_c P(d \mid c) P(e \mid d) \sum_b P(c \mid b) p(b)
\]

\[
= \sum_d \sum_c P(d \mid c) P(e \mid d) p(c)
\]
Elimination in Chains

- Eliminate nodes one by one all the way to the end, we get

\[ P(e) = \sum_d P(e \mid d) p(d) \]

\[ \sum_{x_e} P(x_e \mid x_{\overline{e}}) p(x_{\overline{e}}) \]

- Complexity:
  - Each step costs \( O(|\text{Val}(X_i)| \times |\text{Val}(X_{i+1})|) \) operations: \( O(kn^2) \)
  - Compare to naïve evaluation that sums over joint values of \( n-1 \) variables \( O(n^4) \)

Undirected Chains

- Rearranging terms ...

\[ P = \frac{1}{Z} \sum \phi(\gamma, \phi_e) \]

\[ P(e) = \frac{1}{Z} \sum_d \sum_c \sum_b \sum_a \phi(b, a) \phi(c, b) \phi(d, c) \phi(e, d) \]

\[ = \frac{1}{Z} \sum_d \sum_c \sum_b \phi(c, b) \phi(d, c) \phi(e, d) \sum_a \phi(b, a) \]

\[ = \cdots \]
The Sum-Product Operation

- In general, we can view the task at hand as that of computing the value of an expression of the form:

\[ \sum_{\phi \in \mathcal{F}} \prod_{z} \phi \]

where \( \mathcal{F} \) is a set of factors

- We call this task the *sum-product* inference task.

Outcome of elimination

- Let \( X \) be some set of variables, let \( \mathcal{F} \) be a set of factors such that for each \( \phi \in \mathcal{F} \), \( \text{Scope}[\phi] \subseteq X \), let \( Y \subseteq X \) be a set of query variables, and let \( Z = X \setminus Y \) be the variable to be eliminated.

- The result of eliminating the variable \( Z \) is a factor

\[ \tau(Y) = \sum_{\phi \in \mathcal{F}} \prod_{z} \phi \]

\[ p(Y) = \frac{\tau(Y)}{\sum_{Y'}} \]

- This factor does not necessarily correspond to any probability or conditional probability in this network. (example forthcoming)
Dealing with evidence

- Conditioning as a Sum-Product Operation
  - The evidence potential: \( \delta(E_i, \bar{e}_i) = \begin{cases} 1 & \text{if } E_i = \bar{e}_i \\ 0 & \text{if } E_i \neq \bar{e}_i \end{cases} \)
  - Total evidence potential: \( \delta(E, \bar{e}) = \prod_{i \in E} \delta(E_i, \bar{e}_i) \)
  - Introducing evidence --- restricted factors:

\[
\tau(Y, \bar{e}) = \sum_{z, e} \prod_{\phi \in \mathcal{F}} \phi \times \delta(E, \bar{e})
\]

Inference on General GM via Variable Elimination

General idea:
- Write query in the form
  \[ P(X_1, e) = \sum_{x_n} \cdots \sum_{x_1} \prod_{i \in \mathcal{N}} P(x_i \mid p a_i) \]
  - this suggests an "elimination order" of latent variables to be marginalized
- Iteratively
  - Move all irrelevant terms outside of innermost sum
  - Perform innermost sum, getting a new term
  - Insert the new term into the product
- wrap-up
  \[ P(X_1 \mid e) = \frac{\phi(X_1, e)}{\sum_{a_1} \phi(X_1, e)} \]
The elimination algorithm

Procedure **Elimination** (G, E, Z, X)

1. Initialize (G)
2. Evidence (E)
3. Sum-Product-Elimination (F, Z, <)
4. Normalization (F)

Procedure **Initialize** (G, Z)

1. Let Z₁, . . . , Zᵦ be an ordering of Z such that Zᵢ < Zⱼ if i < j
2. Initialize F with the full set of factors

Procedure **Evidence** (E)

1. for each i ∈ E, 
   F = F ∪ δ(Eᵢ, eᵢ)

Procedure **Sum-Product-Variable-Elimination** (F, Z, <)

1. for i = 1, . . . , k
   F ← Sum-Product-Eliminate-Var(F, Zᵢ)
2. φ⁺ ← Πᵢ∈F φ
3. return φ⁺
4. Normalization (φ⁺)
The elimination algorithm

Procedure Initialize \((G, Z)\)
1. Let \(Z_1, \ldots, Z_k\) be an ordering of \(Z\) such that \(Z_i < Z_j\) iff \(i < j\)
2. Initialize \(\mathcal{F}\) with the full set of factors

Procedure Evidence \((E)\)
1. \(\mathcal{F} = \mathcal{F} \cup \delta(E, e)\)

Procedure Sum-Product-Variable-Elimination \((\mathcal{F}, Z, <)\)
1. for \(i = 1, \ldots, k\)
   \(\mathcal{F} \leftarrow \text{Sum-Product-Eliminate-Var}(\mathcal{F}, Z_i)\)
2. \(\phi^* \leftarrow \prod_{\phi \in \mathcal{F}} \phi\)
3. return \(\phi^*\)
4. Normalization \((\phi^*)\)

Procedure Normalization \((\phi^*)\)
1. \(P(\lambda|E) = \phi^*(\lambda) / \sum\phi^*(\lambda)\)

Procedure Sum-Product-Eliminate-Var \((\mathcal{F}, Z, \text{variable to be eliminated})\)
1. \(\mathcal{F}' \leftarrow \{\phi \in \mathcal{F} : Z \in \text{Scope}[\phi]\}\)
2. \(\mathcal{F}'' \leftarrow \mathcal{F} - \mathcal{F}'\)
3. \(\psi \leftarrow \prod_{\phi \in \mathcal{F}''} \phi\)
4. \(\tau \leftarrow \sum\psi\)
5. return \(\mathcal{F}'' \cup \{\}\)

A more complex network

A food web

What is the probability that hawks are leaving given that the grass condition is poor?
Example: Variable Elimination

- Query: \( P(A \mid h) \)
  - Need to eliminate: \( B,C,D,E,F,G,H \)

- Initial factors:
  \[
P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)P(g \mid e)P(h \mid e,f)
\]

- Choose an elimination order: \( H,G,F,E,D,C,B \)

- Step 1:
  - Conditioning (fix the evidence node (i.e., \( h \)) on its observed value (i.e., \( \hat{h} \))):
    \[
m_h(e, f) = p(h = \hat{h} \mid e, f)
    \]
  - This step is isomorphic to a marginalization step:
    \[
m_h(e, f) = \sum_h p(h \mid e, f) \delta(h = \hat{h})
    \]

Example: Variable Elimination

- Query: \( P(B \mid h) \)
  - Need to eliminate: \( B,C,D,E,F,G \)

- Initial factors:
  \[
P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)P(g \mid e)P(h \mid e,f)
\Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)P(g \mid e)m_g(e, f)
\]

- Step 2: Eliminate \( G \)
  - compute
    \[
m_g(e) = \sum_g p(g \mid e) = 1
    \]
  \[
\Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)m_g(e, f)
= P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)m_g(e, f)
\]
Example: Variable Elimination

- Query: $P(B \mid h)$
  - Need to eliminate: $B,C,D,E,F$

- Initial factors:
  $$P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)P(g \mid e)P(h \mid e,f)$$
  $$\Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)P(g \mid e)m_b(e,f)$$
  $$\Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)m_a(e,f)$$

- Step 3: Eliminate $F$
  - compute
    $$m_f(e,a) = \sum_f p(f \mid a)m_b(e,f)$$
    $$\Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)m_f(a,e)$$

Example: Variable Elimination

- Query: $P(B \mid h)$
  - Need to eliminate: $B,C,D,E$

- Initial factors:
  $$P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)P(g \mid e)P(h \mid e,f)$$
  $$\Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)P(g \mid e)m_b(e,f)$$
  $$\Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)P(f \mid a)m_a(e,f)$$
  $$\Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c,d)m_e(a,c)$$

- Step 4: Eliminate $E$
  - compute
    $$m_e(a,c,d) = \sum_c p(e \mid c,d)m_f(a,e)$$
    $$\Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)m_e(a,c,d)$$
Example: Variable Elimination

- Query: \( P(B \mid h) \)
  - Need to eliminate: \( B, C, D \)

- Initial factors:
  \[
P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c, d)P(f \mid a)P(g \mid e)P(h \mid e, f)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)P(e \mid c, d)P(f \mid a)P(g \mid e)m_i(e, f)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)eP(e \mid c, d)m_i(a, e)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid b)P(d \mid a)m_i(a, c, d)
  \]

- Step 5: Eliminate \( D \)
  - compute \( m_j(a, c) = \sum_d p(d \mid a)m_i(a, c, d) \)
  \[
  \Rightarrow P(a)P(b)P(c \mid d)m_j(a, c)
  \]

Example: Variable Elimination

- Query: \( P(B \mid h) \)
  - Need to eliminate: \( B, C \)

- Initial factors:
  \[
P(a)P(b)P(c \mid d)P(d \mid a)P(e \mid c, d)P(f \mid a)P(g \mid e)P(h \mid e, f)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid d)P(d \mid a)P(e \mid c, d)P(f \mid a)P(g \mid e)m_i(e, f)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid d)P(d \mid a)eP(e \mid c, d)m_i(a, e)
  \]
  \[
  \Rightarrow P(a)P(b)P(c \mid d)P(d \mid a)m_i(a, c, d)
  \]

- Step 6: Eliminate \( C \)
  - compute \( m_j(a, b) = \sum_c p(c \mid b)m_j(a, c) \)
  \[
  \Rightarrow P(a)P(b)P(c \mid d)m_j(a, c)
  \]
Example: Variable Elimination

- **Query:** \( P(B | h) \)
  - Need to eliminate: \( B \)

- **Initial factors:**
  
  \[
  P(a)P(b)P(c|d)P(d|a)P(e|c,d)P(f|a)P(g|e)P(h|e,f) \\
  \Rightarrow P(a)P(b)P(c|d)P(d|a)P(e|c,d)P(f|a)P(g|e)m_f(e,f) \\
  \Rightarrow P(a)P(b)P(c|d)P(d|a)P(e|c,d)P(f|a)m_f(e,f) \\
  \Rightarrow P(a)P(b)P(c|d)P(d|a)eP(e|c,d)m_f(a,e) \\
  \Rightarrow P(a)P(b)P(c|d)P(d|a)eP(e|c,d)m_f(a,e,d) \\
  \Rightarrow P(a)P(b)P(c|d)P(d|a)eP(e|c,d)m_f(a,e,c) \\
  \Rightarrow P(a)P(b)eP(e|c,d)m_f(a) \\
  \Rightarrow P(a)m_f(a)
  \]

- **Step 7: Eliminate** \( B \)
  - Compute
  
  \[
  m_a(a) = \sum_b p(b)m_f(a,b)
  \]

  \[
  \Rightarrow P(a)m_f(a)
  \]

Example: Variable Elimination

- **Query:** \( P(B | h) \)
  - Need to eliminate: \( B \)

- **Initial factors:**
  
  \[
  P(a)P(b)P(c|d)P(d|a)P(e|c,d)P(f|a)P(g|e)P(h|e,f) \\
  \Rightarrow P(a)P(b)P(c|d)P(d|a)P(e|c,d)P(f|a)P(g|e)m_f(e,f) \\
  \Rightarrow P(a)P(b)P(c|d)P(d|a)P(e|c,d)P(f|a)m_f(e,f) \\
  \Rightarrow P(a)P(b)P(c|d)P(d|a)eP(e|c,d)m_f(a,e) \\
  \Rightarrow P(a)P(b)P(c|d)P(d|a)eP(e|c,d)m_f(a,e,d) \\
  \Rightarrow P(a)P(b)P(c|d)P(d|a)eP(e|c,d)m_f(a,e,c) \\
  \Rightarrow P(a)P(b)eP(e|c,d)m_f(a) \\
  \Rightarrow P(a)m_f(a)
  \]

- **Step 8: Wrap-up**
  
  \[
  p(a,\tilde{h}) = p(a)m_f(a), \quad p(\tilde{h}) = \sum_a p(a)m_f(a) \\
  \Rightarrow P(a|\tilde{h}) = \frac{p(a)m_f(a)}{\sum_a p(a)m_f(a)}
  \]
Suppose in one elimination step we compute

\[ m_i(x_1, \ldots, x_k) = \sum_{x} m^i_{x_1}(x, x_1, \ldots, x_k) \]

\[ m^i_{x_1}(x, x_1, \ldots, x_k) = \prod_{i=1}^{k} m_i(x, y_{c_i}) \]

This requires

- \( k \cdot |\text{Val}(X)| \cdot \prod_i |\text{Val}(\mathcal{C}_i)| \) multiplications
  - For each value for \( x, y_{p_1}, \ldots, y_{k_1} \), we do \( k \) multiplications
- \(|\text{Val}(X)| \cdot \prod_i |\text{Val}(\mathcal{C}_i)| \) additions
  - For each value of \( y_{p_1}, \ldots, y_{k_1} \), we do \(|\text{Val}(X)| \) additions

Complexity is exponential in number of variables in the intermediate factor

Complexity of variable elimination