Exact Inference on non Tree Graphs

In the previous lecture we saw how inference was tractable for a graphical model when the underlying graph G was a tree. We also focused on the case of a pairwise MRF.

Now what happens when we have a pairwise MRF that is not a tree? One answer we saw before is that we need to choose an elimination error that does not produce large factors (or equivalently does not generate large cliques in the induced graph). We called the largest such factor the tree-width of the model. So one approach to non-tree inference could be to seek a "good" elimination order for the inference problem we want to solve. This approach is ok, but if we want to solve multiple inference problems (e.g., calculate the marginals $p_i(x_i)$ for all variables) it would be nice if we could use something like the belief propagation (BP) algorithm. But using BP on a non-tree graph will generally not result in exact marginals, and in fact may not converge.

1 A Simple Example - From Square to Edge

What we will do is take our original graphical model and turn it into a graphical model that does have a tree shaped graph. On this graph we will be able to run BP and get exact results. Let's consider a simple example. Say our original graph was a square on four binary variables, with edges $(1,2),(2,3),(3,4),(4,1)$. This is not a tree, and in fact has tree-width two. The model is:

$$p(x_1,x_2,x_3,x_4) \propto \phi_{12}(x_1,x_2)\phi_{23}(x_2,x_3)\phi_{34}(x_3,x_4)\phi_{14}(x_1,x_4) \tag{1}$$

Let's turn it into a tree model. Introduce new variables z_1, z_2. The variable z_1 will correspond to the variables x_1, x_2, x_3 and will thus have 8 possible values. The variable z_2 wil correspond to variables x_2, x_3, x_4 and will thus also have 8 possible values. Denote by $x_1(z_1)$ the value of x_1 corresponding to assignment z_1 and likewise for other x_i and z_i (note that $x_4(z_1)$ is not defined). Also assume that z corresponds to the binary representation of its variables, so the mapping is:

z_1	x_1	x_2	x_3
0	0	0	0
1	0	0	1
$\vdots$			
7	1	1	1

We now define potentials $\psi_1(z_1), \psi_2(z_2)$ and $\psi_{1,2}(z_1,z_2)$ and a new model:

$$q(z_1,z_2) \propto \psi_1(z_1)\psi_2(z_2)\psi(z_1,z_2) \tag{2}$$

and we will want this model to be "isomorphic" to our original model. The first natural thing to do is to have each ψ_i take the potentials from the original graph. Thus we will define:

$$\psi_1(z_1) = \phi_{12}(x_1(z_1),x_2(z_1))\phi_{23}(x_2(z_1),x_3(z_1)) \tag{3}$$

And similarly:

$$\psi_2(z_2) = \phi_{34}(x_3(z_2),x_4(z_2))\phi_{14}(x_1(z_2),x_4(z_2)) \tag{4}$$

So it's now tempting to say that $\psi_1(z_1)\psi_2(z_2)$ is the same as the original model. But that's not quite true. Note that x_3 appears in both z_1 and z_2 (and likewise for x_2). But there is currently nothing forcing this x_3 to have the same values. To show why this is a proble

Consider a case where $z_1 = 0$ (namely corresponds to $x_1 = 0, x_2 = 0, x_3 = 0$) and $z_2 = 2$ (namely corresponds to $x_2 = 0, x_3 = 1, x_4 = 0$). Then we have:

$$\psi_1(0)\psi_2(2) \propto \phi_{12}(0,0)\phi_{23}(0,0)\phi_{34}(1,0)\phi_{14}(0,0) \tag{5}$$

In the original model we would never have such a combination! Thus currently our propose model contains invalid assignments. There is however a simple way of avoiding these assignments: add a term that "blocks" cases where z_1, z_2 are inconsistent. Namely, we define a pairwise term:

$$\psi_{1,2}(z_1, z_2) = \begin{cases} 0 & x_3(z_1) \neq x_3(z_2) \text{ or } x_2(z_1) \neq x_2(z_2) \\ 1 & \text{o}therwise \end{cases} \tag{6}$$

With this, we can now define our new model:

$$q(z_1, z_2) \propto \psi_1(z_1)\psi_2(z_2)\psi(z_1, z_2) \tag{7}$$

Under q, all assignment to z_1, z_2 that are inconsistent will get a value zero. Any other consistent assignment is completely equivalent to an assignment over the original four x_i variables. In other words, q will have exactly 2^4 non zero assignments, and these will be equal to the corresponding assignments in the original MRF. For example:

$$\begin{aligned} q(0,0) &\propto \phi_{12}(0,0)\phi_{23}(0,0)\phi_{34}(0,0)\phi_{14}(0,0) \propto q(0,0,0,0) \\ q(7,7) &\propto \phi_{12}(1,1)\phi_{23}(1,1)\phi_{34}(1,1)\phi_{14}(1,) \propto q(0,0,0,0) \\ q(0,2) &= 0 \end{aligned}$$

Since q is a tree pairwise MRF, we can use BP to calculate exact marginals for its variables. So we can calculate $q_1(z_1)$. But by construction:

$$q_1(z_1) = p(x_1(z_1), x_2(z_1), x_3(z_1)) \tag{8}$$

so we have the marginal $p(x_1, x_2, x_3)$ from which we can also calculate singleton marginals.

2 Constructing Junction Trees

We will want to repeat this procedure, but now for a general graph. We start with a given pairwise MRF on graph G with variables $x_1, \ldots, x_m$. We could like to define new variables $z_1, \ldots, z_m$ such that each z_i corresponds to a set of variables S_i (which we will call cliques) in the original graph (i.e., S_i is a set of x variables. In the example above $S_1 = \{1, 2, 3\}$). We would then like to define a new model:

$$q(z_1, \ldots, z_m) \propto \prod_{i=1}^{m} \psi_i(z_i) \prod_{ij \in \bar{E}} \psi_{ij}(z_i, z_j) \tag{9}$$

For this model to be isomorphic to the original one, and facilitate exact inference, we will require:

- The set of edges $\bar{E}$ forms a tree graph.
- The $\psi_i(z_i)$ are the product of the pairwise terms in the original MRF for pairs of variables in S_i.
- The potentials $\psi_{ij}(z_i, z_j)$ enforce consistency between z_i and z_j for variables in $S_i \cap S_j$.

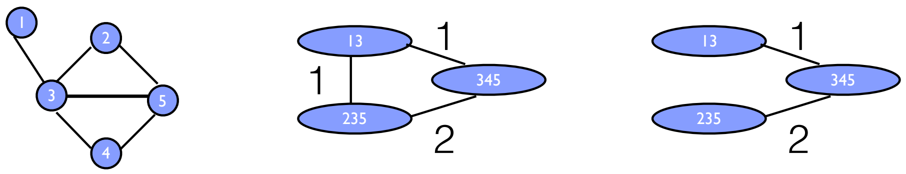

Figure 1: Example of junction tree construction algorithm. Left: induced graph. Middle: Clique graph with edge weights. Right: Maximum weight spanning tree that is a junction tree. Note that the tree corresponding to weights $1, 1$ is not a junction tree since it doesn't satisfy RIP for variable 5.

- For any x_j the set of z_i that contain it form a connected component. Otherwise, we could have two inconsistent "copies" of x_j. This is known as the Running Intersection Property (RIP).

A tree $\bar{E}$ that satisfies the above is known as a junction tree. In what follows, we provide a method for constructing such a tree.

We will begin with the induced graph G_π for a given elimination order π. Refer to the slides from lecture 2 to see how this graph is formed. Also, recall that the complexity of inference is exponential in the size of the largest clique in this graph.

Now consider the set of maximal cliques of G_π. Denote these by $S_1, \ldots, S_m$. These will correspond to our new variables z_i. So we are now considering a new graph with m edges corresponding to these cliques. What remains to be shown is that we can find a tree $\bar{E}$ on these nodes that satisfies the RIP property.

Recall that an edge between S_i and S_j in the junction tree is meant to enforce the constraint that the variables in these cliques are consistent. Thus there is no point connecting an edge between two cliques that have no common variables. This naturally leads to the notion of a clique graph, which is a graph over the S_i with an edge between S_i and S_j for cliques such that $S_i \cap S_j \neq \emptyset$. Denote the clique graph for G by C_G. It turns out that the clique tree for G_π has a subgraph that is a junction tree.

Theorem 2.1. *The clique tree of G_π has a sub-graph that is a junction tree.*

Proof. This can be shown by induction. For $n = 1$ variables it is obvious. Now assume it is true for n variables. We want to show it holds for any elimination order over $n+1$ variables. Consider the clique graph for G_π of these $n+1$ variables. Now assume wlog that the first eliminated variable is 1. Then it and all its neighbors in G_π form a clique (since eliminating it connected all of them). Now eliminate 1 and all of its outcoming edges from G_π for get $G_{\pi\setminus 1}$. The resulting graph is also an induced graph for the ordering π without the first element (but where the initial graph is G plus the factors generated by eliminating node 1). Thus we can use the inductive hypothesis to conclude that $G_{\pi\setminus 1}$ has a junction tree.

We now take the junction tree for $G_{\pi\setminus 1}$ and use it to build a junction for G_π. Assume first that the neighbors of node 1 are a maximal clique in $G_{\pi\setminus 1}$. Then we can use the same junction tree from $G_{\pi\setminus 1}$ in G_π. Otherwise, assume the neighbors of 1 are part of a larger maximal clique S in $G_{\pi\setminus 1}$. This will also be a maximal clique in G_π, and G_π will also have a clique S' with 1 and all its neighbors. Note that S and S' will have a connecting edge in the clique tree for G_π, since they have a non-empty intersection (namely the neighbors of 1). The junction tree for G_π will simply be the junction tree for $G_{\pi\setminus 1}$ with the added edge (S, S'). Since 1 doesn't appear in any other clique, it satisfies the RIP. And so do the neighbors of i since every path they appear in is connected to S. $\square$

Now that we know each induced graph has a junction tree, we present a simple algorithm for finding it.

Algorithm 2.2. *The junction tree construction algorithm. Given an induced graph G_π, do the following:*

1. *Construct a clique graph C for G_π. To each edge S_i, S_j in the clique graph assign a weight $w_{ij} = |S_i \cap S_j|$. Namely the size of the intersection of the two cliques.*
2. *Return a maximum weight spanning tree in the clique graph.*

Theorem 2.3. *Algorithm 2.2 returns a valid junction tree.*

Proof. The proof borrows from lecture notes by Michael Jordan and Devavrat Shah. Given a candidate spanning tree T over the clique tree, let's see what its overall weight is. Denote by V the set of variables in the MRF.

$$\begin{aligned}\sum_{ij\in T} w_{ij} &= \sum_{ij\in T} |S_i \cap S_j| \\ &= \sum_{ij\in T}\sum_{v\in V} I[v \in S_i \cap S_j] \\ &= \sum_{v\in V}\sum_{ij\in T} I[v \in S_i \cap S_j]\end{aligned}$$

where $I[A]$ is 1 iff the condition in A is satisfied. For a given v the expression on the RHS is the number of clique intersections that v participates in, in the tree T. Denote by n_v the number of cliques v belongs to. Then we are only considering edges in T that touch these nodes. This cannot be more than $n_v - 1$ (since the maximum number of edges in a tree is number of nodes minus one). Also, we will have equality if and only if all the nodes are connected. In other words, we will have equality if and only if the RIP is satisfied for variable v, and that will happen if and only if T is a junction tree! Thus we can write:

$$\sum_{ij\in T} w_{ij} \leq \sum_{v\in V}(n_v - 1) = \sum_{v\in V} n_v - |V| = \sum_i |S_i| - |V|$$

We can conclude that if a junction tree exists, only it can maximize the sum of weights. Since we know there exists a junction tree, we conclude that Algorithm 2.2 will return it. □

To summarize, we have shown how one can take a non-tree graph, and convert it into a tree of cliques over which exact inference can be performed. The size of the largest clique determine the complexity of the inference operation. The best we can hope for in terms of clique size is the tree-width of the graph, but unfortunately there are no efficient algorithms for finding the elimination order that will provide this in the general case. Although there are heuristics that often work well in practice.