Message passing in clique trees

June 5, 2013

- an alternative approach to Variable-Elimination
- manipulation of factors is a basic computational step
- clique tree: a global data structure for scheduling these operations
- all operations can be performed on normalized as well as unnormalized measures
  - the unnormalized measure (unnormalized joint probability): $\tilde{P}_{\Phi}(\mathcal{X}) = \sum_{\phi_i \in \Phi} \phi_i(\mathbf{X_i})$, where $\mathcal{X}$ is a set of all variables, Φ set of all factors and $\mathbf{X_i}$ set of variables in a scope of the factor ϕ_i

1 Message passing in clique trees $\Leftrightarrow$ Variable-Elimination

In this section I would like to show that message passing in clique trees is an approach how to compute marginals of a joint probability distribution, equivalent to variable-elimination.

1.1 Variable-Elimination algorithm

An algorithm to compute marginals from a joint probability distribution.

Example:

We would like to compute $P(C)$ for a distribution represented by a Markov Net in Figure (TODO).

Instead of computing normalized measure $P(C)$ we can put off the normalization at the very end and work with unnormalized measures ($\tilde{P}$). The marginal of C is simply a joint distribution after marginalizing out all variables except for C. Joint distribution is defined as a product of all factors.

$$\tilde{P}(C) = \sum_{A,B,D,E} \tilde{P}(A,B,C,D,E) = \sum_{A,B,D,E} \phi(A,B)\phi(B,C)\phi(B,E)\phi(C,D)\phi(C,E) =$$

We eliminate all variables except for C one after another. We pick a variable A to start with. We can group all factors which A participates in (in this case just $\phi(A,B)$), multiply them all into a single factor $\psi_1(A,B) = \phi(A,B)$ and marginalize out A by summing over values of A to get a new factor $\tau_1(B) = \sum_A \psi_1(A,B)$.

$$= \sum_{B,D,E} \phi(B,C)\phi(B,E)\phi(C,D)\phi(C,E) \sum_A \phi(A,B) = \sum_{B,D,E} \phi(B,C)\phi(B,E)\phi(C,D)\phi(C,E)\tau_1(B) =$$

We proceed by eliminating B, which again consists of a regroupin, multiplying to a new factor $\psi_2(B,C,E) = \phi(B,C)\phi(B,E)\tau_1(B)$ and marginalizing out B to get a new factor $\tau_2(C,E) = \sum_B \psi_1(B,C,E)$.

$$= \sum_{D,E} \phi(C,D)\phi(C,E) \sum_B \phi(B,C)\phi(B,E)\tau_1(B) = \sum_{D,E} \phi(C,D)\phi(C,E)\tau_2(C,E)$$

We will continue in the same manner until we eliminate all variables except for C. The resulting function of C is then an unnormalized marginal distribution $\tilde{P}(C)$.

1.2 Clique tree

We have to define a basic structure message passing processes on - a clique tree.

Definition Clique tree is an undirected tree such that:

- nodes are clusters of variables $C_i \subseteq \mathcal{X} = \{X_1, \ldots, X_n\}$
- edge between C_i and C_j is associated with sepset $S_{i,j} = C_i \cap C_j$
- Family preservation property (FPP): for each factor$\phi_k \in \Phi$, it is assigned to single cluster $C_{\alpha(k)}$, so that $Scope[\phi_k] \subseteq C_{\alpha(k)}$[1]
- Running intersection property (RIP): for each pair of clusters C_i and C_j and variable $X \in C_i \cap C_j$there exists a unique path between C_i and C_j for which all clusters and sepsets contain X

1.3 Variable-Elimination⇒Message passing in clique trees

We want to show that V-E produces a clique tree. Let us construct a clique tree as follows. The multiplied factor ψ_i produced in a single step of V-E forms a cluster $C_i = Scope(\psi_i)$. We connect clusters C_i and C_j by an edge if the factor τ_i produced by eliminating ψ_i is used in making of ψ_j.

Lemma The graph produced by V-E is a tree.

Proof Each intermediate factor ψ_i (representing a node) is used only once in computing τ_i (an edge) and never appears again.

[1] $Scope[\phi(X)] = X$, so it is a set of variables associated with a factor

Lemma The tree satisifies an FPP.

> **Proof** Each original factor ϕ_k is used in construction of some ψ_i (a cluster $C_{\alpha(k)}$) and never reappears. All factors ϕ must be eliminated in V-E.

Lemma The tree satisfies an RIP.

> **Proof** Suppose, we have clusters C,$C^{‘}$ and C_X and a variable X, which is in the scope of each of the three clusters. Furthermore, C_X is a cluster where X is eliminated (or the cluster that remains at the end of V-E).
> C must take place earlier in the algorithm than C_X. X is eliminated in C_X, i.e. all factors containing X are multiplied into C_X and the result of summation does not contain X in its scope. Thus, X cannot appear after C_X is created.
> By assumption, $X \in Scope(C)$. X is not eliminated in C, so $X \in Scope(\tau_C)$. A neighbor in the tree just multiplies τ_C. This happens for all clusters upstream from C until X is eliminated in C_X. Thus, X appears in all clusters between C and C_X.
> The same holds for $C^{‘}$, so there exist a unique X - path between C and $C^{‘}$.

Corollary Variable-Elimination produces a clique tree.

1.4 Message passing in a clique tree

In the following we show a simplified version of MP algorithm, in which we calculate the unnormalized marginal $\tilde{P}(Scope(C_{root}))$ of all variables associated with a given cluster C (the root). Given this marginal, we can further compute a marginal of any variables fully contained in the $Scope(C)$ by means of V-E.

1. Select a root cluster C_{root} containing all variables the marginal that we search for consists of

2. Calculate initial potentials for all C_i:

$$\psi(C_i) = \prod_{\phi_j \in C_i} \phi_j$$

3. Pass message if possible, i.e. all incoming messages have been received (except for the one from the cluster we are passing to):

$$\delta_{i \to j} = \sum_{C_i \setminus S_{i,j}} \psi_i \prod_{k \in Nb_i \setminus \{j\}} \delta_{k \to i}$$

4. After the root has received all messages, calculate belief:

$$\beta_{root} = \psi_{root} \prod_{k \in Nb_{root}} \delta_{k \to root}$$

The calculated belief represents, as we show below:

$$\beta_{root} = \tilde{P}(Scope(C_{root})) = \sum_{\mathcal{X}\setminus Scope(C_{root})} \prod_{\phi} \phi$$

1.5 Message passing in clique trees ⇒Variable-Elimination

Our aim is to show the correctness of the algorithm, that MP over a clique tree that statisfies FPP and RIP produces the same marginals as V-E as it is schematized by the last equation. We start with an auxilliary lemma.

Lemma X is eliminated when a message from C_i to C_j is sent. Then X does not appear anywhere in the tree on the C_j side of the edge $(i \leftrightarrow j)$.

Proof A consequence of RIP. Assume by contradicton there is a node C_k containing variable X on the C_j side of the edge $(i \leftrightarrow j)$. So C_j lies on the path from C_k (containing X) to C_i (containing X). However, C_i does not contain X, which violates RIP.

Let us introduce a notation used in the last proof. $\mathcal{F}_{<(i\leftrightarrow j)}$ is a set of all factors asociated with clusters on the C_i side of the edge. $\mathcal{V}_{<(i\leftrightarrow j)}$ is a set of variables that appear on the C_i side but not in the edge $(i \leftrightarrow j)$ itself.

Theorem Let $\delta_{i\to j}$ be a message from C_i to C_j. Then

$$\delta_{i\to j} = \sum_{\mathcal{V}_{<(i\leftrightarrow j)}} \prod_{\phi\in\mathcal{F}_{<(i\leftrightarrow j)}} \phi$$

Proof By induction from leafs to the root.
If C_i is a leaf, the equation follows from the definition of a message.
If C_i is not a leaf and C_k, $k \in \{i_1,\ldots,i_m\} = Nb_i \setminus \{j\}$ are the neighbours of C_i (excluding C_j) then:

$$\begin{aligned}
\mathcal{F}_{<(i\leftrightarrow j)} &= \mathcal{F}_{<(i_1\leftrightarrow i)} \cup \ldots \cup \mathcal{F}_{<(i_m\leftrightarrow i)} \cup \mathcal{F}_i;\ \mathcal{F}_{<(i_1\leftrightarrow i)} \cap \ldots \cap \mathcal{F}_{<(i_m\leftrightarrow i)} \cap \mathcal{F}_i = \emptyset \\
\mathcal{V}_{<(i\leftrightarrow j)} &= \mathcal{V}_{<(i_1\leftrightarrow i)} \cup \ldots \cup \mathcal{V}_{<(i_m\leftrightarrow i)} \cup Y_i;\ \mathcal{V}_{<(i_1\leftrightarrow i)} \cap \ldots \cap \mathcal{V}_{<(i_m\leftrightarrow i)} \cap Y_i = \emptyset
\end{aligned}$$

where Y_i is a set of variables eliminated in C_i itself. Due to this property we can partition sums and products in the equation as follows:

$$\sum_{\mathcal{V}_{<(i\leftrightarrow j)}} \prod_{\phi\in\mathcal{F}_{<(i\leftrightarrow j)}} \phi = \sum_{Y_i}(\sum_{\mathcal{V}_{<(i_m\leftrightarrow i)}} \cdots (\sum_{\mathcal{V}_{<(i_1\leftrightarrow i)}} \prod_{\phi\in\mathcal{F}_{<(i_1\leftrightarrow i)}} \phi) \cdots \prod_{\phi\in\mathcal{F}_{<(i_m\leftrightarrow i)}} \phi) \prod_{\phi\in\mathcal{F}_i} \phi =$$

After reordering the sums and products:

$$= \sum_{Y_i}(\prod_{\phi\in\mathcal{F}_i} \phi) \sum_{\mathcal{V}_{<(i_1\leftrightarrow i)}} (\prod_{\phi\in\mathcal{F}_{<(i_1\leftrightarrow i)}} \phi) \cdots \sum_{\mathcal{V}_{<(i_m\leftrightarrow i)}} (\prod_{\phi\in\mathcal{F}_{<(i_m\leftrightarrow i)}} \phi) =$$

Using the induction step:

$$= \sum_{Y_i} \psi_i \delta_{i_1 \to i} \dots \delta_{i_m \to i} = \delta_{i \to j}$$

Corollary $\beta_i = \tilde{P}(Scope(C_i)) = \sum_{\mathcal{X} \setminus Scope(C_i)} \prod_{\phi} \phi$

Proof From the definition of belief:

$$\beta_i = \psi_i \prod_{k \in Nb_i} \delta_{k \to i} =$$

Using the theorem above:

$$= \psi_i \prod_{k \in Nb_i} \sum_{\mathcal{V}_{<(k \leftrightarrow i)}} \prod_{\phi \in \mathcal{F}_{<(k \leftrightarrow i)}} \phi =$$